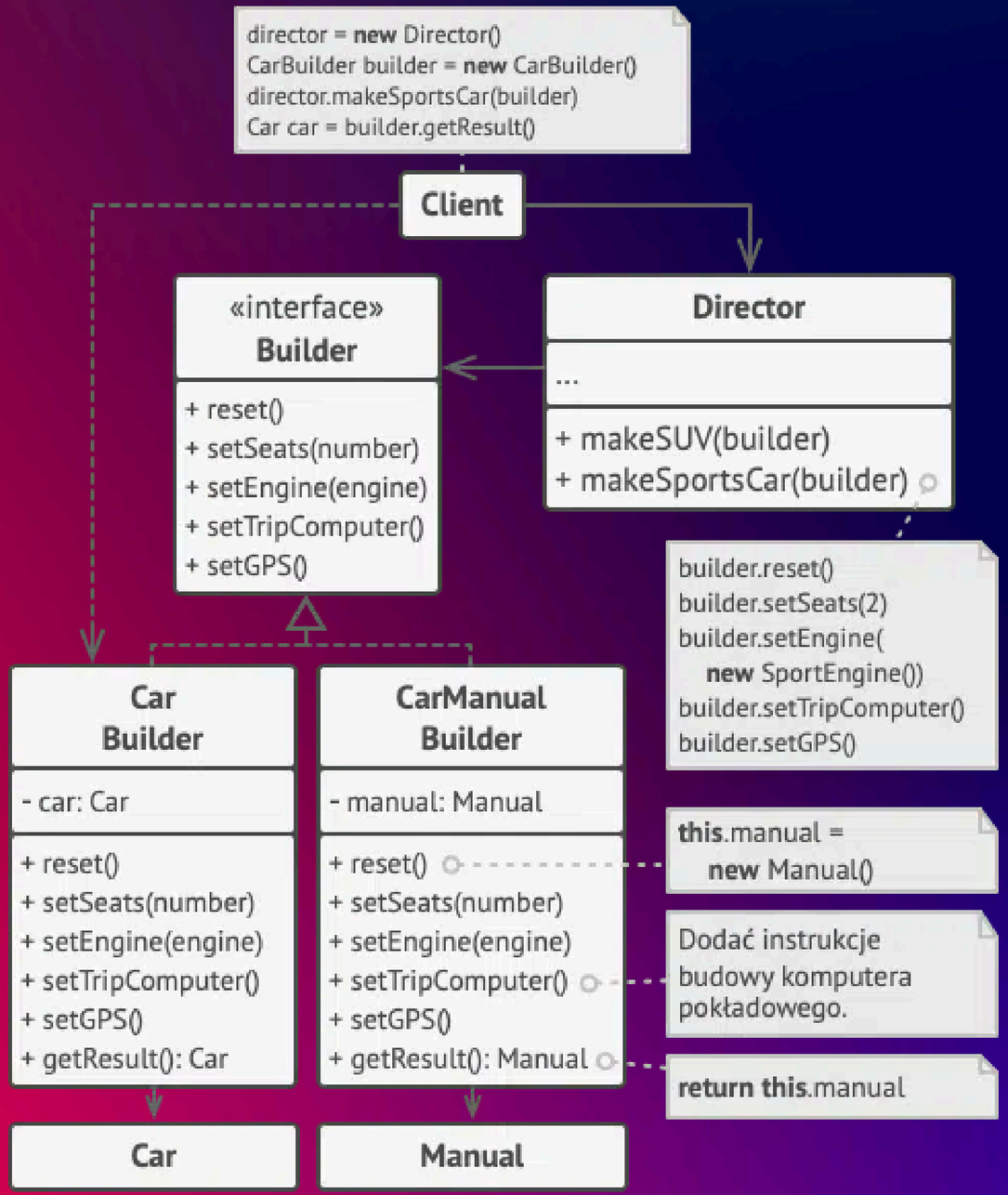
PROJEKTOWANIE SYSTEMÓW INFORMATYCZNYCH

7.11.2026



- Klasy są elementami składowymi modelowania strukturalnego (modelowanie jest abstrakcją rzeczywistości).
- Klasy stanowią abstrakcję obiektów, tj. stanowią wzorzec do ich tworzenia.
- Obiekty to jednostki strukturalne rzeczywistego systemu informacyjnego (system informacyjny opisuje rzeczywistość).
- Obiekt systemu informacyjnego jest dynamiczną czarną skrzynką; wchodzi w interakcje z podmiotami zewnętrznymi w celu świadczenia usług, ale ukrywa swoje wewnętrzne działanie.
- Wewnętrzna struktura obiektu jest znana tylko samemu obiektowi.

Bloki

Klasa



Interfejs



Klasa abstrakcyjna

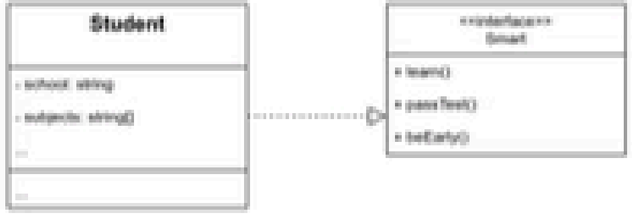


Relacje

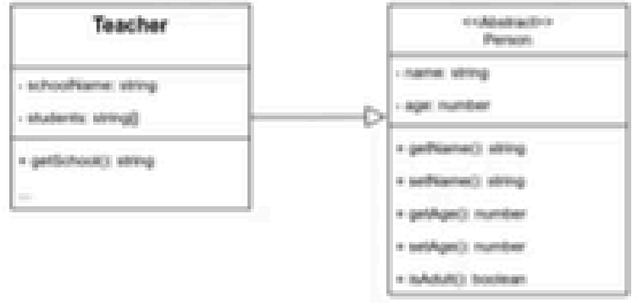
Zależność



Implementacja



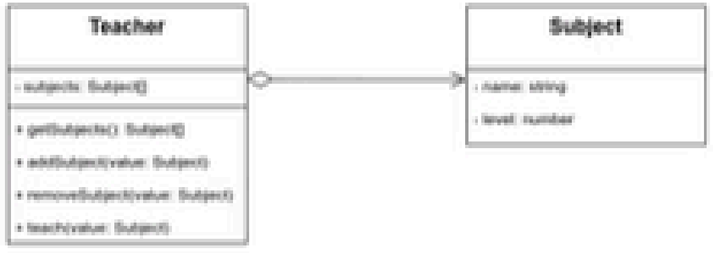
Dziedziczenie



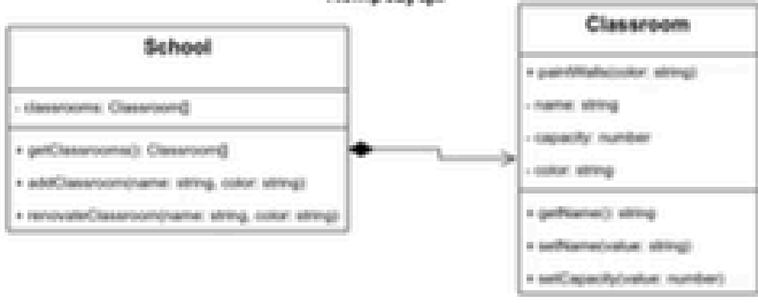
Asocjacja



Agregacja



Kompozycja



ENKAPSULACJA

Enkapsulacja to zamknięcie danych i procesów w ramach jednej jednostki.
Enkapsulacja umożliwia obiektowi egzekwowanie reguł biznesowych z określonym i uprawnieniami.
W wyniku enkapsulacji powstają dwie przestrzenie:

- Prywatna. Dane i procesy, które znajdują się wewnątrz obiektu, są oznaczone jako "prywatne".
- Publiczna. Cokolwiek obiekt eksponuje - to znaczy czyni widocznym dla świata zewnętrznego - jest "publiczne".



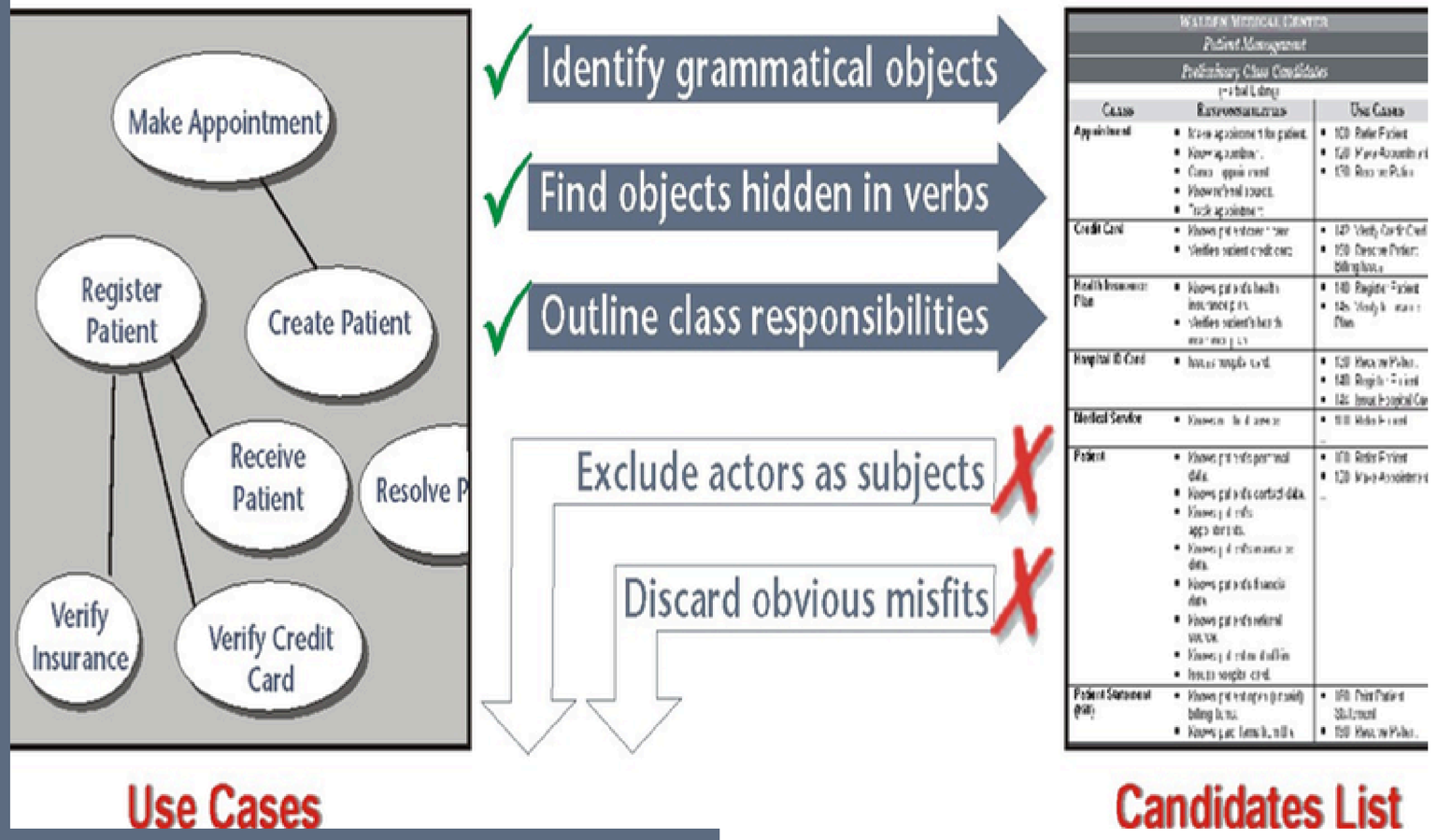


Widoczność

Symbol	Visibility	Description
+	Public	The attribute or operation is visible to all entities.
-	Private	The attribute or operation is private and cannot be (directly) accessed by outside entities.
#	Protected	The attribute or operation is available only to the object or its descendants.
~	Package (Friend)	Only other objects in the package (or component) can use the attribute.

Finding Class Candidates

Parsing the Behavior for Structure



Aby odkryć obiekty biznesowe, musimy zacząć od analizy scenariusza przypadków użycia.

Komunikaty wymieniane między aktorami a systemem odnoszą się do obiektów, na które wpływa interakcja między nimi:

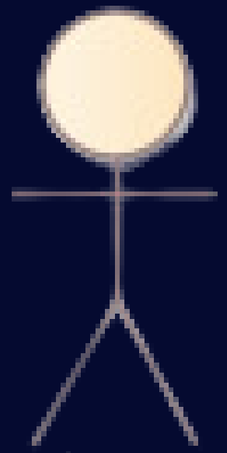
- analizując komunikaty (aktywności), które określają kroki w scenariuszu przypadku użycia, możemy rozpocząć odkrywanie klas.

Odkrywanie kandydatów na klasy

Utwórz nową wizytę	**Scenariusz podstawowy:**	1. **Pracownik ds. umawiania wizyt sprawdza, czy potrzebne świadczenie medyczne jest realizowane przez szpital.** 2. **Pracownik działu rezerwacji zapisuje dane osobowe i kontaktowe pacjenta.** 3. **Pracownik rejestracji zapisuje informacje o źródle skierowania.** 4. **Konsultant konsultuje grafik szpitala w celu znalezienia wolnego miejsca na wykonanie żądanej usługi medycznej.** 5. **Pracownik ds. umawiania wizyt sprawdza, czy pacjent jest dostępny na wizytę.** **Pętla 1: Powtarzaj kroki 4-5, aż harmonogram szpitala będzie odpowiadał dostępności pacjenta.** **Pętla 2: Powtórz kroki 4-6 dla każdej wizyty.**



Aktor



LEKARZ, JAKO AKTOR, JEST BYTEM SPOZA SYSTEMU

Prescribe medication

Class



LEKARZ, JAKO KLASA, JEST BYTEM SYSTEMOWYM

Class	Responsibilities	Use Cases
Appointment	• Make appointment for patient. • Know appointment. • Cancel appointment. • Know referral source. • Track appointment.	• 100: Refer Patient • 120: Make Appointment • 130: Receive Patient
Credit Card	• Knows patient credit card. • Verifies patient credit card.	142: Verify Credit Card 190: Resolve Patient Billing Issue
Health Insurance Plan	• Knows patient's health insurance plan. • Verifies patient's health insurance plan.	140: Register Patient 145: Verify Insurance Plan
Hospital ID Card	• Issues hospital card.	130: Receive Patient 140: Register Patient 144: Issue Hospital Card
Medical Service	• Knows medical service.	100: Refer Patient
Patient	• Knows patient's personal data. • Knows patient's contact data. • Knows patient's appointments. • Knows patient's insurance data. • Knows patient's financial data. • Knows patient's referral source. • Knows patient next of kin. • Issues hospital card.	100: Refer Patient 120: Make Appointment ...
Patient Statement (Bill)	Knows patient open (unpaid) billing items. Knows paid items from the last statement. Prints billing statement.	160: Print Patient Statement 190: Resolve Patient Billing Issue
Payment	Knows the payment credited to patient's account.	190: Resolve Patient Billing Issue
Referral Source	Knows the referral source.	120: Make Appointment
Hospital Schedule	Knows appointments. Knows openings for medical services.	120: Make Appointment



Odkrywanie klas

Pełne zdefiniowanie klasy lub obiektu polega na szczegółowym określeniu obowiązków.

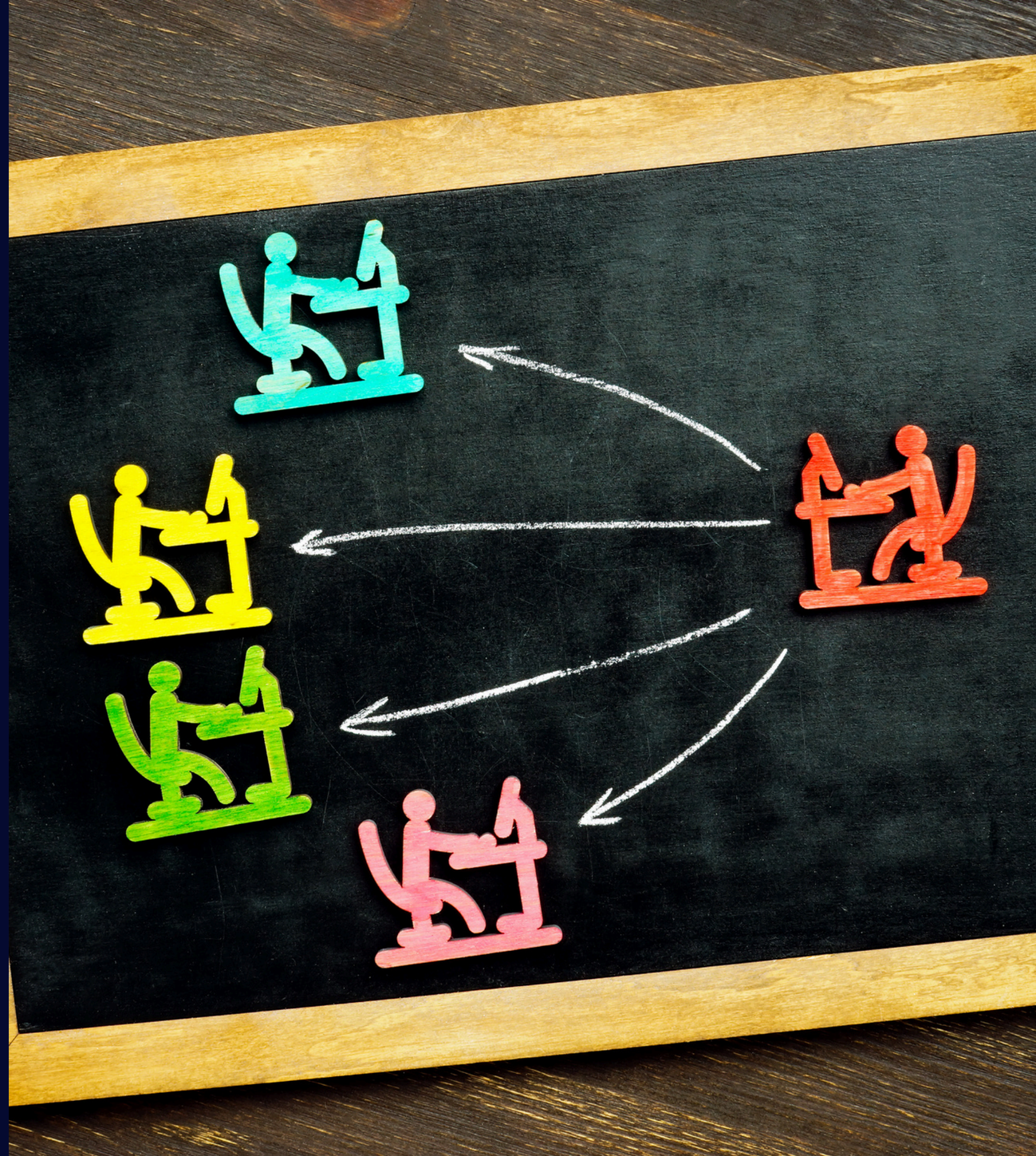
Klasa Pacjent ma szereg obowiązków:

- Zna dane osobowe pacjenta.
- Zna dane kontaktowe pacjenta.
- Zna terminy wizyt pacjenta.
- Zna dane ubezpieczeniowe pacjenta.
- Zna dane finansowe pacjenta.
- Znajomość źródła skierowania pacjenta.
- Znajomość historii medycznej pacjenta.
- Znajomość najbliższej rodziny pacjenta.
- Wystawia kartę szpitalną.



Delegowanie odpowiedzialności

- Obiekt może delegować obowiązki do obiektów, które z nim współpracują.
- "Jeśli typ [atrybutu] ma złożoną strukturę danych, często musimy stworzyć nową klasę typu atrybutu...".

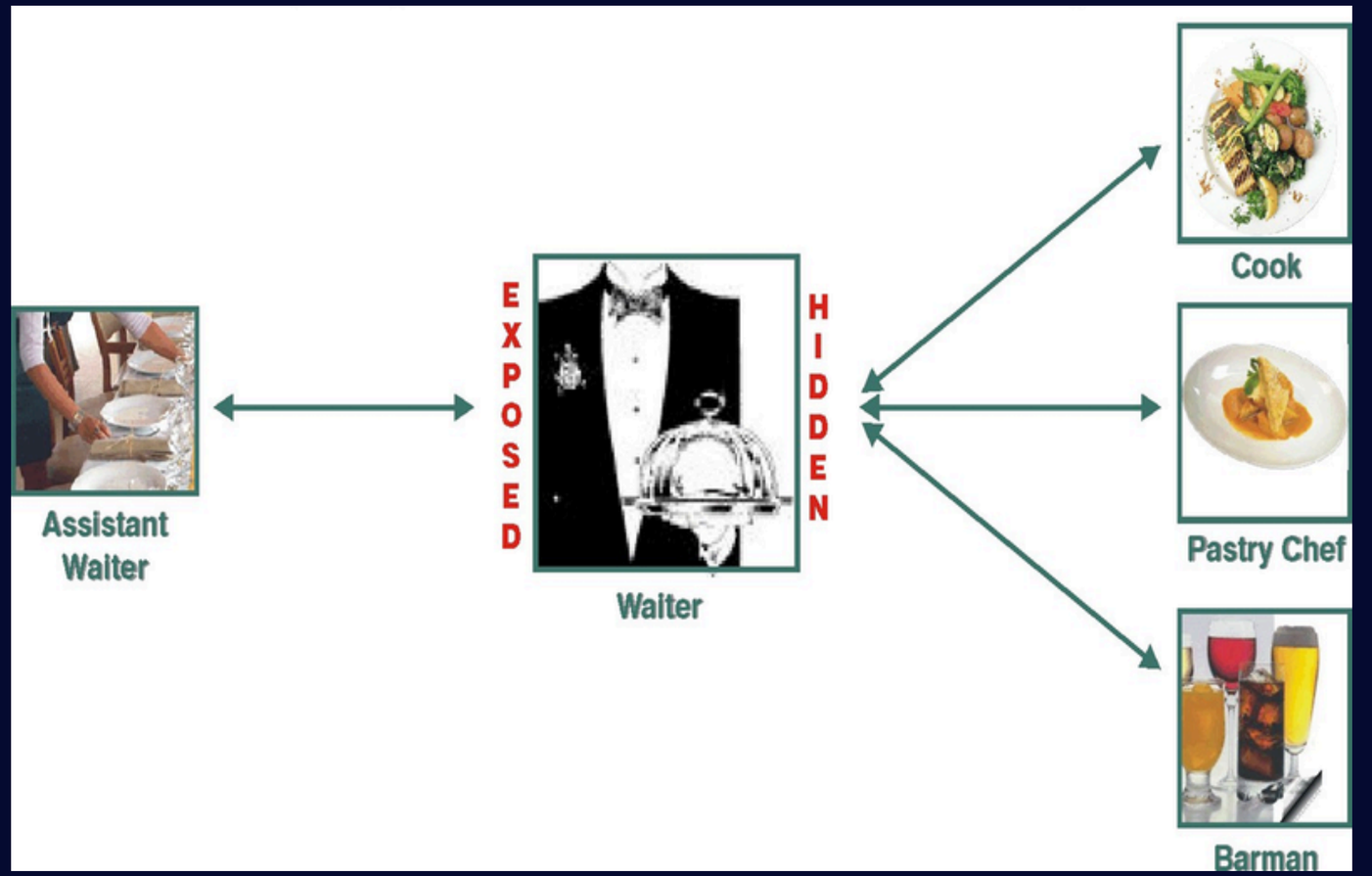


Przykład

Wzorzec Delegacja
Używany do przekazywania wywołań metod z jednej klasy do innej.

Wzorzec Pełnomocnik (Proxy)
Klasa pełnomocnik deleguje zadania do innej klasy, kontrolując dostęp do niej.

Wzorzec Fasada
Zapewnia uproszczony interfejs, delegując zadania do klas wewnętrznych.



Atrybut

patientID
lastName
firstName
midInit
sex
birthDate
age
placeOfBirth
nationality
addresses
phones
insurancePolicy
creditCard
validCardID
medicalHistory

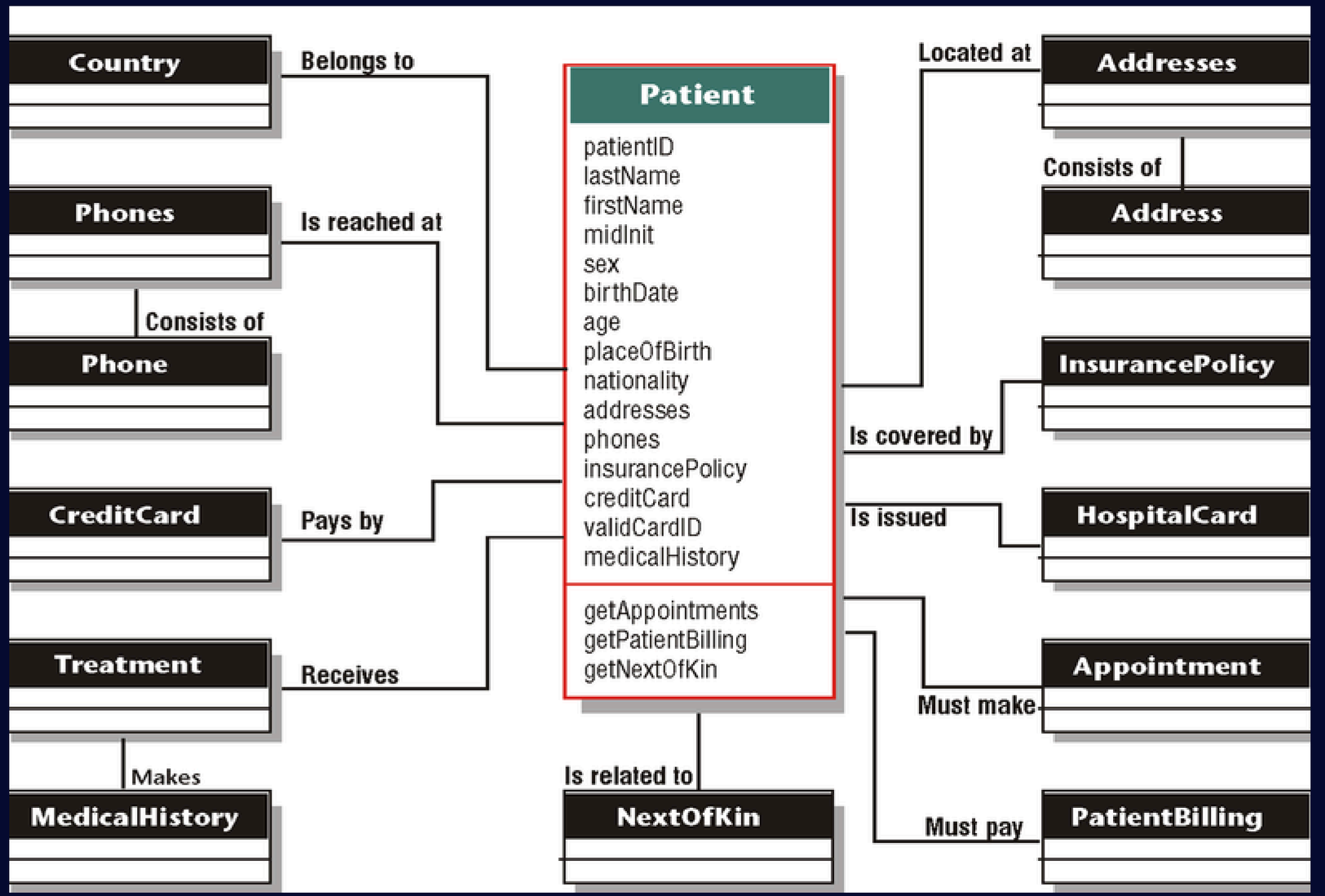
Współpracownicy

Country
Address
Addresses
Phone
Phones
InsurancePolicy
CreditCard
HospitalCard
Treatment
MedicalHistory



Operacje

Operation	Collaborators
getAppointments	Appointment
getPatientBilling	PatientBilling
getNextOfKin	NextOfKin


Country
Belongs to
Patient
patientID
lastName
firstName
midInit
sex
birthDate
age
placeOfBirth
nationality
addresses
phones
insurancePolicy
creditCard
validCardID
medicalHistory
getAppointments
getPatientBilling
getNextOfKin
Located at
Addresses
Consists of
Address
Phones
Is reached at
Consists of
Phone
InsurancePolicy
Is covered by
CreditCard
Pays by
Is issued
HospitalCard
Treatment
Receives
Appointment
Must make
Makes
MedicalHistory
Is related to
NextOfKin
Must pay
PatientBilling

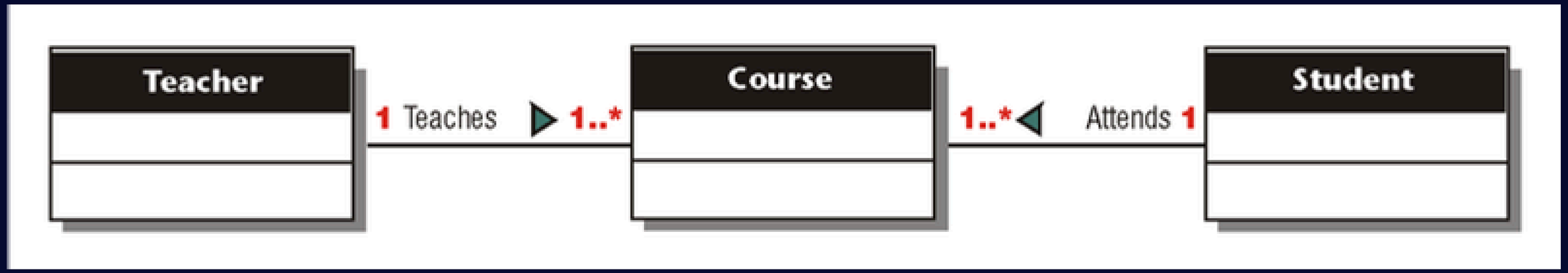
BRAIN BREAK



Wyjdźmy na chwilę z klasy...

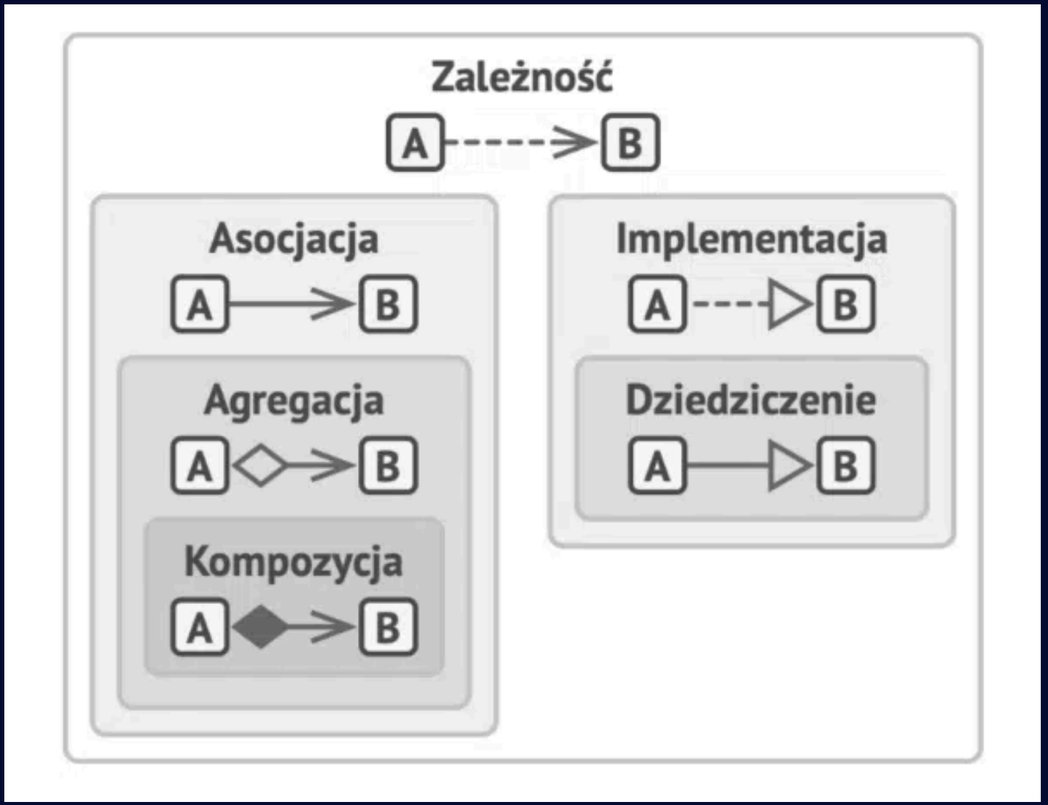
Asocjacja to relacja strukturalna, która określa związek między obiektam i jednej klasy z obiektam i innej klasy.

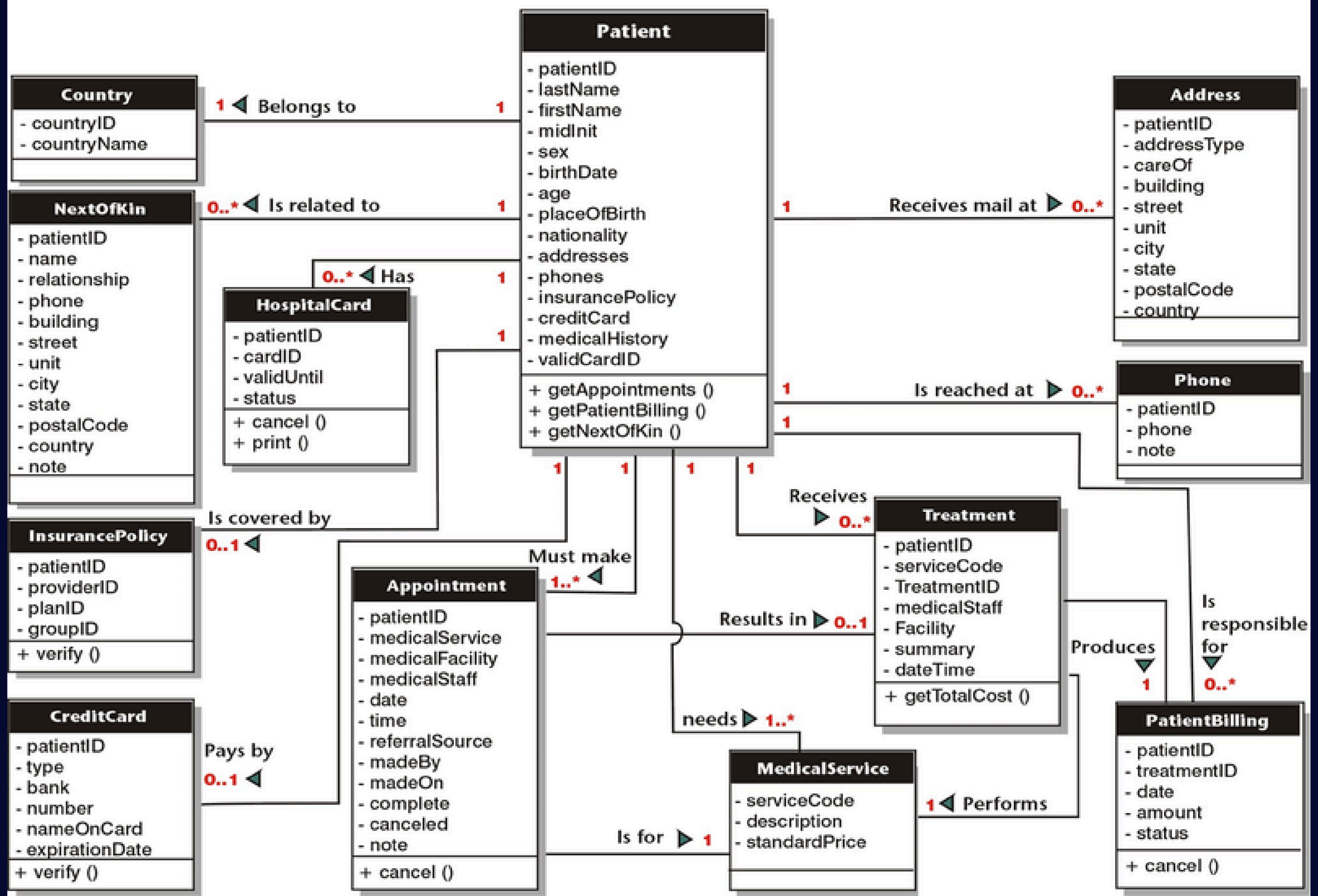


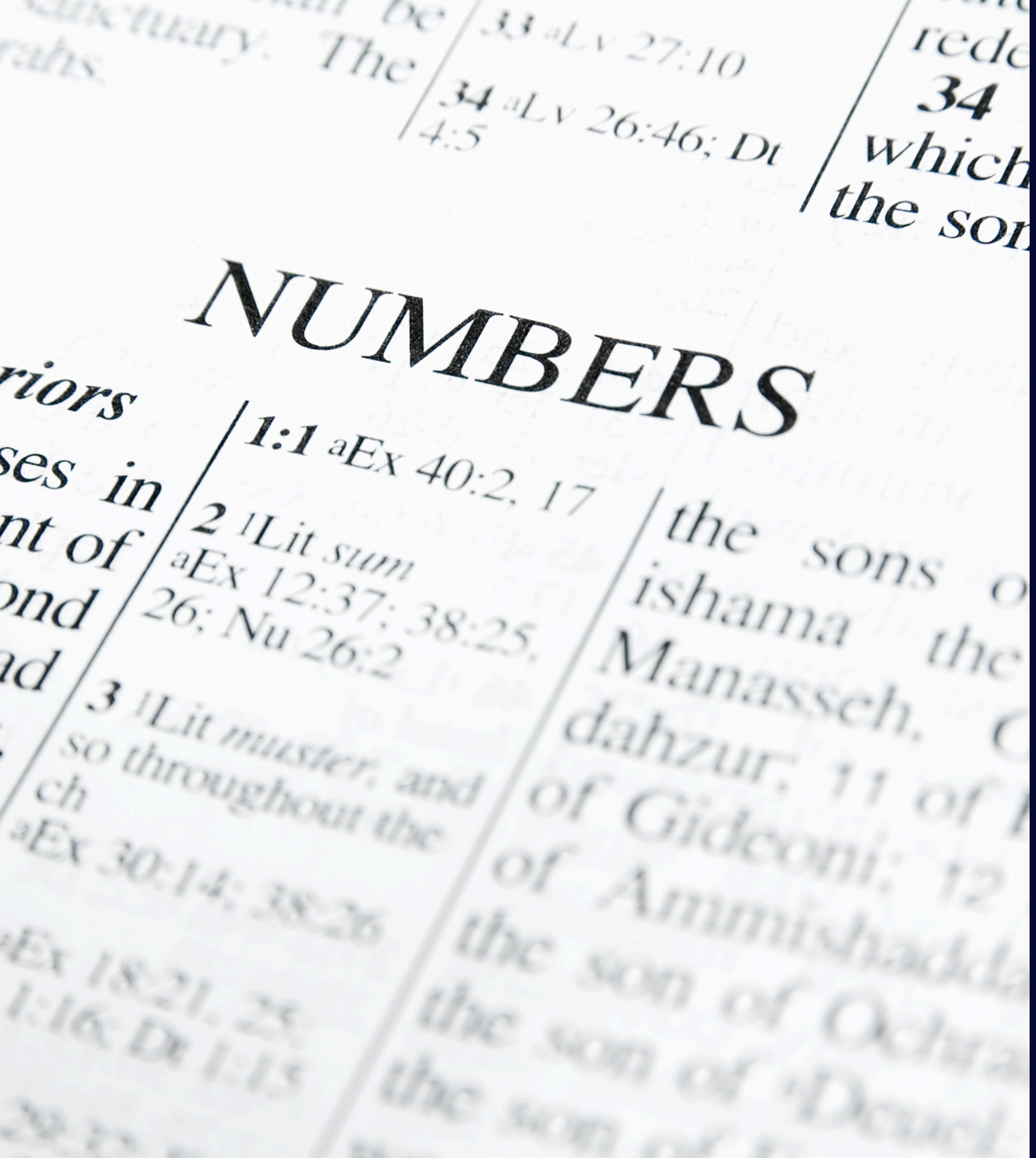


Asocjacja to nie wszystko. W UML określamy inne rodzaje powiązań pomiędzy klasami, takie jak:

- **Kompozycja/Agregacja**
- **Implementacja**
- **Dziedziczenie**



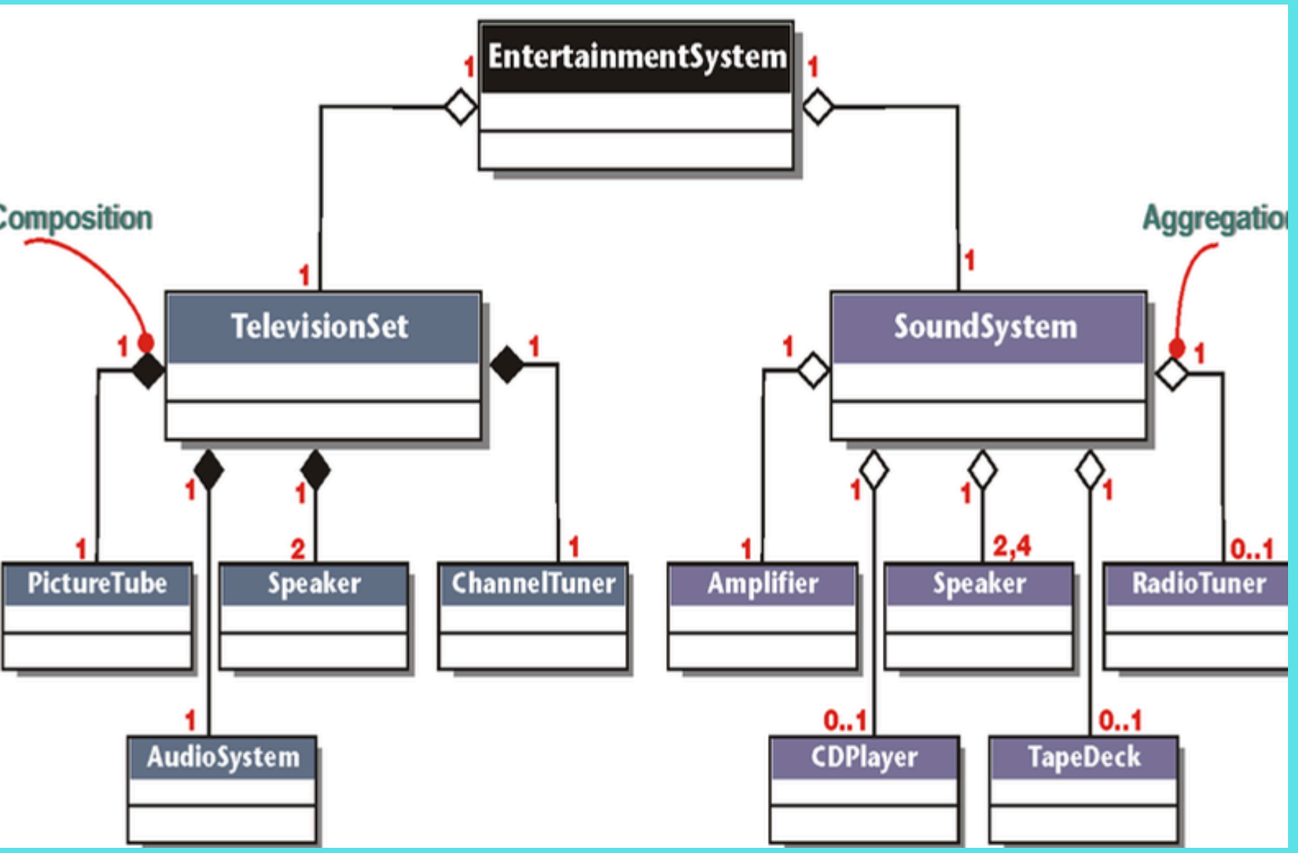

Country
- countryID
- countryName
1 Belongs to 1
Patient
- patientID
- lastName
- firstName
- midInit
- sex
- birthDate
- age
- placeOfBirth
- nationality
- addresses
- phones
- insurancePolicy
- creditCard
- medicalHistory
- validCardID
+ getAppointments ()
+ getPatientBilling ()
+ getNextOfKin ()
Address
- patientID
- addressType
- careOf
- building
- street
- unit
- city
- state
- postalCode
- country
1 Receives mail at 0..*
NextOfKin
- patientID
- name
- relationship
- phone
- building
- street
- unit
- city
- state
- postalCode
- country
- note
0..* Is related to 1
0..* Has 1
HospitalCard
- patientID
- cardID
- validUntil
- status
+ cancel ()
+ print ()
1
1 Is reached at 0..*
Phone
- patientID
- phone
- note
1
1 1 1 1
Receives 0..*
Treatment
- patientID
- serviceCode
- TreatmentID
- medicalStaff
- Facility
- summary
- dateTime
+ getTotalCost ()
InsurancePolicy
- patientID
- providerID
- planID
- groupID
+ verify ()
Is covered by 0..1
Must make 1..*
Appointment
- patientID
- medicalService
- medicalFacility
- medicalStaff
- date
- time
- referralSource
- madeBy
- madeOn
- complete
- canceled
- note
+ cancel ()
Results in 0..1
Produces 1
Is responsible for 0..*
PatientBilling
- patientID
- treatmentID
- date
- amount
- status
+ cancel ()
needs 1..*
MedicalService
- serviceCode
- description
- standardPrice
1 Performs
Is for 1
CreditCard
- patientID
- type
- bank
- number
- nameOnCard
- expirationDate
+ verify ()
Pays by 0..1



Krotność

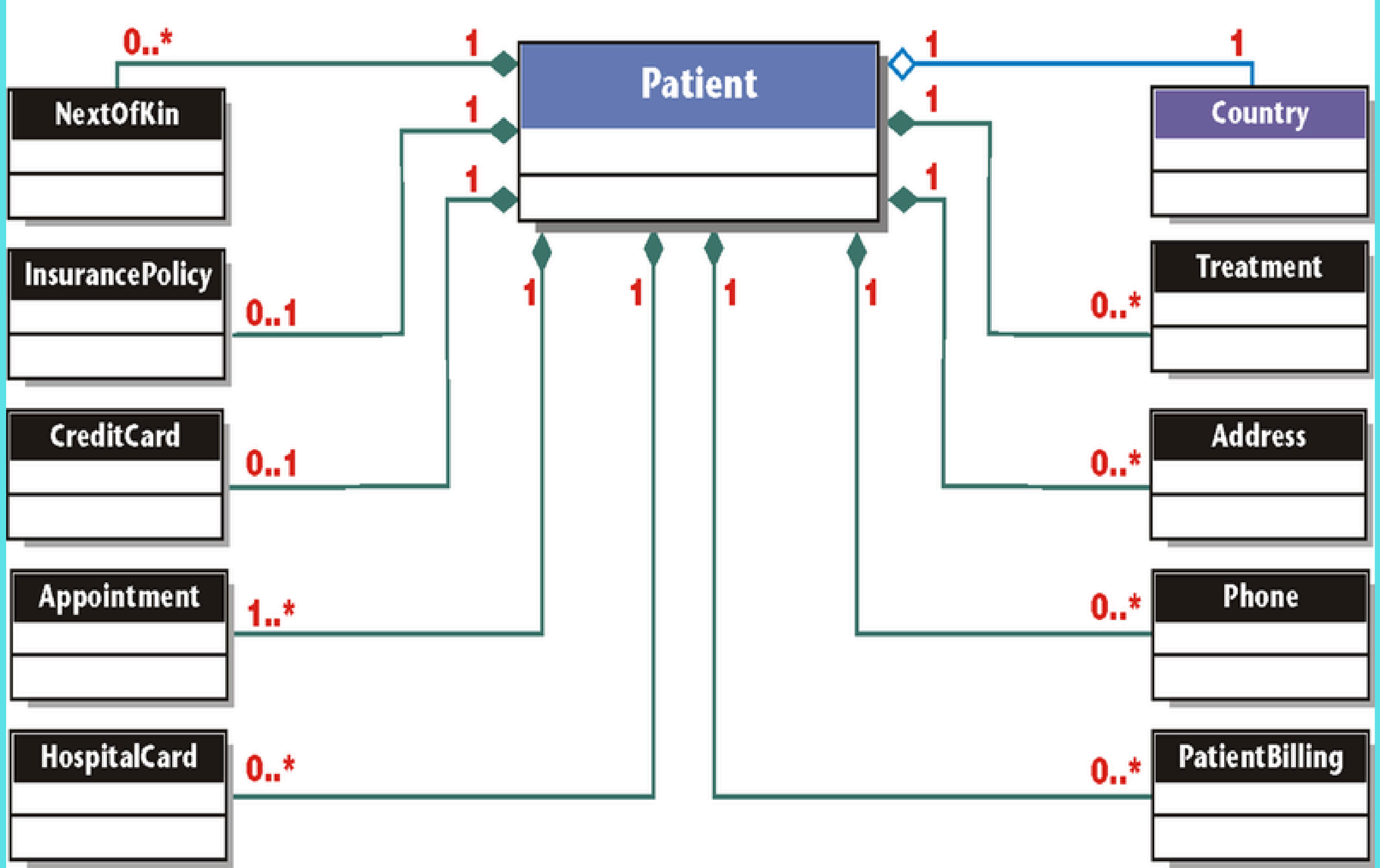
określa, ile instancji jednej klasy może wejść w relację z instancjam i innej klasy.

a..b	od a do b obiektów.
0..1	zero lub jeden obiekt.
*	dowolna liczba obiektów.
1..*	co najmniej jeden obiekt.
n1, n2, ..., nk	konkretne możliwości na ilość obiektów.



Agregacja vs Kompozycja

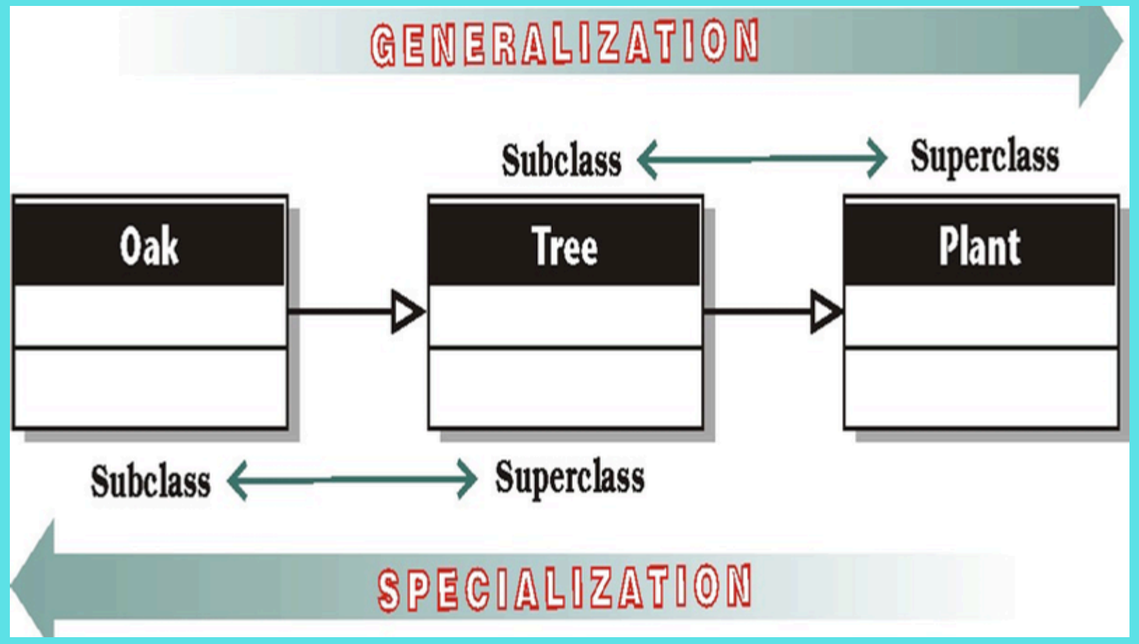
- Agregacja reprezentuje relację całości do części.
- Kompozycja jest formą agregacji, w której część należy wyłącznie do całości, a **jej cykl życia jest zależny od cyklu życia całości.**


Patient
NextOfKin
InsurancePolicy
CreditCard
Appointment
HospitalCard
Country
Treatment
Address
Phone
PatientBilling
0..*
1
0..1
1..*

Generalizacja - Specjalizacja

Podklasy dziedziczą atrybuty i operacje (jest to relacja przechodnia).
Mogą jednak nadpisywać ich zachowanie.









Blood Test

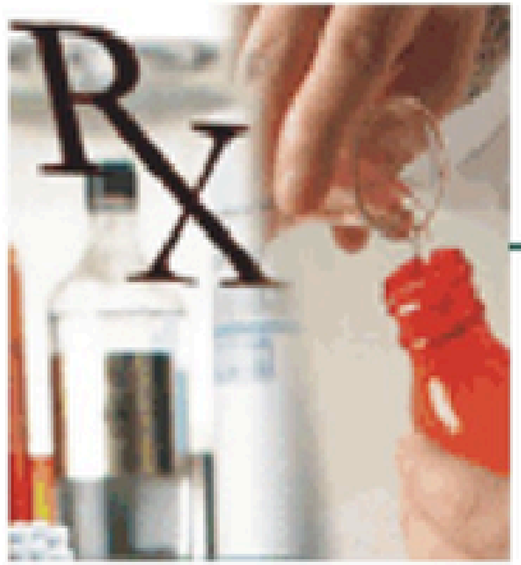
Treatment

- patientID
- serviceCode
- TreatmentID
- medicalStaff
- Facility
- summary
- dateTime

+ getTotalCost ()



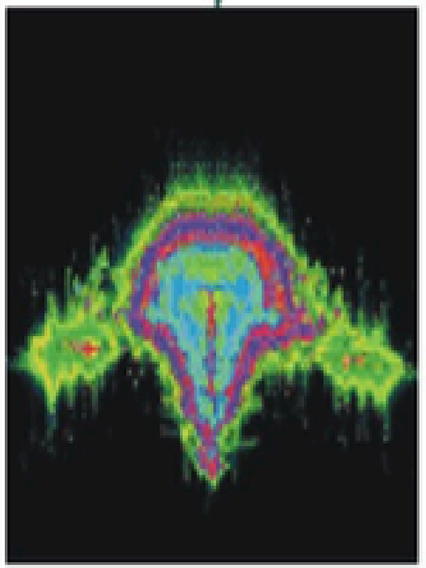
X-Ray



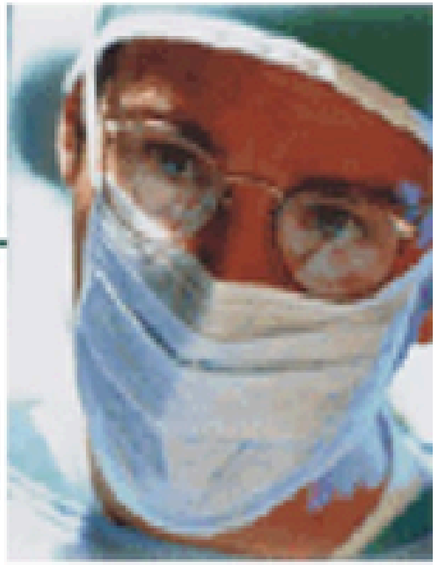
Prescription



Medical Procedure



CAT Scan



Examination

BRAIN BREAK



Pomyśl ile osób ze swojej klasy znasz, a ile nie...
tak samo jest z elementami diagramu klas

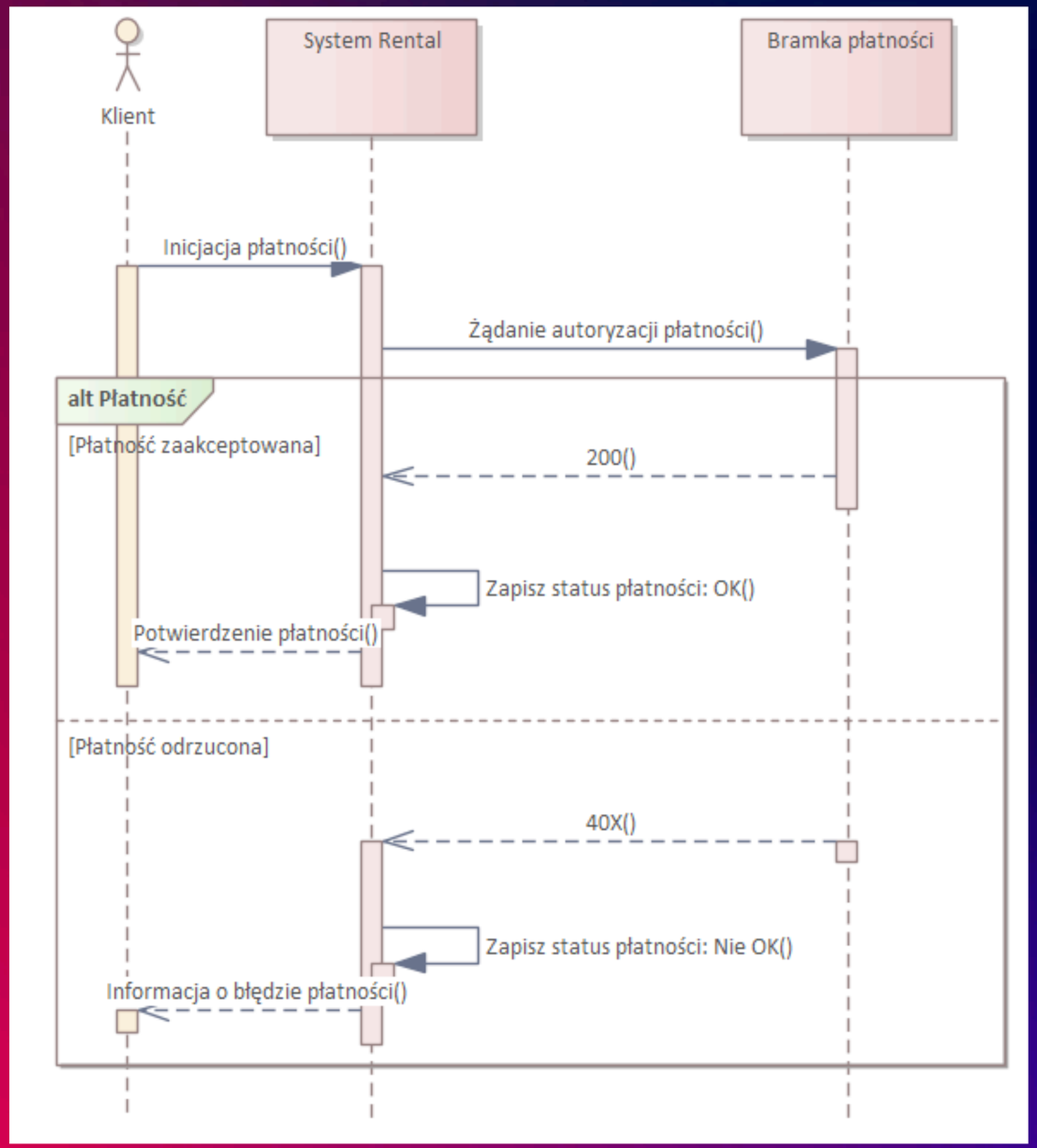


Diagram sekwencji

- Diagram sekwencji jest formą diagramu interakcji, który pokazuje obiekty jako linie życia biegnące w dół strony, z ich interakcjami w czasie reprezentowanymi jako komunikaty narysowane jako strzałki od źródłowej linii życia do docelowej linii życia.
- Diagramy sekwencji są dobre w pokazywaniu, które obiekty komunikują się z innymi obiektami i jakie komunikaty wyzwalają tę komunikację.
- Diagramy sekwencji nie są przeznaczone do pokazywania złożonej logiki proceduralnej.
- Pomijają natomiast całkowicie aspekt dostępu i operacji na danych, związany z komunikacją

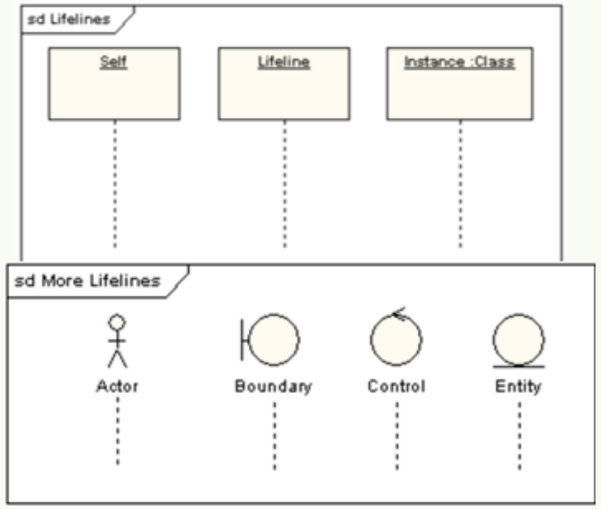
DIAGRAM SEKWENCJI

Obiekty umieszczone na górze diagramu wchodzą w interakcje między sobą poprzez wysyłanie komunikatów oraz komunikatów zwrotnych (poziomo).

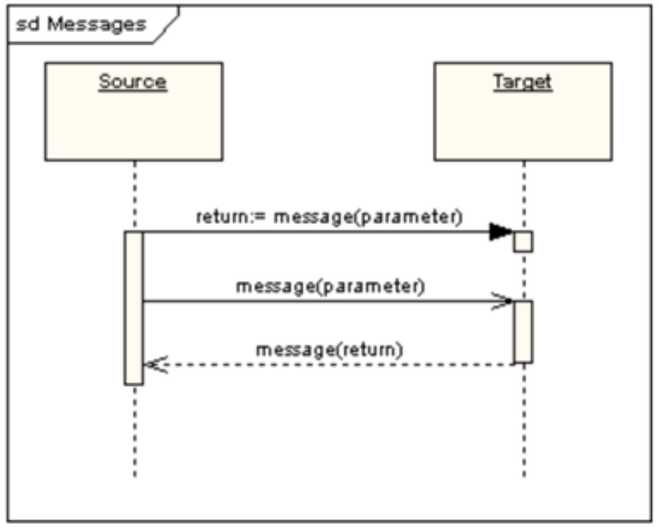
Aktor, jest to użytkownik, którego interakcje z systemem opisane w każdym przypadku użycia(poziomo).

Linia życia prezentuje czas oraz moment początkowy wysyłania komunikatu (pionowo).

Notacja w diagramie przypadków użycia

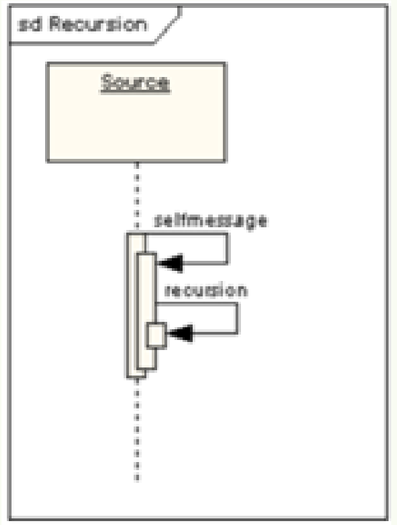


Linia życia reprezentuje indywidualnego uczestnika na diagramie sekwencji. Linia życia ma zwykle prostokąt zawierający nazwę obiektu.
Czasami diagram sekwencji będzie miał linię życia z symbolem elementu aktora na początku. Zazwyczaj dzieje się tak, gdy diagram sekwencji jest własnością przypadku użycia



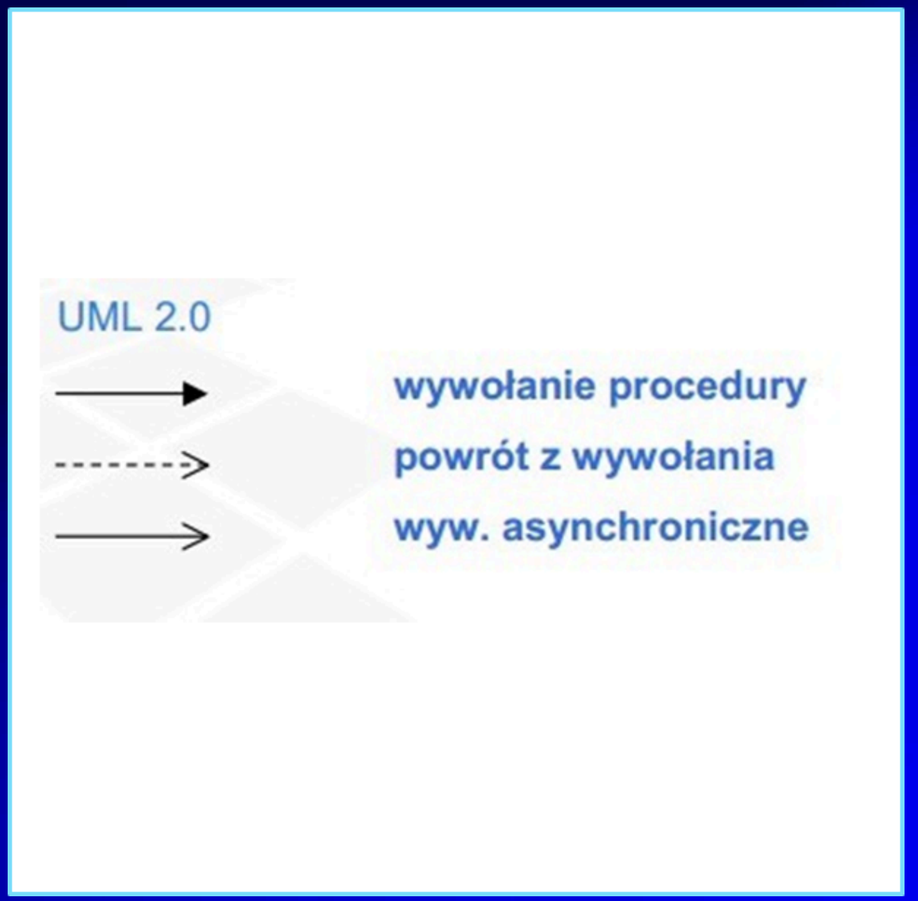
Komunikaty są wyświetlane jako strzałki. synchroniczne lub asynchroniczne; wywołanie lub sygnał.

- Pierwsza wiadomość jest wiadomością synchroniczną (oznaczoną ciągłym grotem strzałki) wraz z domyślną wiadomością zwrotną;
- druga wiadomość jest asynchroniczna (oznaczona grotem strzałki),
- trzecia jest asynchroniczną wiadomością zwrotną (oznaczoną przerywaną linią).



Self-Message - może reprezentować rekurencyjne wywołanie operacji lub jedną metodę wywołującą inną metodę należącą do tego samego obiektu. Jest on wyświetlany jako tworzący zagnieżdżone skupienie kontroli w wystąpieniu wykonania linii życia.

RODZAJE KOMUNIKATÓW



Wywołanie procedury – komunikat synchroniczny oznacza przekazanie sterowania do klasyfikatora odbiorcy. Przepływ sterowania u nadawcy ulega przerwaniu, obiekt jest zajęty oczekiwaniem na zakończenie akcji przez odbiorcę.

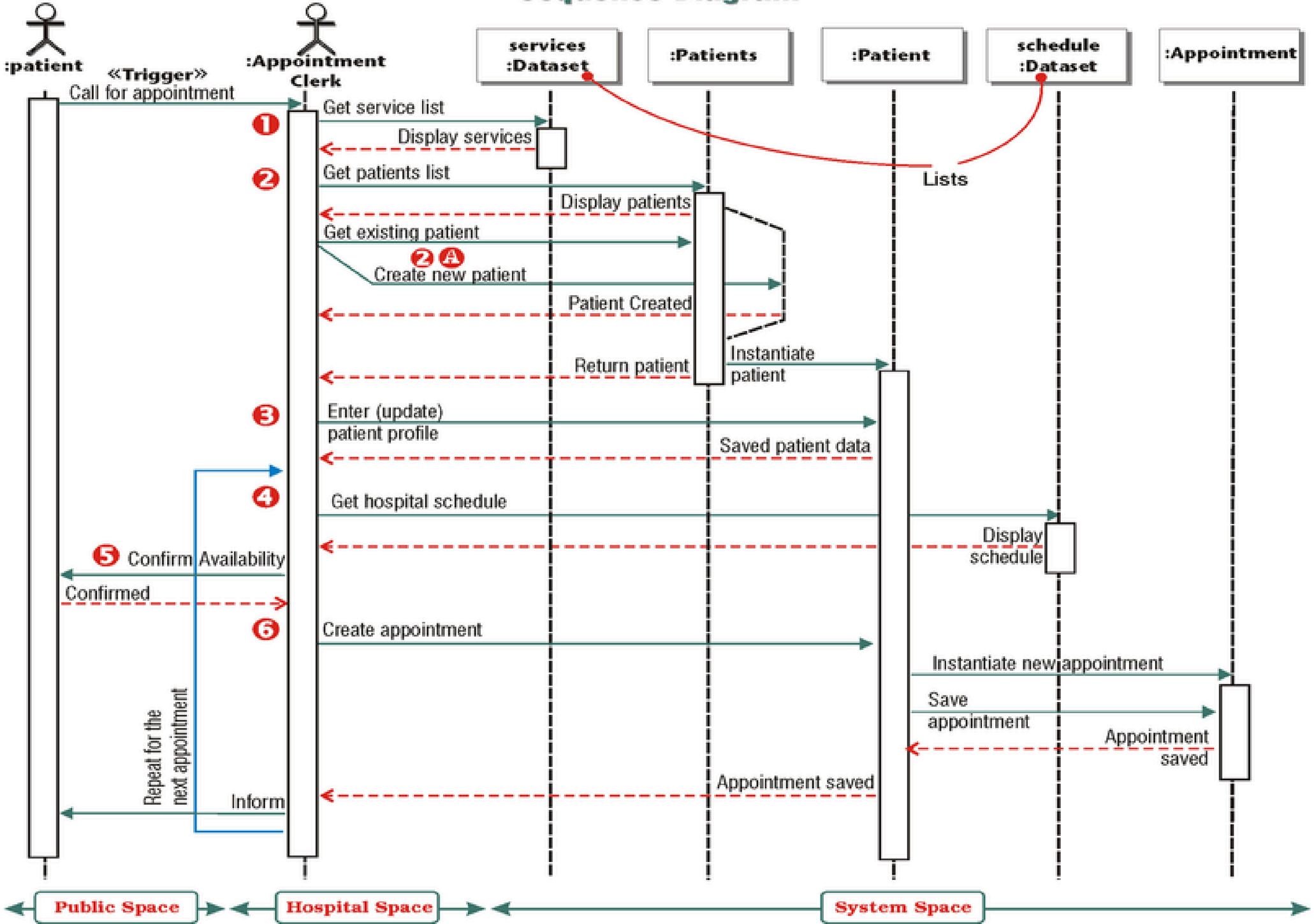
Powrót z wywołania – komunikat zwrotny wskazuje na powrót sterowania do nadawcy po wykonaniu komunikatu synchronicznego

Wywołanie asynchroniczne – nie powodują aktualnego przerwania przepływu sterowania nadawcy. Nadawca nie oczekuje odpowiedzi, więc pozostaje aktywny.

UMAWIANIE WIZYTY

Przeanalizujmy kilka istotnych elementów

Przepływ podstawowy:	1.	Appointment clerk verifies that the needed medical service is provided by the hospital.
	2.	Appointment clerk records patient's personal and contact data.
	3.	Appointment clerk records information about the referral source.
	4.	Appointment clerk consults hospital's schedule to find a free slot for the required medical service.
	5.	Appointment clerk verifies that the patient is available for the appointment.
	➔	Loop 1: **Repeat** steps 4-5 until hospital's schedule matches patient's availability.
	6.	Appointment clerk makes the appointment.
	➔	Loop 2: **Repeat** steps 4-6 for each appointment.
Przepływ alternatywny/ wyjątki:	**2.a**	Patient is not on file. Create new patient. (Extend: 141 - Create Patient.)


:patient
:Appointment Clerk
services :Dataset
:Patients
:Patient
schedule :Dataset
:Appointment
«Trigger»
Call for appointment
1
Get service list
Display services
2
Get patients list
Display patients
Get existing patient
2A
Create new patient
Patient Created
Return patient
Instantiate patient
Lists
3
Enter (update) patient profile
Saved patient data
4
Get hospital schedule
Display schedule
5
Confirm Availability
Confirmed
6
Create appointment
Instantiate new appointment
Save appointment
Appointment saved
Appointment saved
Repeat for the next appointment
Inform
Public Space
Hospital Space
System Space

WSKAZÓWKI

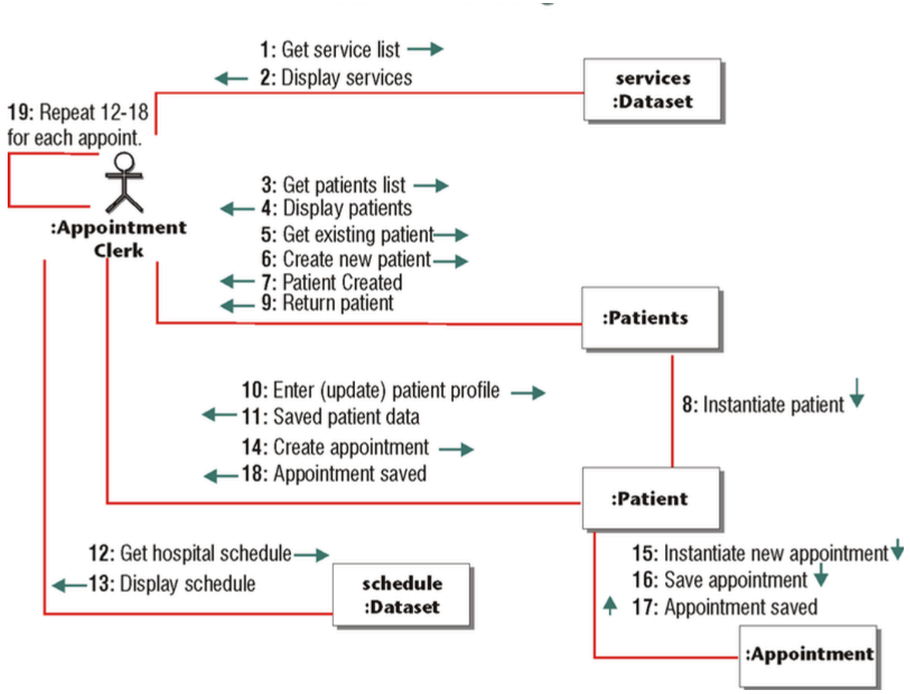
1. Narysuj diagram sekwencji dla każdego przypadku użycia.
2. Wklej tekst scenariusza PU do notatek.
3. Zmapuj tekst scenariusza do treści komunikatów realizowanych przez diagram sekwencji.
4. Opisz strzałki komunikatów oraz je ponumeruj.
5. Zawrzyj przepływy alternatywne oraz pętle.

Diagram komunikacji

Współpraca = komunikacja.

Diagram komunikacji koncentruje się na organizacji obiektów, które muszą ze sobą współpracować, poprzez wymianę komunikatów w celu realizacji określonego zachowania systemu.

Najlepszym sposobem narysowania diagramu kolaboracji jest wyjście od diagramu sekwencji.



BRAIN BREAK



To nie koniec...

ŹRÓDŁA

- https://sparxsystems.com/resources/tutorials/uml2/
- https://wolski.pro/diagramy-uml/
- https://www.guru99.com/pl/uml-notation-symbol.html
- https://www.omg.org/spec/UML/2.5.1/PDF