

Temat : *Wzorzec behawioralny Command*  
*Wzorzec strukturalny Composite*

**Historia zmian**

<i>Data</i>	<i>Wersja</i>	<i>Autor</i>	<i>Opis zmian</i>
19.04.2010	1.0	Tomasz Kowalski	Utworzenie dokumentu i wprowadzenie zadania laboratoryjnego.
04.04.2011	1.1	Tomasz Kowalski	Modyfikacja treści. Zmiana zestawu wzorców.
17.05.2011	1.2	Tomasz Kowalski	Drobne poprawki.
2.03.2012	1.3	Tomasz Kowalski	Uwzględnienie zmian w hierarchii sterowników
7.05.2013	2.0	Tomasz Kowalski	Aktualizacja w związku z reorganizacją laboratorium
9.4.2015	3.0	Dominik Żurek	Aktualizacja aplikacji i zmiana nazw na język angielski
6.10.2016	3.1	Tomasz Kowalski	Zmiana repozytorium z svn-a na github

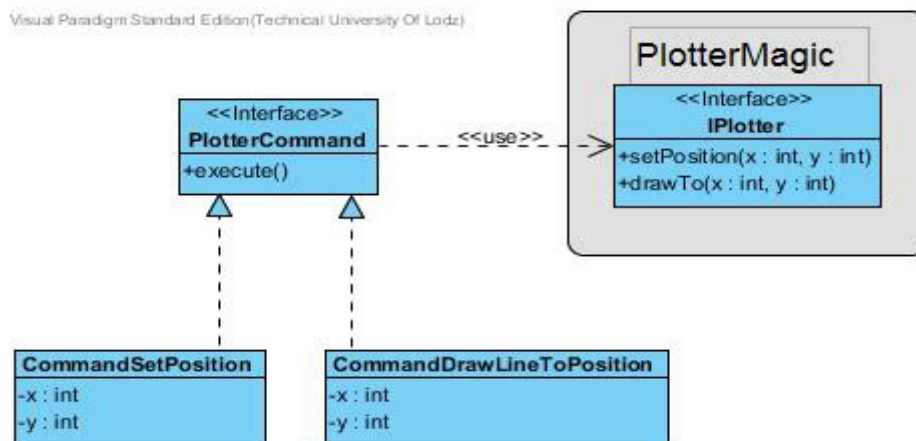
# 1. Cel laboratorium

Głównym celem laboratoriów jest zapoznanie się z wzorcami projektowymi: Command (polecenie), Composite (kompozyt). Należą one do różnych grup wzorców projektowych (behawioralne i strukturalne). Zajęcia powinny pomóc studentom rozpoznawać omawiane wzorce w projektach informatycznych, samodzielnie implementować wzorce oraz dokonywać odpowiednich modyfikacji wzorca w zależności od potrzeb projektu.

## 2. Laboratorium:

1. Laboratorium jest kontynuacją poprzednich laboratoriów dotyczących wzorca **adapter**. Rozwiązanie poniższych zadań należy utworzyć w nowych pakietach, których nazwy będą odpowiadać stosowanym wzorcom projektowym (np. *edu.iis.powp.command*).
2. Wyniki prac nad oprogramowaniem **PlotterMagic** uzyskane dzięki zastosowaniu wzorca projektowego adapter są do tej pory jak najbardziej pozytywne. Skłoniły to szefostwo do próby rozszerzenia oprogramowania **PlotterMagic**. Łatwo jest dokonać specjalizacji adaptera do linii (*LinePlotterAdapter*) zaimplementowanego w poprzednim zadaniu. Niestety, nie jest to możliwe dla rzeczywistego sterownika plotera opartego na interfejsie *IPlotter*, do którego implementacji nie mamy dostępu. Konieczne (i jak się okaże przydatne) jest rozdzielenie hierarchii sterowników od hierarchii klas dostarczających nową funkcjonalność (tzw. hierarchii abstrakcji). Szef rozważa wykorzystać w tym celu wzorec **command**.
3. **UWAGA:** pod koniec zajęć wyniki prac na laboratorium muszą być każdorazowo oznaczane w repozytorium jako osobny *release*. Braki w tym zakresie są równoważne z brakiem obecności na zajęciach.

### 3.1. Projektowanie hierarchii poleceń



Ilustracja 1: Diagram UML z przykładowym zastosowaniem wzorca command

1. Według diagramu na ilustracji 1 zaprojektuj i zaimplementuj hierarchię klas reprezentujących podstawowe funkcje plotera. Podobnie jak w przypadku menadżerów plotera wykorzystaj do wykonywania poleceń zaimplementowaną w ramach poprzednich zadań hierarchię sterowników plotera. Każda implementacja *PlotterCommand* będzie korzystała ze sterownika poprzez interfejs *IPlotter*, można więc przyjąć, że metoda *execute* może przyjmować driver jako argument.
2. W hierarchii poleceń zaprojektuj klasę *ComplexCommand*, która reprezentowałaby ciąg poleceń.
3. \*Jakie wzorce występują w hierarchii poleceń, poprzez fakt posiadania klasy

### *ComplexCommand?*

4. Zaimplementuj kilka „fabryk”, które będą zwracały złożone polecenie rysujące figurę (np. prostokąt, okrąg).
5. \*Zaimplementuj klasę umożliwiającą generację złożonego ciągu poleceń, z których składają się przykładowe figury rysowane przez skrypty z klasy *FiguresJoe*. Za każdy wykorzystany do rozwiązania wzorzec projektowy otrzymasz \* :).
6. \*Jakie jeszcze wzorce (jawnie nie nazwane do tej pory) można odnaleźć w projekcie?