Rozdział 4

Zasada wzorca i kopii – o podobieństwie metod stosowanych w rzemiośle, filozofii i programowaniu

Jakub Jernajczyk
Akademia Sztuk Pięknych im. Eugeniusza Gepperta we Wrocławiu

Streszczenie. Zasada wzorca i kopii, jako niezwykle efektywny model działania, stanowi metodologiczny fundament dla wielu obszarów ludzkiej aktywności: od sztuki i rzemiosła począwszy, poprzez technikę i projektowanie, na programowaniu i filozofii skończywszy. Twierdzę, że właśnie w owej zasadzie należałoby upatrywać źródła zaskakujących podobieństw, jakie dostrzegamy pomiędzy paradygmatem programowania obiektowego i Platońską teorią idei. W artykule opisuję również, bazujące na zasadzie wzorca i kopii, cyfrowe narzędzia graficzne, służące do tworzenia rzeczywistości wirtualnych, a także wybrane elementy systemów filozoficznych Arystotelesa oraz Plotyna. Całość zwieńczona została epistemologiczną refleksją dotyczącą możliwych przyczyn zaistnienia rozważanych tu analogii.

Słowa kluczowe: teoria idei, rzemiosło, wzorzec, programowanie obiektowe, klasa, metoda wirtualna

1. Wprowadzenie[1]

Tradycyjne rzemiosło, filozofia oraz programowanie to bardzo odległe obszary ludzkiej aktywności, które, jak się wydaje, nie powinny mieć ze sobą wiele wspólnego. Łączy je jednak to, że w ramach każdej z tych dziedzin wytwarzane są pewnego rodzaju przedmioty. Rzemieślnicy wytwarzają przedmioty materialne, efektem refleksji filozoficznej są przedmioty teoretyczne, zaś programi-

[1] Na wstępie pragnę podziękować organizatorom i uczestnikom *Seminarium naukowego z filozofii nauki* (Wydział Administracji i Nauk Społecznych Politechniki Warszawskiej), w ramach którego w kwietniu 2018 r. miałem okazję przedstawić i przedyskutować wiele spośród podejmowanych tutaj zagadnień. Dziękuję również redaktorom i recenzentom niniejszego tomu za cenne uwagi i sugestie, których uwzględnienie bez wątpienia wpłynęło korzystnie na jakość tego artykułu.

ści powołują do istnienia przedmioty wirtualne[2]. Pisząc o przedmiotach wirtualnych, zazwyczaj będę miał na myśli definicję informatyczną, która określa je jako ugruntowane obliczeniowo przedmioty intencjonalne [Stacewicz, 2019]. Trzeba jednak zaznaczyć, że w szerszym ujęciu przedmiotem wirtualnym określić możemy także każdy przedmiot, który jest teoretycznie możliwy, czyli może zaistnieć [zob. Kopaliński, 2007]. Takie potencjalne istnienie w szczególnie wyraźny sposób ujawnia się w przypadku przedmiotów, które wytwarzane są na bazie istniejących uprzednio wzorców, stanowiących niejako zapowiedź ich powstania. Dotyczy to zarówno zaawansowanych urządzeń technicznych, artystycznych odbitek z metalowej matrycy, złożonych elementów środowisk cyfrowych czy też prostych glinianych misek. Co więcej, wedle Platona każdy element świata fizycznego powstaje na bazie idealnego wzorca, zatem rozumiana tak szeroko wirtualność mogłaby stanowić cechę przynależną każdej rzeczy.

W literaturze z zakresu filozofii informatyki dyskutowany jest problem zaskakującego podobieństwa pomiędzy Platońską teorią idei a współczesnym paradygmatem programowania obiektowego [zob. np. Rayside, Campbell, 2000; Janusz, 2002; Giguette, 2006; Tylman, 2018][3]. Analogia ta stanowi ogromną pokusę dla spekulatywnych umysłów. Sugeruje bowiem, że twórcy nowoczesnych narzędzi programistycznych mogli zaczerpnąć swoje pomysły wprost ze starożytnej myśli Platona, dowodząc tym samym, że ta wysoce abstrakcyjna, pozornie oderwana od rzeczywistości ontologia, może być podłożem najbardziej efektywnego, mającego niezliczone praktyczne zastosowania paradygmatu programowania. Nie sposób zaprzeczyć, że istnieją wyraźne podobieństwa pomiędzy wskazanymi dziedzinami – przyjrzymy się im dokładniej w kolejnych częściach artykułu. Wydaje się jednak, że zamiast orzekać o bezpośrednim wpływie jednego obszaru na drugi, należałoby raczej szukać ich wspólnego źródła, sprawiającego, że nawet w tak odległych dziedzinach ujawniają się podobieństwa metodologiczne. Takim właśnie wspólnym fundamentem zdaje się być tytułowa zasada wzorca i kopii, która, poza wspomnianą już filozofią i programowaniem, obecna jest również w tradycyjnym rzemiośle, technice, szeroko pojętym projektowaniu oraz w sztuce, zarówno tej tradycyjnej, jak i cyfrowej.

Artykuł ten rozpoczniemy od ogólnej charakterystyki zasady wzorca i kopii. Następnie przyjrzymy się wybranym obszarom, w których zasada ta się ujawnia. Na koniec zastanowimy się również nad jej epistemologicznym i ontologicznym znaczeniem.

---

[2] Porównania wybranych cech przedmiotów realnych, wirtualnych a także fikcyjnych dokonuje w niniejszym tomie Bondecka-Krzykowska [2019].

[3] Temat ten podjąłem również w pracy [Jernajczyk, 2008], dotyczącej związków sztuki mediów z filozofią.

2. Istota oraz źródło zasady wzorca i kopii

Zasadę wzorca i kopii charakteryzują cztery proste punkty:
1) na bazie jednego wzorca powstaje wiele kopii, które zachowują wszystkie własności wzorca;
2) modyfikacja wzorca powoduje zmianę jego kolejnych kopii;
3) modyfikacja pojedynczej kopii nie wpływa ani na wzorzec, ani na pozostałe kopie;
4) na bazie jednej z kopii można wytworzyć nowy wzorzec, w którym pewne własności ulegną zmianie bądź pojawią się całkiem nowe.

Źródeł tak określonej zasady nie należy szukać w nowoczesnym programowaniu czy nawet w starożytnej myśli filozoficznej, lecz raczej w bardziej pierwotnych intuicjach, które pozwalały rzemieślnikom i artystom[4] osiągać kunszt na długo zanim rozwinęła się myśl naukowa. Z zasadą wzorca i kopii spotykamy się wszędzie tam, gdzie na bazie jednej formy tworzone są liczne egzemplarze danej rzeczy. W sensie metodologicznym takie podejście pozwala osiągać maksymalną wydajność oraz niezawodność – podchodząc bowiem ponownie do danego zagadnienia, nie musimy za każdym razem szukać nowych rozwiązań, lecz możemy odwołać się do sprawdzonych wzorów. Nic więc dziwnego, że zasada wzorca i kopii stała się fundamentem nowożytnej, opartej na schematach myśli technicznej[5]; obecna jest również w wielu współczesnych dyscyplinach projektowania oraz sztuki, takich jak wzornictwo, ceramika, grafika warsztatowa oraz grafika cyfrowa.

Według J. D. Boltera w każdej epoce wskazać można techniki bądź technologie, które wyjątkowo mocno pobudzają ludzką wyobraźnię i oddziałują na różne sfery kultury i nauki, w tym również na myśl humanistyczną. W okresie starożytnym takie szerokie wpływy przypisuje on technikom rzemieślniczym, w szczególności zaś wrzecionu i kołu garncarskiemu [Bolter, 1990, s. 49–56][6]. O tym, jak znaczący wpływ na myśl greckich filozofów miało rzemiosło, świadczą liczne nawiązania, które znaleźć możemy w ich pismach. Platon – twórca teorii idei, która w znacznej mierze opiera się na zasadzie wzorca i kopii, w centrum swojej kosmologii postawił Demiurga –

---

[4] W starożytności nie istniało wyraźne rozróżnienie pomiędzy rzemiosłem a tym, co dziś określamy mianem sztuk pięknych [zob. Tatarkiewicz, 1985, s. 37–38].

[5] O tym jak wyraźny wpływ na rozwój cywilizacji europejskiej miała umiejętność tworzenia planów i rysunków technicznych pisze Kordos [2006, s. 52–54].

[6] Więcej na temat Bolterowskiej koncepcji *technik definiujących*, stanowiących ważne „źródło metafor poznawczych", pisze Polak [2015].

boskiego Rzemieślnika, który zbudował świat na bazie idealnych form [zob. Platon *Timajos*, 29a]. Arystoteles, tłumacząc kluczową dla swojego systemu teorię aktu i możności, wielokrotnie odwołuje się do pracy artystów i rzemieślników, którzy wydobywają z materii kształty, potencjalnie już w niej zawarte [zob. np. Arystoteles *Metafizyka*, IX 6, 1048a]. Z kolei w pochodzącym z połowy XV wieku dialogu Mikołaja z Kuzy *Laik o umyśle*, tytułowy prosty rzemieślnik, trudniący się sztuką wytwarzania łyżek, poprzez odniesienia do swojej profesji, tłumaczy istotę różnic zachodzących pomiędzy systemami filozoficznymi Platona i Arystotelesa [zob. Mikołaj z Kuzy *Laik o umyśle*, II, 62–68]. Przykłady te zdają się wspierać tezę, iż filozofowie zaczerpnęli istotną dla swych koncepcji zasadę wzorca i kopii ze sztuki rzemieślniczej. Do zagadnień tych powrócimy w dalszej części artykułu. Najpierw jednak zobaczmy, w jakim stopniu owa zasada realizowana jest w zakresie programowania obiektowego i projektowania nowoczesnych narzędzi cyfrowych, używanych do tworzenia przedmiotów wirtualnych[7].

3. Klasy i obiekty

Programowanie obiektowe należy do najbardziej efektywnych i najczęściej stosowanych modeli programowania. Ścisła definicja tego paradygmatu nastręcza jednak wiele trudności. Teoretycy oraz praktycy wskazują bowiem na różne cechy jako te najważniejsze, najbardziej odkrywcze, stanowiące o istocie obiektowości. W różny sposób określane są też warunki konieczne i wystarczające tego paradygmatu[8]. Dlatego też w tym miejscu ograniczę się do opisu jedynie tych cech programowania obiektowego, które są istotne w kontekście prowadzonych tutaj rozważań, pamiętając przy tym, że niektóre z owych charakterystyk odnoszą się także do starszych paradygmatów programowania[9].

Zacznijmy od tzw. *hermetyzacji*, określanej również mianem *enkapsulacji* bądź *kapsułkowania*. Mówiąc obrazowo, hermetyzacja polega na zamknięciu danych oraz działających na tych danych operacji wewnątrz abstrakcyjnych kapsuł [zob. Subieta, 1999]. Owe kapsuły to właśnie obiekty, które posiadają swą wewnętrzną strukturę oraz funkcjonalność i komunikują się ze środowi-

[7] O inżynierskich aspektach wytwarzania przedmiotów wirtualnych piszą w tym tomie Brzeziński i Lubacz [2019].

[8] Pewną wersję „minimum języka obiektowego" proponuje Janusz [2002, s. 167–168].

[9] Historię rozwoju paradygmatów programowania opisuje Stroustrup [2002, s. 24–44].

skiem zewnętrznym za pomocą stosownego interfejsu, dzięki czemu pozostałe elementy środowiska (programu, w którym zaistniał dany obiekt) mają dostęp tylko do tych jego elementów (danych oraz operacji), które udostępnia interfejs. Sens hermetyzacji łatwo uchwycić, kiedy odwołamy się do przykładów naszych kontaktów ze złożonymi systemami w życiu codziennym. Chcąc np. uruchomić samochód, nie musimy przecież znać budowy tego skomplikowanego urządzenia czy też samodzielnie realizować wszystkich niezbędnych etapów inicjowanego procesu – wystarczy, że przekręcimy kluczyk, będący w tym przypadku właśnie interfejsem.

Skoro zamknęliśmy już dane i operacje w hermetycznej kapsule, możemy potraktować ją jako wzorzec, na podstawie którego tworzyć będziemy wiele różnych obiektów, charakteryzujących się podobną budową. W programowaniu obiektowym taka możliwość określana jest jako *definiowanie abstrakcyjnych typów* i najczęściej realizowana jest za pośrednictwem *klas*. Klasa stanowi wzorzec dla wszystkich wirtualnych obiektów należących do zdefiniowanego przez nią typu. Na bazie jednej klasy powołać możemy dowolną liczbę obiektów posiadających taką samą strukturę danych (określanych jako własności klasy) oraz operacji (określanych jako jej metody).

Ostatnią cechą programowania obiektowego, na którą zwrócimy tutaj uwagę[10] jest *dziedziczenie*. Klasy zbudowane na podstawie innych klas (tzw. nadklas czy też klas podstawowych) nazywa się klasami pochodnymi bądź podklasami. Dziedziczą one po swoich nadklasach własności oraz metody. W ten sposób powstaje *hierarchia klas*. Klasy podstawowe są bardziej abstrakcyjne i ogólne od swoich podklas (np. klasa *pojazd* jest nadklasą dla klas *samochód*, *rower* czy *furmanka*). Klasa pochodna nie może istnieć bez swojej nadklasy; nie zachodzi natomiast relacja odwrotna.

Złożone relacje pomiędzy klasami oraz ich obiektami stanowią osobny przedmiot rozważań natury ontologicznej. Co więcej, ponieważ siłą programowania obiektowego jest łatwość tworzenia struktur reprezentujących abstrakcyjne pojęcia, które nie dają się łatwo sprowadzić od klasycznych typów i operacji liczbowych [zob. Stroustrup, 2002, s. 17, 47], klasy i obiekty mogą się okazać subtelnym narzędziem, służącym do modelowania wielu zaawansowanych problemów filozoficznych [zob. Janusz, 2002, s. 175–185]. Zanim jednak przyjrzymy się bliżej związkom programowania obiektowego z filozofią, poświęćmy jeszcze chwilę uwagi narzędziom cyfrowym, które łączą w sobie cechy tradycyjnych narzędzi oraz technik programistycznych, dzięki którym powstały.

---

[10] W dalszej części artykułu omówione zostanie jeszcze jedno, kluczowe dla obiektowości zagadnienie *metod wirtualnych*.

4. Cyfrowe rzemiosło

Chociaż współczesne narzędzia cyfrowe służą do wytwarzania przedmiotów wirtualnych a nie fizycznych, stanowią one swoiste zamienniki klasycznych narzędzi rzemieślniczych. Nie dziwi zatem fakt, że również one bazują na zasadzie wzorca i kopii, która okazała się tak skuteczna w tradycyjnym rzemiośle. Za przykład niech posłużą nam środowiska przeznaczone do projektowania i edycji ruchomego obrazu cyfrowego, będącego nieodłącznym elementem światów wirtualnych[11]. W wielu tego typu programach mamy możliwość definiowania ogólnych wzorców oraz tworzenia ich licznych instancji[12]. Ponieważ każdy program posługuje się własnym nazewnictwem, aby nie mnożyć terminologii, zasadę wzorca i kopii omówię tu na przykładzie wybranego środowiska Adobe Animate, znanego we wcześniejszych wersjach jako Flash.

W programie Animate złożoną multimedialną strukturę, zawierająca animowane elementy graficzne oraz dźwięki, umieszcza się wewnątrz symbolu, który początkowo może istnieć tylko w bibliotece projektu, poza jego główną sceną (warstwą prezentacyjną). Mając zdefiniowany symbol, możemy umieszczać na scenie wiele jego instancji, które zachowywać będą wewnętrzną strukturę symbolu (wygląd oraz zachowania), ale mogą różnić się między sobą cechami zewnętrznymi, takimi jak położenie, wielkość, kolor itp. Podobnie jak w przypadku programistycznych obiektów oraz klas, zmiana cech zewnętrznych (parametrów) pojedynczej instancji nie wpływa na pozostałe instancje, natomiast każda zmiana wewnętrznych cech symbolu powoduje zmiany we wszystkich jego instancjach. Ponadto, usunięcie ze sceny pojedynczej instancji nie ma wpływu na inne instancje, podczas gdy usunięcie symbolu z biblioteki spowoduje usunięcie ze sceny wszystkich jego instancji.

Ponieważ koncepcja symboli oraz ich instancji bazuje bezpośrednio na koncepcji klas i obiektów[13], ujawniają się tu również inne mechanizmy obecne w programowaniu obiektowym. Podobnie jak klasy, które korzystają w swojej strukturze z obiektów innych klas, także symbole mogą zawierać w sobie instancje innych symboli. Mamy tutaj również pewien odpowiednik mechanizmu dziedziczenia – chcąc utworzyć nowy symbol, który zachowywał bę-

[11] Problematykę rzeczywistości wirtualnych podejmuje w tym tomie Bondecka-Krzykowska [2019].

[12] Dotyczy to np. środowiska Toom Boom Harmony oraz programów firmy Adobe: Premiere Pro, After Effects, Animate, czy, nie rozwijany już dzisiaj, Director.

[13] W środowisku Animate każdy symbol stanowi pewną klasę – dziedziczącą po ogólnej klasie *Object* [Braunstein, Wright, Noble, 2010, s. 215–216] – a instancje symbolu są obiektami tej klasy.

dzie wybrane cechy danego symbolu, ale pozwoli też na zmianę lub dodanie innych elementów, możemy utworzyć jego duplikat. Modyfikacja wewnętrznych cech zduplikowanego symbolu nie wpłynie na cechy oryginału.

Należy przy tym wyraźnie podkreślić, że cała opisana tu funkcjonalność symboli oraz ich instancji dostępna jest z poziomu graficznego interfejsu użytkownika i choć w warstwie programistycznej bazuje na strukturze obiektowej, może być obsługiwana przez projektanta bądź artystę w sposób czysto manualny, podobnie jak tradycyjne narzędzia.

5. Idee i rzeczy

W niniejszej części przyjrzymy się dokładniej roli, jaką zasada wzorca i kopii odgrywa w ontologii. Zacznijmy od ogólnej charakterystyki systemu Platona, w którym chyba najdobitniej uwidoczniło się znaczenie tej zasady.

5.1. Platońska teoria idei

Na wstępie trzeba zaznaczyć, że Platon posługiwał się terminem *idea* w znaczeniu *formy*, *wzorca* czy też *modelu*[14], podczas gdy nam słowo to kojarzy się raczej z pewną myślą lub koncepcją [por. Reale, 2012a, s. 88 oraz Sady, 2010, s. 87]. Tutaj jednak odwoływać się będziemy właśnie do tego Platońskiego rozumienia idei, jako wzorów dla wszystkich rzeczy: „…te postacie stoją w naturze jakby pierwowzory, a inne przedmioty są ich odwzorowaniami…" [Platon *Parmenides*, 132d]. Wedle Platona prawdziwą rzeczywistość stanowi świat wiecznych i niezmiennych idei, będący wzorcem dla świata fizycznego, który jest tylko jego niedoskonałym odbiciem, a wszystkie postrzegane zmysłami przedmioty są jedynie kopiami swych idealnych modeli. Elementy nauki o ideach rozsiane zostały po licznych pismach Platona. Jedno z jej najbardziej zwięzłych ujęć znaleźć można w liście siódmym [Platon *Listy*, 342a–343c], natomiast syntetyczną charakterystykę idei w odniesieniu do przedmiotów świata zmysłowego zawiera *Timajos*:

„Jeśli tak się rzeczy mają, trzeba się pogodzić z tym, że pierwszy rodzaj bytu przedstawia się zawsze w ten sam sposób, nie rodzi się ani nie ginie, nie przyj-

[14] Stosowane przez Platona greckie terminy ἰδέα oraz εἶδος wywodzą się od oznaczającego *widzenie* czasownika ἰδεῖν. Dosłownie znaczą one więc tyle, co *wygląd*, *kształt*, *postać* [Dembiński, 2003, s. 24].

muje w siebie znikąd czegoś innego, nie przechodzi nigdy w żadną inną rzecz, jest niedostrzegalny wzrokiem ani innymi zmysłami, sam tylko rozum jest w stanie go oglądać. Istnieje jeszcze drugi rodzaj, który nosi tę samą nazwę; jest podobny do tamtego, lecz jest postrzegalny zmysłowo, jest zrodzony, zawsze w ruchu, rodzi się w pewnym miejscu i znów z niego znika, jest przystępny mniemaniu złączonemu z postrzeganiem zmysłowym" [Platon *Timajos*, 52a].

W podobny sposób idee scharakteryzowane zostały w *Fedonie*: „…wobec tych [istot], które są niezmienne, nie ma niczego, czym mógłbyś je uchwycić poza racjonalnym wnioskowaniem. Tego typu [istoty] są więc niewidzialne i zobaczyć się ich nie da…" [Platon *Fedon*, 79a]. Do głównych cech idei należą zatem: *wieczność*, *niezmienność*, *niecielesność*, *inteligibilność* (poznawalność tylko rozumem) a także *jedność*, stojąca w opozycji do wielości rzeczy postrzeganych zmysłami, dla których idee stanowią wzorce. Ponadto idee są *samoistne* – nie są od niczego ani od nikogo zależne i istnieją same przez się [zob. Platon *Kratylos*, 386e].

5.2. Porównanie idei i klas

Dokonane wyżej ogólne charakterystyki paradygmatu programowania obiektowego oraz Platońskiego idealizmu pozwolą nam teraz wskazać podobieństwa zachodzące pomiędzy klasami oraz ideami (i odpowiednio: obiektami oraz rzeczami)[15]:

1. Zarówno klasy jak i idee stanowią wzorce dla obiektów oraz rzeczy, definiując ich wewnętrzną strukturę oraz sposób działania.
2. Istnieje zawsze jedna klasa danego typu oraz jedna idea danej rzeczy, natomiast powstałych na ich bazie obiektów oraz rzeczy zmysłowych może być wiele.
3. Zarówno klasy jak i idee są niezmienne, podczas gdy obiekty oraz rzeczy powstają i giną. To ostatnie stwierdzenie wymaga doprecyzowania: dana klasa istnieje przed uruchomieniem programu i w czasie jego wykonywania pozostaje niezmienna, natomiast obiekty tej klasy mogą być wielokrotnie tworzone i niszczone.

Podsumowując, możemy powiedzieć, że „pojęcie *klasy* w programowaniu obiektowym jest Platońskie w tym sensie, że w kontekście działania programu klasy istnieją przed obiektami (tak jak *idee* preegzystują względem rzeczy materialnych) i są używane jako wzory do wytwarzania obiektów" [Rayside, Campbell, 2000].

[15] Podobieństwo idei oraz klas omawia także Tylman [2018].

5.3. Hierarchia i dziedziczenie

Osobnego omówienia wymaga obecna w systemie Platona hierarchiczna struktura rzeczywistości. Z zachowanych pism wynika jasno, że zdaniem filozofa ponad światem zmysłowym istnieje świat idei, zaś pomiędzy tymi dwoma poziomami jest jeszcze pośredni świat bytów matematycznych, które podobnie jak idee są wieczne i niezmienne, ale tak jak przedmioty zmysłowe charakteryzują się wielością [zob. Arystoteles *Metafizyka*, A 6, 987b]. Co więcej, wedle tzw. *nauk niepisanych* Platona[16], ponad światem idei miałby istnieć jeszcze poziom pierwszych zasad, z *Jednem* jako zasadą najwyższą oraz następującą po nim *Nieokreśloną Dyadą*. Jeśli uwzględnimy nauki niepisane, Platońska struktura rzeczywistości przedstawia się następująco: 1) poziom pierwszych zasad, 2) świat idei[17], 3) świat bytów matematycznych, 4) świat bytów zmysłowych. Pomiędzy tymi poziomami zachodzi jednostronna zależność ontologiczna – „…poziom niższy nie może istnieć […] bez poziomu wyższego, ale nie odwrotnie" [Reale, 2012a, s. 158]. Z podobną zależnością mamy do czynienia w przypadku hierarchii klas w programowaniu obiektowym.

Poszukując jednak pełniejszej analogii, która uwzględniałaby również zasadę dziedziczenia, należałoby sięgnąć do neoplatońskiej filozofii Plotyna, wedle którego cechą konstytuującą każdy byt jest *jedność* [Plotyn *Enneady*, VI, 9, 1], przekazywana z najwyższego poziomu (pierwszej *hipostazy – Jedna samego w sobie*), poprzez poziomy pośrednie (drugą i trzecią *hipostazę*), aż do najniższego poziomu świata fizycznego[18]. Mamy tu więc do czynienia ze swoistym dziedziczeniem przez wszystkie byty cechy *jedności*, pochodzącej od *Jedna*, które jest zasadą najwyższą, najbardziej ogólną i zarazem najprostszą[19]. W kontekście rozważanych analogii warto zauważyć, że w wielu obiektowych językach programowania, takich jak C#, Java czy Ruby, wszystkie

---

[16] Uczeni toczą wciąż spory w kwestii znaczenia nauk niepisanych. Spotkać można zarówno stanowiska, uznające nauki niepisane za kluczowy element myśli Platona, niezbędny do zrozumienia całości jego systemu (przedstawicielem tego nurtu jest między innymi G. Reale), jak i takie, które marginalizują znaczenie nauk niepisanych [zob. Sady, 2010, s. 70–72]. Wyważone podejście proponuje B. Dembiński, uznając nauki niepisane za ostatni – najdojrzalszy etap ewolucji filozofii Platona [zob. Dembiński, 1999 oraz Dembiński, 2003].

[17] Na poziomie idei również panuje wewnętrzna hierarchia: na szczycie znajdują się liczby i figury idealne, potem następują idee najogólniejsze (tzw. *metaidee*), w końcu zaś idee szczegółowe [Reale, 2012a, s. 157–158].

[18] Obrazuje to słynna wizualna metafora światła, promieniującego z niewyczerpanego źródła w postaci kolejnych świetlistych kręgów [zob. Plotyn *Enneady*, VI, 3, 17]. Zaprezentowane tu syntetyczne ujęcie myśli Plotyna zostało opracowane na podstawie [Reale, 2012b, s. 511–573].

[19] Choć *Jedno* jest czymś absolutnie prostym [zob. Plotyn *Enneady*, V, 4, 1], ma ono nieskończoną moc, nieskończony potencjał [zob. Plotyn *Enneady*, VI, 9, 6].

klasy pochodne wyprowadzane są z jednej, najbardziej ogólnej klasy podstawowej, która, podobnie jak *Jedno* w systemie Plotyna, stanowi wspólne źródło dla całej struktury programu, a jej cechy dziedziczone są przez wszystkie klasy (byty) pochodne.

5.4. Forma kształtująca materię

Analogia pomiędzy systemem filozoficznym Platona oraz paradygmatem programowania obiektowego wyczerpuje się w momencie, kiedy weźmiemy pod uwagę samoistność idei oraz ich całkowitą transcendencję wobec świata. Oznacza to bowiem, że żaden umysł nie może wpływać na wieczne i niezmienne idee. Nawet Platoński Demiurg tworząc świat musiał opierać się na owych doskonałych i zupełnie od niego niezależnych modelach. Zatem, w przeciwieństwie do programistycznych klas, Platońskie idee z założenia są niestwarzalne – można próbować je poznać, lecz nie można ich tworzyć ani zmieniać [por. Janusz, 2002, s. 48]. Spróbujmy więc poszukać precyzyjniejszej analogii, która pozwoliłaby nam uwzględnić istotną rolę programisty – kreatora wirtualnej rzeczywistości programu. W tym celu sięgniemy do myśli Arystotelesa, który za zbędne uznał przyjmowanie idei, niezależnych od świata zmysłowego. Twierdził bowiem, że w każdej rzeczy zawarta jest forma, która kształtując materię, ustanawia jej jakość. Forma, podobnie jak Platońska idea jest niezmienna, ale nie stanowi odrębnego przedmiotu – nie istnieje poza rzeczą [zob. Arystoteles *Metafizyka*, VII 8, 1033b–1034a]. Oderwane od świata fizycznego idee zastąpione tu zostały inteligibilnymi formami tego co zmysłowe [Reale, 2012a].

Obok samej materii oraz samej formy Arystoteles wyróżnia także ich złączenie, stanowiące jego zdaniem odrębny rodzaj substancji [zob. Arystoteles *Metafizyka*, VIII 2, 1043a]. Właśnie w tym rozróżnieniu moglibyśmy poszukiwać pełniejszej analogii do programistycznych klas oraz obiektów: klasie odpowiadałaby inteligibilna forma, zaś powstałym na bazie klasy obiektom odpowiadałyby owe konkretne, uchwytne zmysłami połączenia formy i materii.

W ujęciu Arystotelesa ludzki intelekt wyabstrahowuje ogólne formy z poszczególnych, podpadających pod zmysły przedmiotów. Stąd też „możemy powiedzieć, że klasy w programowaniu obiektowym są Arystotelesowskie w takim sensie, że programista w obszarze danego zagadnienia ogarnia najpierw elementy jednostkowe (obiekty) a dopiero potem opracowuje zawierające je pojęcia abstrakcyjne (klasy)" [Rayside, Campbell, 2000]. Warto w tym kontekście dodać, że w ramach programowania obiektowego wyróżnia się dwa modele: oparty na klasach (realizowany przez takie języki jak C++, C#, PHP czy Java) oraz oparty na prototypach. W tym drugim modelu, realizowanym między innymi w językach JavaScript oraz

Python[20], nie definiuje się zewnętrznych klas, lecz pierwszy zdefiniowany bezpośrednio w kodzie programu obiekt danego typu stanowić może prototyp dla kolejnych tego typu obiektów. Można więc powiedzieć, że ogólna forma obiektów danego typu jest w tym wypadku wytwarzana na podstawie rzeczy jednostkowej.

W tym miejscu kończy się jednak podobieństwo do Arystotelesowskiej metafizyki, ponieważ postulowana przezeń forma, podobnie jak Platońska idea, nie może być tworzona przez poznający umysł. Szukając nurtu filozoficznego, który przyznawałby umysłowi moc wytwarzania pojęć ogólnych, należałoby zwrócić się w stronę rozwijającego się od średniowiecza nominalizmu [zob. Tylman, 2018].

6. Zasada wzorca i kopii jako metoda wirtualna

Dokonawszy porównania wybranych cech paradygmatu programowania obiektowego oraz koncepcji filozoficznych Platona i jego następców, rozważmy teraz dokładniej możliwe przyczyny wyraźnego podobieństwa, jakie zachodzi pomiędzy tymi, zdawałoby się bardzo odległymi obszarami. Na wstępie odrzuciłbym wariant całkowicie przypadkowej zbieżności, gdyż wydaje się on mało prawdopodobny, a nade wszystko najmniej interesujący. Należałoby też odrzucić możliwość bezpośredniej i świadomej inspiracji programistów koncepcjami filozoficznymi, ponieważ w literaturze specjalistycznej nie odnajdujemy takich nawiązań. Bardziej prawdopodobna jest inspiracja nieuświadomiona, oznaczająca niejawne odwołanie się twórców języków programowania do koncepcji, które po raz pierwszy tak jasno wyrażone zostały w filozofii Platona i przez wieki stanowiły fundament naszej kultury i nauki. Za takim wariantem przemawiać może skala wpływu teorii idei na całą filozofię europejską, która miałaby być – jak dobitnie stwierdził Whitehead – jedynie „serią przypisów do Platona" [Whitehead, 1978, s. 39][21].

Rozważmy jednak jeszcze inny wariant, zakładający, że na myśl filozofów, inżynierów, projektantów i programistów wpłynęła ta sama, bardzo podstawowa koncepcja, która już wcześniej stanowiła fundament dla sztuki i rzemiosła. Zasada wzorca i kopii – bo o niej rzecz jasna tu mowa – okazałaby

[20] Z uwzględnieniem dodatkowej biblioteki *prototype.py*

[21] O sile oddziaływania tej filozofii na myśl zachodnią pisze także Russell [2012, s. 137]. Z kolei B. Dembiński zauważa, że liczne nawiązania do Platona wybitnych przedstawicieli współczesnej nauki (między innymi W. Heiseberga i R. Penrose'a) świadczyć mogą „o szczególnej współzgodności myślenia" [Dembiński, 1999, s. 197].

się cywilizacyjnym archetypem, meta-zasadą przenoszoną przez wieki z dziedziny na dziedzinę, potwierdzającą swą niezwykłą skuteczność w różnych, często bardzo odległych obszarach ludzkiej aktywności.

Nawiązując do terminologii informatycznej moglibyśmy również określić zasadę wzorca i kopii *metodą wirtualną.* W programowaniu obiektowym za pomocą metod wirtualnych opisuje się operacje w klasach odpowiadających pojęciom najbardziej ogólnym [zob. Stroustrup, 2002, s. 18]. Projektując strukturę klas dla programu, metody wirtualne deklaruje się w klasach podstawowych. Na tak wysokim poziomie ogólności metody te nie mają jeszcze konkretnego zastosowania, dlatego też nie posiadają żadnej definicji. Jednak już w konkretnych klasach pochodnych przyjmować mogą różne ustalone formy – określone definicje [zob. Stroustrup, 2002, s. 39–43]. Chcąc np. stworzyć oprogramowanie służące do przeprowadzania operacji matematycznych, jedną z wirtualnych metod, której istnienie powinniśmy uwzględnić przy opracowywaniu najwyższych poziomów hierarchii klas, jest dodawanie. Choć będzie ono zupełnie inaczej realizowane w konkretnych klasach pochodnych, odpowiedzialnych za obliczenia na liczbach naturalnych, rzeczywistych, zespolonych bądź jeszcze innych strukturach (np. wektorach czy zbiorach), ma w sobie pewną ogólną, wspólną tym wszystkim klasom koncepcję sumowania dwóch obiektów tego samego typu [por. Janusz, 2007]. Metody wirtualne są zatem metodami potencjalnymi, mogącymi zaistnieć w danej hierarchii klas[22].

O zasadzie wzorca i kopii możemy myśleć właśnie jako o wirtualnej metodzie, charakteryzującej w sposób ogólny pewien typ myślenia i działania. Chociaż konkretne formy realizacji tej zasady różnią się znacząco w poszczególnych obszarach zastosowań (inaczej tworzy się kopie przedmiotów materialnych z fizycznej matrycy, inaczej powielane są obiekty cyfrowe, w inny jeszcze sposób musiałyby powstawać niedoskonałe odbicia inteligibilnych idei), na poziomie najbardziej ogólnym funkcjonuje wciąż ta sama koncepcja wytwarzania wielu elementów z pojedynczego modelu.

Czym jest (jeśli w ogóle jest) ów poziom ogólny, z którego wyłaniają się uniwersalne schematy naszego działania? Czy zasada wzorca i kopii jest echem jakiejś ogólniejszej zasady, ukrytej głęboko w tkance rzeczywistości, która sprawia, że „zhierarchizowaną strukturę informacji wpisaną w świat można łączyć wirtualnie w łańcuchy ogólnych metod" [Janusz, 2007]? Czy może zasada ta stanowi jedynie szczególną cechę ludzkiego umysłu, który nie zna bardziej wydajnego sposobu organizowania rzeczywistości, zarówno tej fizycznej, jak i wirtualnej? To zaledwie kilka spośród licznych pytań, które

[22] Widzimy tu więc odwołanie do szerokiej definicji „wirtualności", rozumianej jako teoretyczna możliwość zaistnienia [zob. Kopaliński, 2007].

dotykają kluczowych problemów ontologii i teorii poznania, a sprowokowane zostały przez rozważane tutaj analogie.

7. Zakończenie – zasada umysłu, czy zasada świata?

Zasada wzorca i kopii jawi się jako uniwersalna reguła, opisująca wytwarzanie powtarzalnych elementów na podstawie pojedynczego wzoru. Przykłady zaczerpnięte z wielu różnych dziedzin ludzkiej aktywności potwierdzają metodologiczną doniosłość tej zasady i nie pozwalają wątpić, iż jest ona jedną z podstawowych reguł, wedle których poznajemy i zmieniamy świat. Z drugiej jednak strony zasada wzorca i kopii nie jest wyłącznie domeną człowieka – jej powszechne działanie obserwujemy również w naturze (wystarczy choćby wspomnieć replikację łańcuchów DNA). Zatem rozważania dotyczące istoty owej zasady nie powinny ograniczać się jedynie do analizy przykładów ludzkiej działalności.

Poszukując fundamentalnych źródeł pochodzenia zasady wzorca i kopii należy wyodrębnić dwa podejścia: epistemologiczne i ontologiczne, przy czym to drugie jest rozszerzeniem pierwszego. Podejście pierwsze zakłada, że istnienie zasady wzorca i kopii świadczy jedynie o określonych cechach ludzkiego umysłu. Podejście ontologiczne akceptuje rzecz jasna doniosłość tej zasady w ludzkim poznaniu i działaniu, ale zakłada dodatkowo, że jest ona jakoś ugruntowana w strukturze rzeczywistości. Tym samym przyjmuje ono pewną postać platonizmu. Posługując się metaforą zaczerpniętą z informatyki, możemy powiedzieć, że w ramach podejścia ontologicznego zasada wzorca i kopii stanowi metodę wirtualną w obiektowo zaprojektowanym programie wszechświata [por. Janusz, 2002].

Rozważmy kilka argumentów, które przemawiają za lub przeciw dwóm wyróżnionym wyżej podejściom. Sama obecność oraz skuteczność zasady wzorca i kopii w różnych obszarach twórczej aktywności człowieka niczego jeszcze nie rozstrzyga, wzmacnia bowiem obydwie ścieżki argumentacji: w oczywisty sposób utwierdza nas w przekonaniu, że myślimy i działamy wedle tej reguły, ale może też świadczyć na korzyść jej głębokich związków z ontologiczną strukturą rzeczywistości. Tym, co pozwala śmielej spojrzeć w stronę interpretacji ontologicznej jest obserwacja świata przyrody, która dowodzi, że zasada wzorca i kopii nie jest wcale zależna od ludzkiego umysłu. Mogłaby więc wyłaniać się z jakiegoś podstawowego poziomu istnienia. Należy jednak pamiętać, że wszystko, co obserwujemy, przetwarzane jest przez nasz umysł. Zatem to, że dostrzegamy w przyrodzie metody podobne do tych,

którymi sami się posługujemy, nie musi jeszcze dowodzić uniwersalnego charakteru owych metod. Tendencja do zakładania istnienia ogólnych zasad, panujących na różnych poziomach rzeczywistości, wynikać może bowiem ze specyficznych uwarunkowań naszych zdolności poznawczych.

Nie mamy rozstrzygających argumentów, które pozwalałyby nam dowieść bądź odrzucić tezę o ontologicznych źródłach zasady wzorca i kopii. Z jednej strony świadomi jesteśmy tego, że nasz umysł narzuca na obserwowane zjawiska pewne własne kategorie. Z drugiej zaś strony sam umysł też jest częścią przyrody, nie można zatem wykluczyć, że sposób jego działania wynika z jakichś podstawowych, ogólnych reguł, wpisanych głęboko w tkankę rzeczywistości. Choć struktura świata, w którym żyjemy, wciąż pozostaje przed nami ukryta, znamy dobrze reguły rządzące tworzoną przez nas rzeczywistością wirtualną, ponieważ sami owe reguły wyznaczamy. Jedną z takich reguł, stanowiących ontologiczny fundament świata cyfrowego, jest zasada wzorca i kopii.

BIBLIOGRAFIA

1. Arystoteles, (2013), *Metafizyka*, przeł. K. Leśniak, Warszawa, PWN.
2. Bolter J.D., (1990), *Człowiek Turinga – kultura Zachodu w wieku komputera*, Warszawa, Państwowy Instytut Wydawniczy.
3. Bondecka-Krzykowska I., (2019), *Cechy obiektów rzeczywistości wirtualnej*, w tym tomie: Przedmioty wirtualne, Warszawa, Oficyna Wydawnicza Politechniki Warszawskiej, s. 24–35.
4. Braunstein R., Wright M.H., Noble J.J., (2010), *ActionScript 3.0 Biblia*, przeł. S. Rataj, Gliwice, Helion.
5. Brzeziński K., Lubasz J., (2019), *Skąd się biorą przedmioty wirtualne*, w tym tomie: Przedmioty wirtualne, Warszawa, Oficyna Wydawnicza Politechniki Warszawskiej, s. 11–23.
6. Dembiński B., (1999), *Teoria idei – ewolucja myśli Platońskiej*, Katowice, Wydawnictwo Uniwersytetu Śląskiego.
7. Dembiński B., (2003), *Późna nauka Platona – związki ontologii i matematyki*, Katowice, Wydawnictwo Uniwersytetu Śląskiego.
8. Giguette R., (2006), *Building objects out of Plato: applying philosophy, symbolism, and analogy to software design*, Communications of the ACM, 49 (10), s. 66–71.
9. Janusz R., (2002), *Program dla Wszechświata. Filozoficzne aspekty języków obiektowych*, Kraków, WAM.
10. Janusz R., (2007), *O metodach wirtualnych w paradygmacie obiektowym*, Zagadnienia Filozoficzne w Nauce XLI, s. 125–131.
11. Jernajczyk J., (2008), *Elementy filozofii w sztuce mediów dyskretnych*, Wrocław, Akademia Sztuk Pięknych im. Eugeniusza Gepperta we Wrocławiu (maszynopis).
12. Kopaliński W., (2007), *Słownik wyrazów obcych i zwrotów obcojęzycznych z almanachem*, hasło: wirtualny, Warszawa, Oficyna Wydawnicza RYTM.
13. Kordos M., (2006), *Wykłady z historii matematyki*, Warszawa, SCRIPT.

14. Mikołaj z Kuzy, (2008), *Laik o umyśle*, przeł. A. Kijewska, Kęty, Wydawnictwo Marek Derewiecki.
15. Platon, (1986), *Timajos*, przeł. P. Siwek, Warszawa, PWN.
16. Platon, (1987), *Listy*, przeł. M. Maykowska, Warszawa, PWN.
17. Platon, (1990), *Kratylos*, przeł. Z. Brzostowska, Lublin, Redakcja Wydawnictw KUL.
18. Platon, (2002), *Parmenides*, przeł. W. Witwicki, Kęty, ANTYK.
19. Platon, (2017), *Fedon*, przeł. R. Legutko, Kraków–Warszawa, Teologia Polityczna.
20. Plotyn, (2001), *Enneady IV-V*, przeł. A. Krokiewicz, Warszawa, AKME.
21. Plotyn, (2003), *Enneady VI*, przeł. A. Krokiewicz, Warszawa, AKME.
22. Polak P., (2015), *Sieć, software czy obliczenia naturalne? Jakie techniki definiują myślenie filozoficzne Homo informaticus?*, Studia Metodologiczne nr 34, s. 143–170.
23. Rayside D., Campbell G.T., (2000), *An Aristotelian Understanding of Object-Oriented Programming*, ACM SIGPLAN Notices, 35 (10), s. 337–353.
24. Reale G., (2012a), *Historia filozofii starożytnej*, t. II, przeł. E.I. Zieliński, Lublin, Wydawnictwo KUL.
25. Reale G., (2012b), *Historia filozofii starożytnej*, t. IV, przeł. E.I. Zieliński, Lublin, Wydawnictwo KUL.
26. Russell B., (2012), *Dzieje zachodniej filozofii*, Warszawa, ALETHEIA.
27. Sady W., (2010), *Dzieje religii, filozofii i nauki. Od Talesa z Miletu do Mahometa*, Kęty, Wydawnictwo Marek Derewiecki.
28. Stacewicz P., (2019), *Wirtualność w perspektywie obliczeniowej*, w tym tomie: Przedmioty wirtualne, Warszawa, Oficyna Wydawnicza Politechniki Warszawskiej, s. 36–46.
29. Stroustrup B., (2002), *Język C++*, przeł. J. Mincer-Daszewicz, Warszawa, Wydawnictwa Naukowo-Techniczne.
30. Subieta K., (1999), *Słownik terminów z zakresu obiektowości*, hasło: hermetyzacja, Warszawa, Akademicka Oficyna Wydawnicza PLJ.
31. Tatarkiewicz W., (1985), *Historia estetyki: Estetyka starożytna*, Warszawa, Arkady.
32. Tylman W., (2018), *Computer Science and Philosophy: Did Plato Foresee Object-Oriented Programming?*, Foundations of Science, 23 (1), s. 159–172.
33. Whitehead A., (1978), *Process and reality. An essay in cosmology*, New York, The Free Press.

The Principle of Model and Copies: On the Similarities of Methods Used in Handicrafts, Philosophy and Programming

A b s t r a c t

The rule of model and copies, which is surely an exceptionally effective mode of operation, offers a methodological foundation for many areas of human activity: from art and handicrafts, through engineering and design, to computer programming and philosophy. I claim that this principle appears to be a source of some surprising similarities observable in relation to the paradigm of object-oriented programming and Plato's theory of forms. In this article, I also describe some digital graphic tools based on this principle, such as are used to create virtual realities. Selected elements of the philosophical systems of Aristotle and Plotinus are also analysed. The article ends with some epistemological reflections on possible sources for the analogies discussed here.