

Instrukcja

Wprowadzenie do tworzenia

oprogramowania

Relacja 1 do 1..0– instrukcja z lab1

Cele ćwiczenia

Należy:

- wybrać projekt z podanej listy dostępnej za pomocą linku podanego w w laboratorium 1
- sformułować wymagania funkcjonalne i нефunkcjonalne dla wybranego projektu jako zadanie domowe. Zadanie domowe będzie stanowić podstawę do zaprojektowania przypadków użycia na kolejnych laboratorium.
- wykonać projekt UML i wykonać prosty program stanowiący realizację projektu zgodnie z materiałem zawartym na slajdach 5-84. Jest to ćwiczenie, które pozwala poznać narzędzie Visual Paradigm 10 for UML 2.0 wykorzystane w ramach zajęć laboratoryjnych z przedmiotu Podstawy Inżynierii Oprogramowania.

Java

język programowania

- obiektowo zorientowany
- wysokiego poziomu

platforma Javy

- z maszyny wirtualnej VM
- API (interfejs programowania aplikacji).

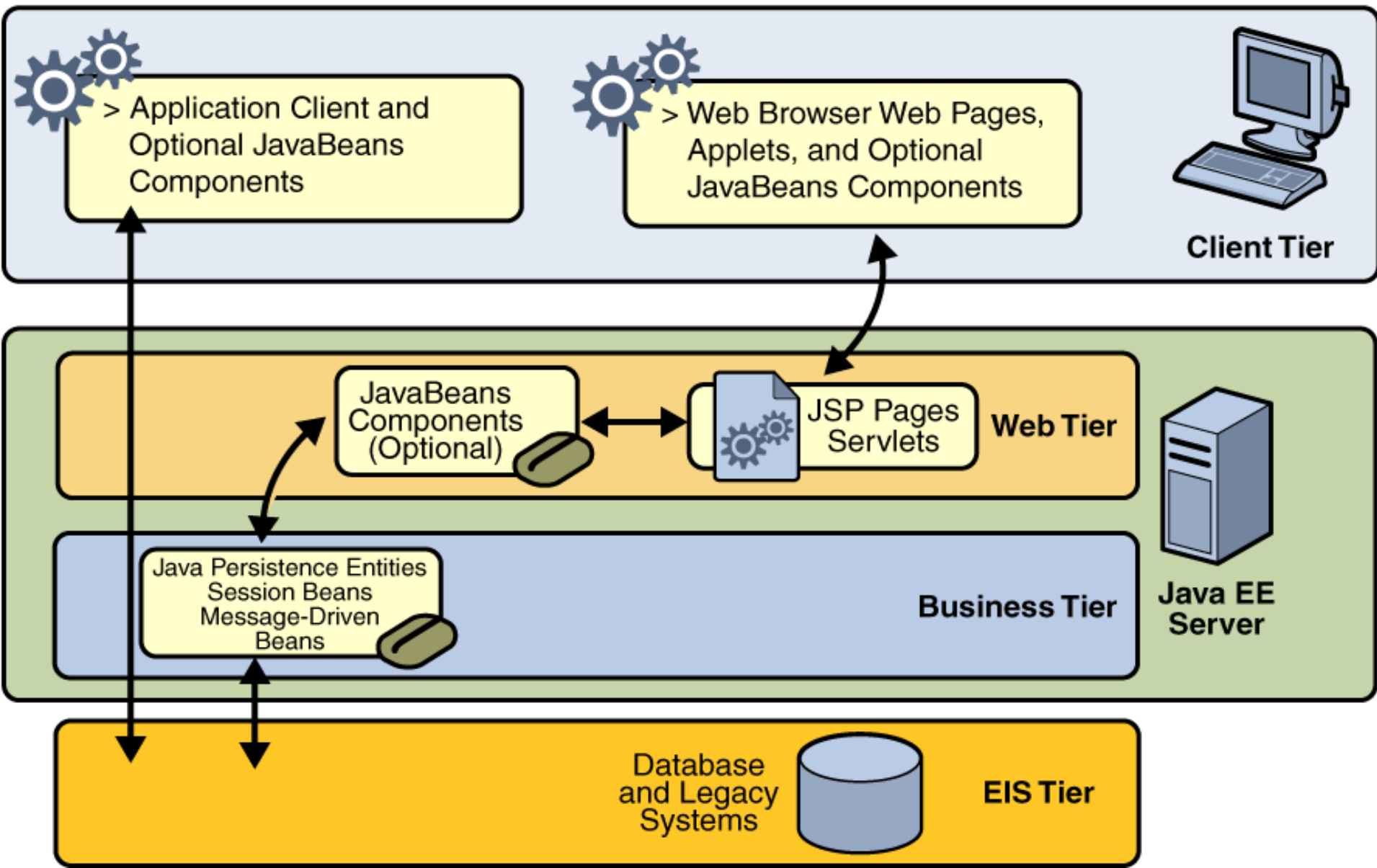
Rezultat

- niezależność od platformy,
- duże możliwości,
- stabilność,
- łatwość rozwoju,
- bezpieczeństwo

Rodzaje platform Javy:

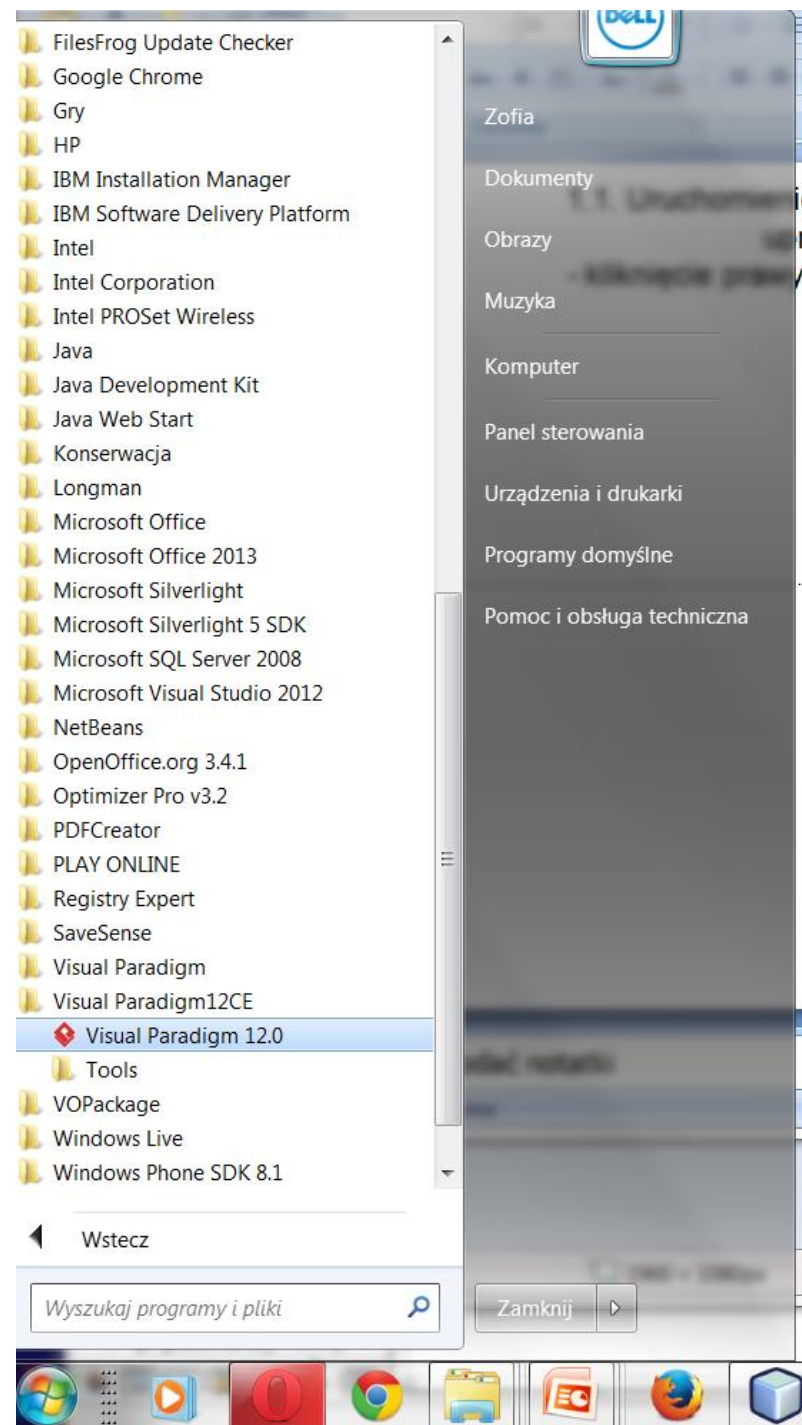
- ◆ Java Platform, Standard Edition (Java SE)
- ◆ Java Platform, Enterprise Edition (Java EE)
- ◆ Java Platform, Micro Edition (Java ME)
- ◆ Java Platform CARD

1. Warstwy aplikacji (Java EE)

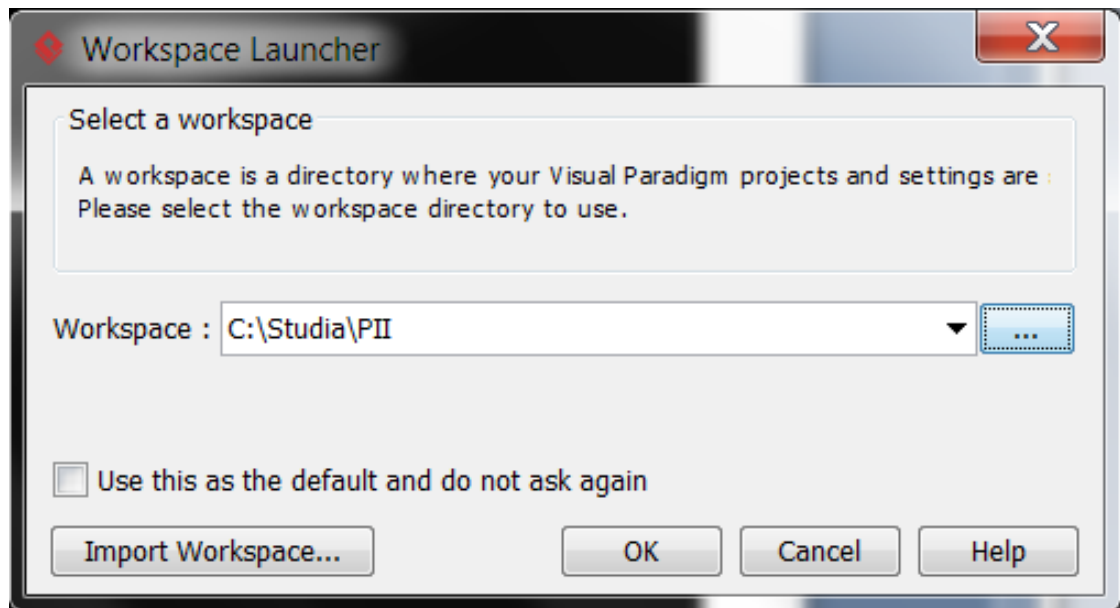


2. Uruchomienie programu programu *Visual Paradigm* 12.0 z uprawnieniami administratora

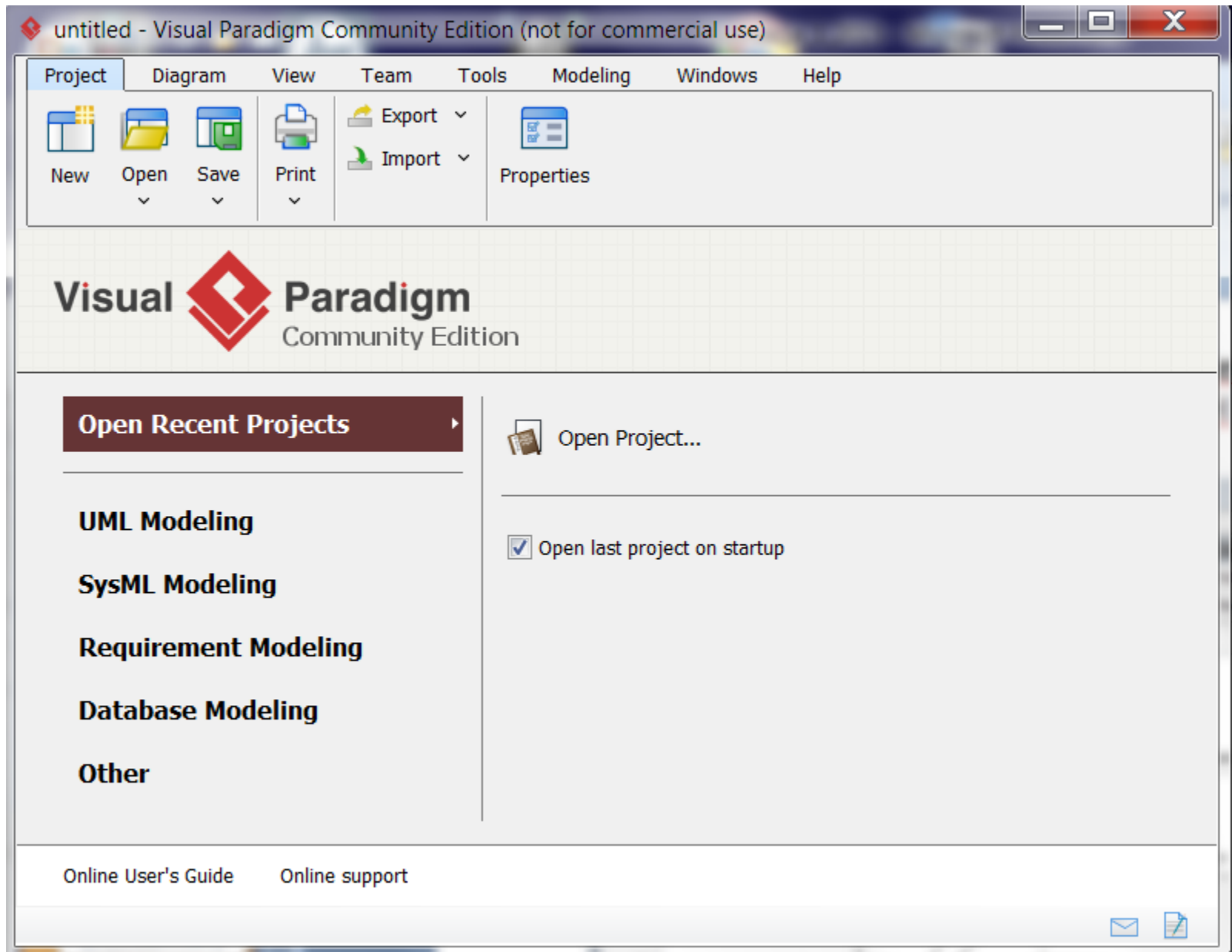
- kliknięcie prawym klawiszem myszy na nazwę programu



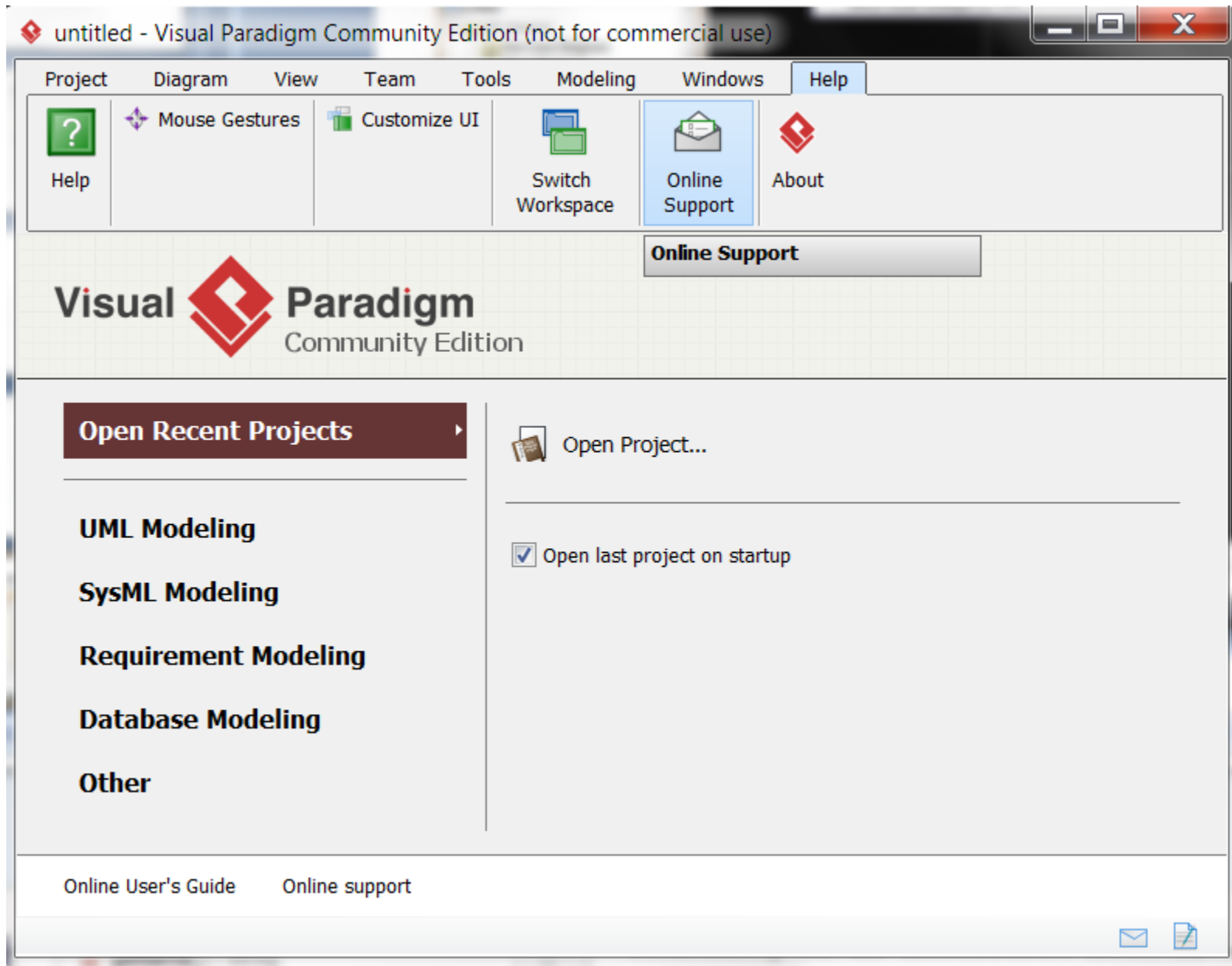
3. Wybór katalogu do tworzenia projektów



4. Start programu bez projektów w wybranej lokalizacji (workspace)



5./5.1 Korzystanie z *Help* ze strony producenta narzędzia



5.2. Przykładowe informacje dostarczone z części *Help*

The screenshot shows a web browser window with the URL <http://www.visual-paradigm.com/support/>. The browser's address bar and menu bar are visible. The page content includes a navigation menu with links for WHAT'S NEW, FEATURES, LEARNING, SUPPORT, BUY, and a prominent DOWNLOAD button. A search bar is located in the top right corner. The main heading asks "What can we help you with?" and suggests finding solutions through various channels. Three channels are highlighted: "VPE Training" (Free online lectures), "User's Guide" (Assist you in using Visual Paradigm), and "YouTube" (Stay-tuned with us). At the bottom, there is a section for submitting a ticket, with a dropdown menu currently set to "Technical Support". The footer contains social media sharing options (Share, Tweet, +1, Like) and a language selection dropdown.

http://www.visual-paradigm.com/support/

Plik Edycja Widok Ulubione Narzędzia Pomoc

Strona Bezpieczeństwo Narzędzia

WHAT'S NEW FEATURES LEARNING SUPPORT BUY **DOWNLOAD** Search

What can we help you with?
You may find a solution from the following channels.

VPE Training
Free online lectures

User's Guide
Assist you in using
Visual Paradigm

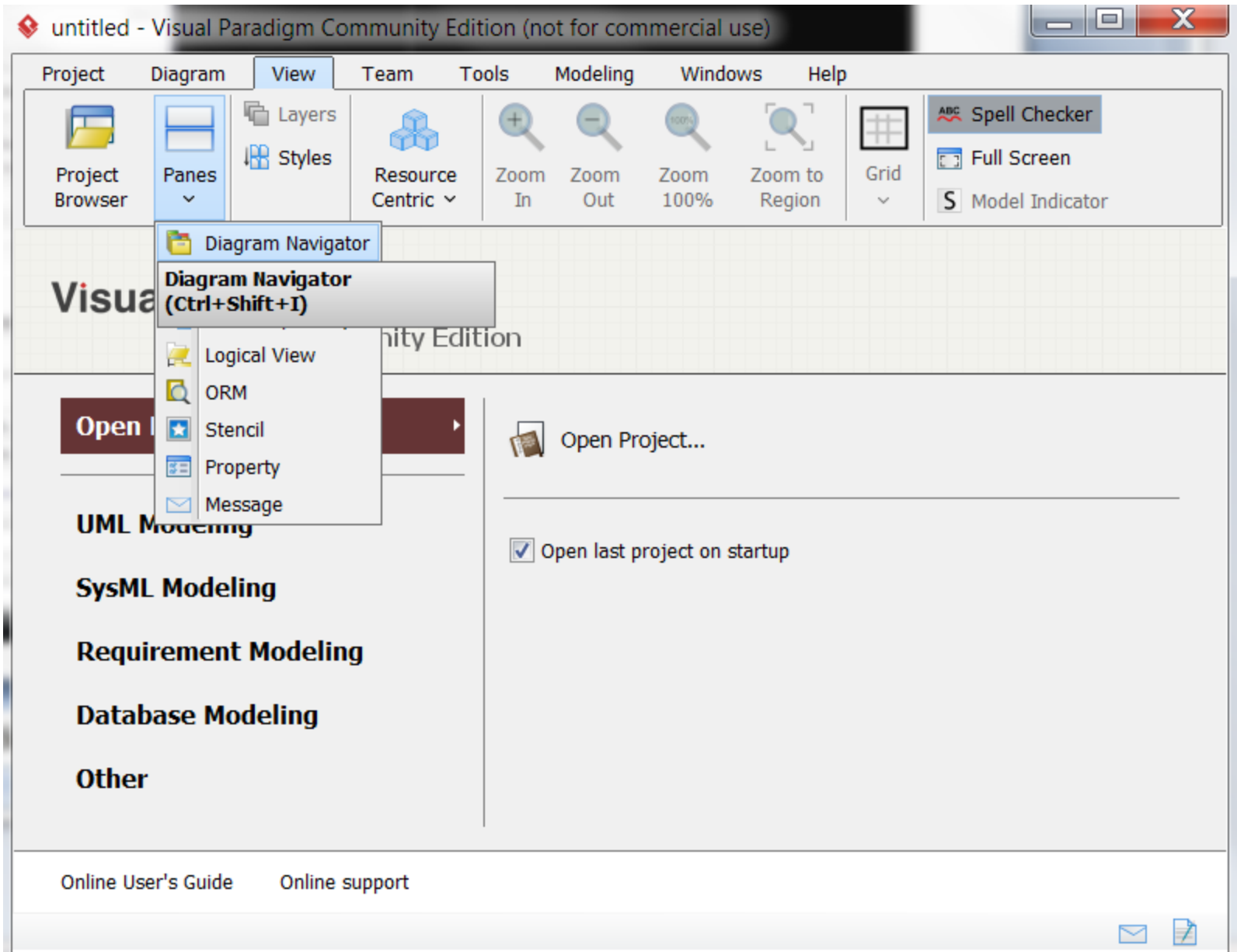
YouTube
Stay-tuned with us

If you can't find a solution to your problem, please submit a ticket below.

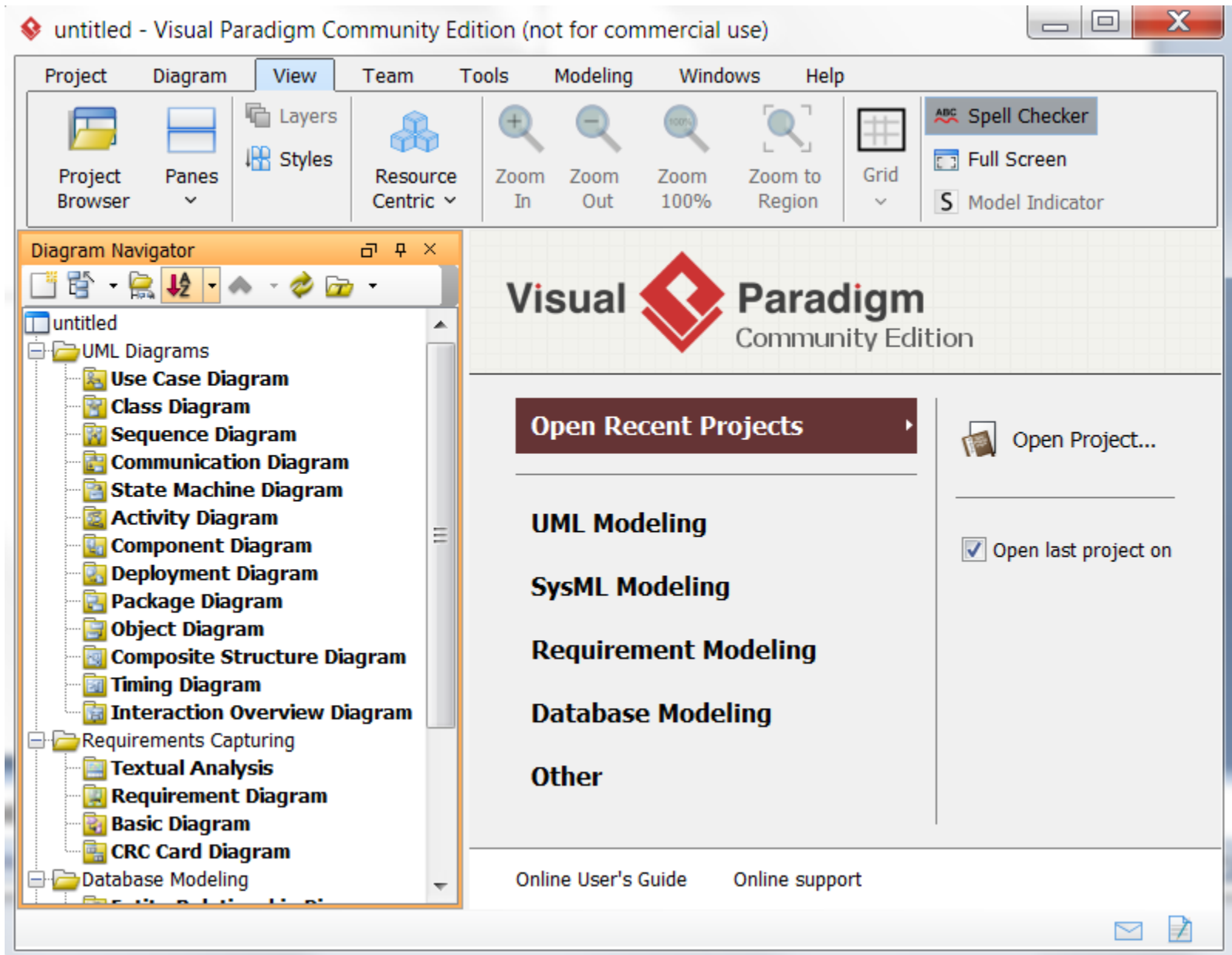
To: Technical Support

Share Tweet +1 6 Like 2 Wybierz język Visual Paradigm 12.0 Build 20150302 ChangeLog

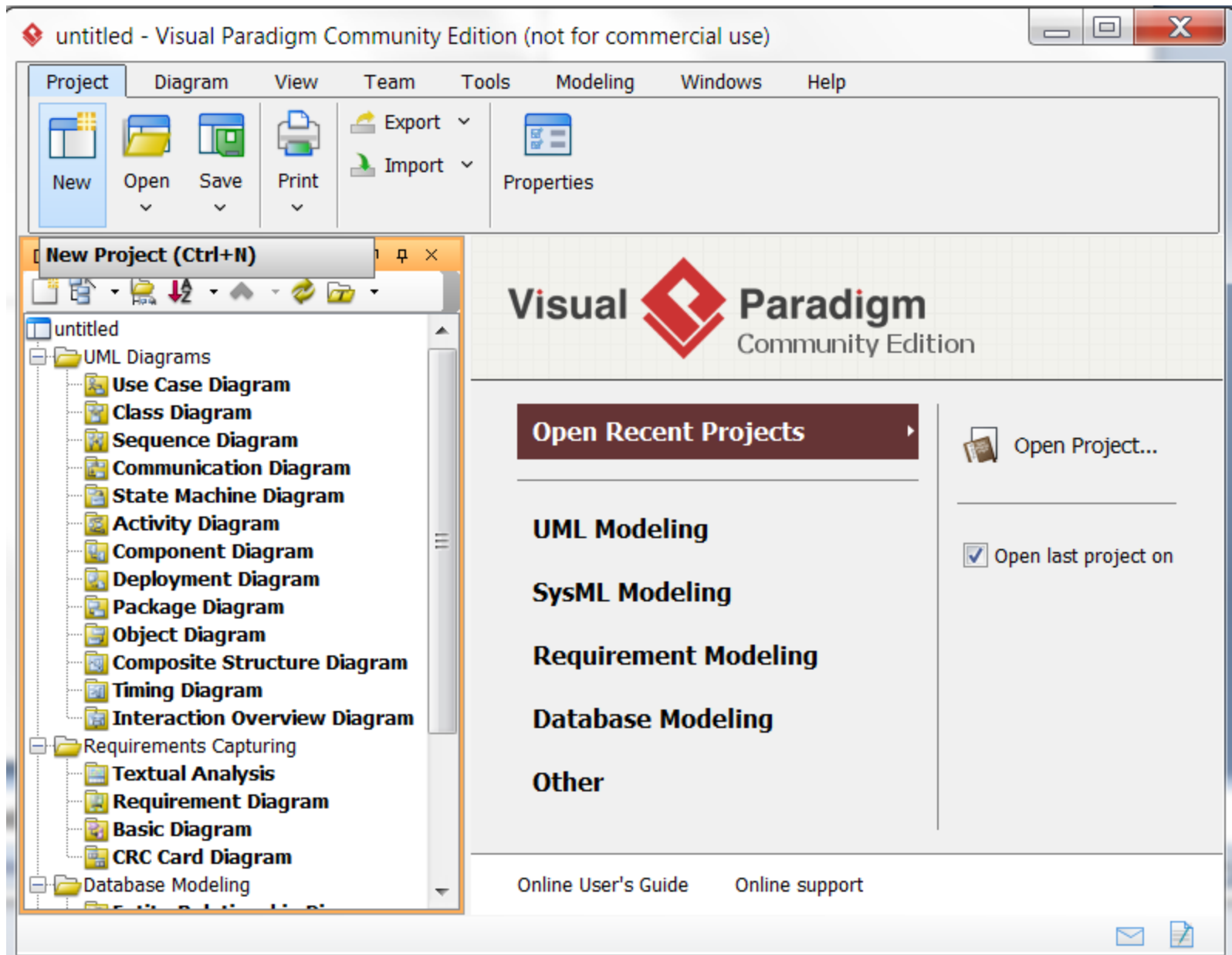
5.3. Wybór sposobu wyświetlania projektu - View/Panes/Diagram Navigator lub Ctrl+Shift+I



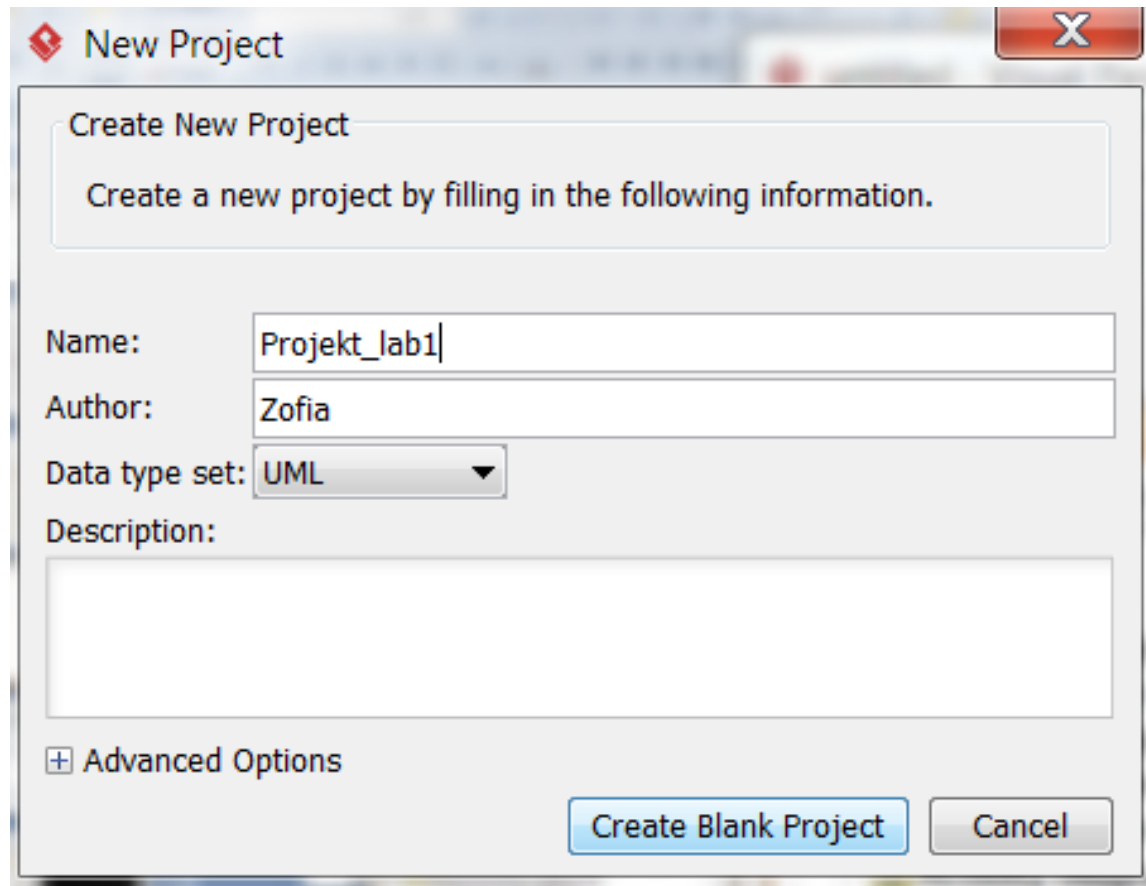
5.4. Wybór sposobu wyświetlania projektu - View/Panes/Diagram Navigator lub Ctrl+Shift+I - rezultat



6/6.1. Utworzenie nowego projektu – *File/New Project*



6.2. Utworzenie nowego projektu – nadanie nazwy nowemu projektowi w formularzu *New Project* w polu *Project name*



New Project

Create New Project

Create a new project by filling in the following information.

Name: Projekt_lab1

Author: Zofia

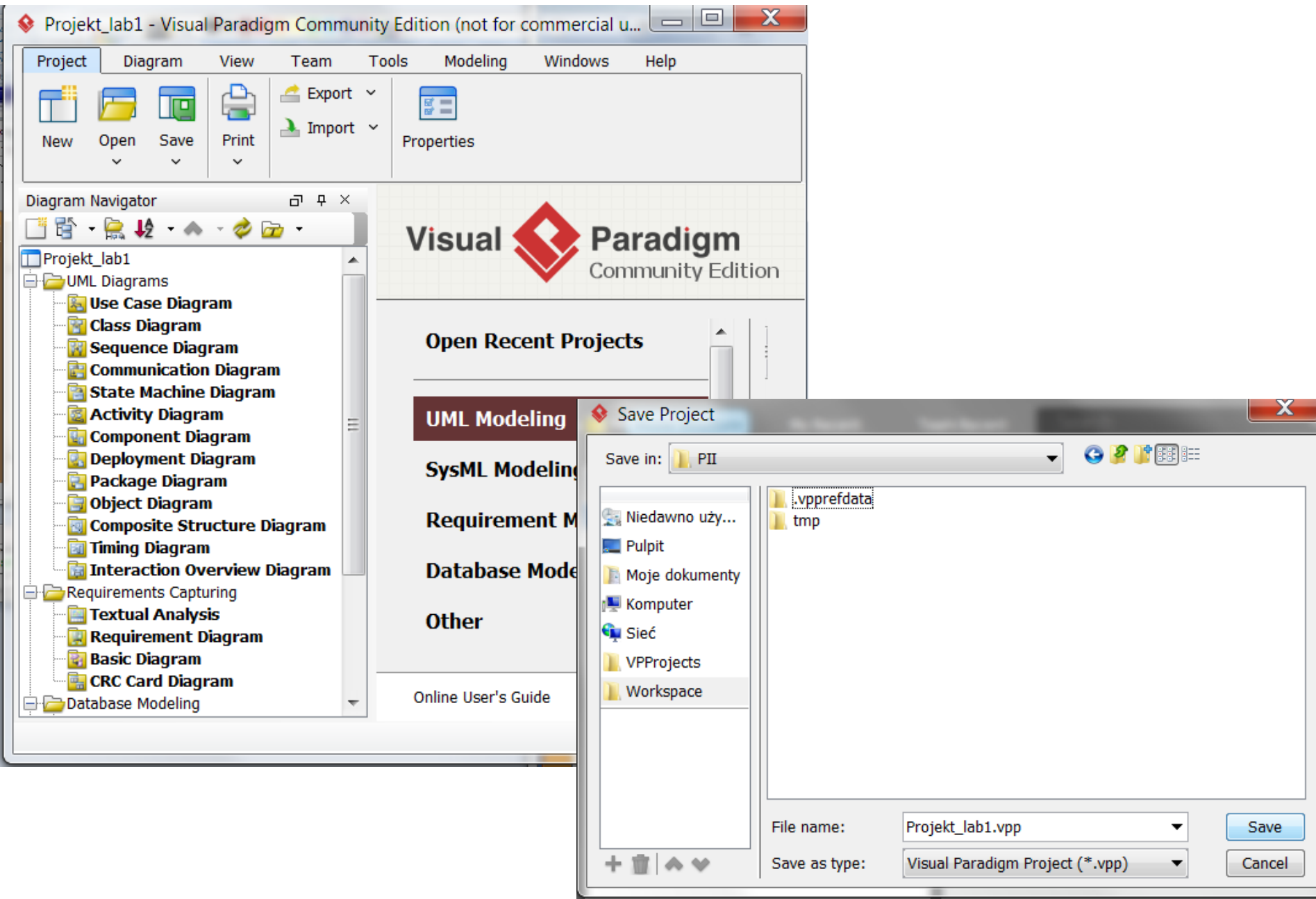
Data type set: UML

Description:

+ Advanced Options

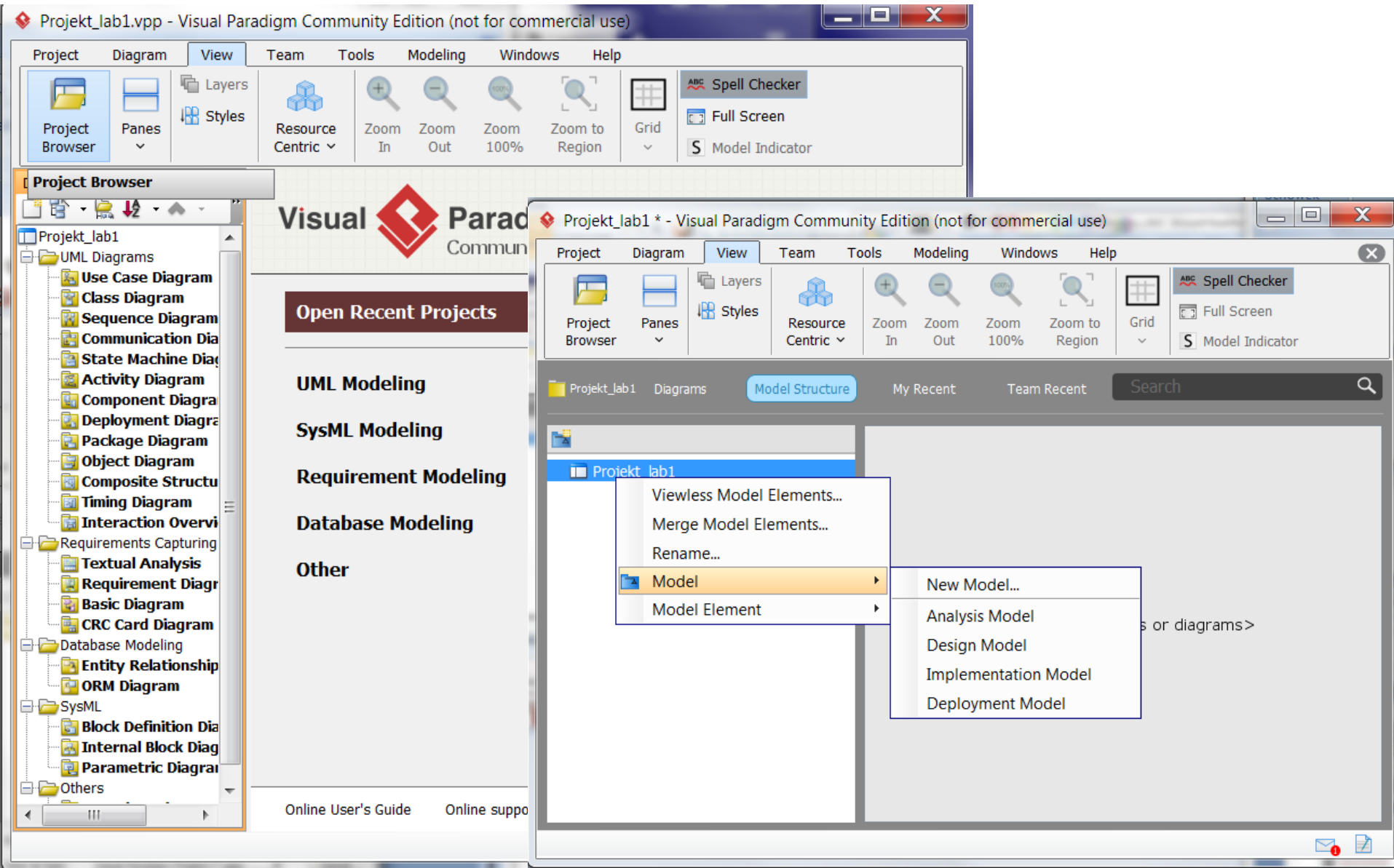
Create Blank Project Cancel

6.3. Widok nowego, pustego projektu – w stylu *Diagram Navigator*

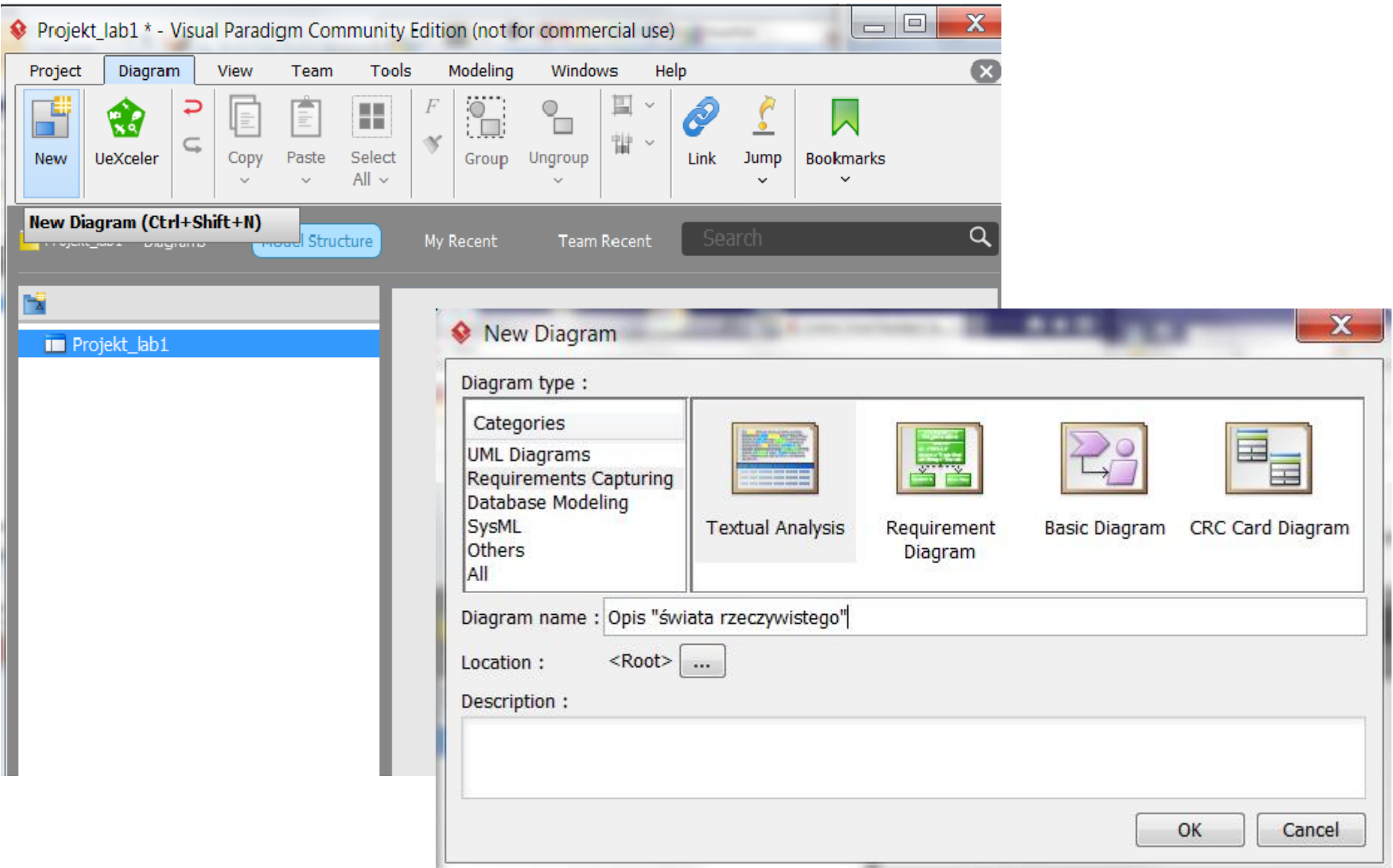


6.4. Tworzenie modelu analizy w projekcie (w widoku *View/Project Browser*)

– po kliknięciu prawym klawiszem na nazwę projektu należy wybrać z list:
Model/Analysis Model



7/7.1. Dodanie diagramu reprezentującego opis „świata rzeczywistego”
– po kliknięciu prawym klawiszem myszy na nazwę modelu wybór z listy: *Diagram/New/Requirements Capturing/Textual Analysis*



7.2. Należy wykonać opis biznesowy „świata rzeczywistego” – Katalog tytułów i książek w bibliotece

1. **Opis zasobów ludzkich**

Pracownik wypożyczalni może dodawać do katalogu tytułów nowe tytuły. Każdy tytuł jest reprezentowany przez następujące dane: tytuł, autor, wydawnictwo, ISBN oraz informacje o liczbie egzemplarzy i miejscu ich przechowywania i występuje w bibliotece jako pojedyncza informacja dla każdego tytułu. Pewna grupa tytułów opisuje książki nagrane na kasety, dlatego dodatkowo tytuł zawiera dane nagrania np nazwisko aktora. Każdy egzemplarz, niezależnie, czy jest książką czy kasetą, jest opisany odrębną informacją zawierającą numer egzemplarza i ewentualnie (dotyczy to wyodrębnionych egzemplarzy) informację o liczbie dni, na które można wypożyczyć egzemplarz. Numery egzemplarzy mogą się powtarzać dla różnych tytułów. Pracownik biblioteki (bibliotekarz) może dodawać nowe tytuły i egzemplarze oraz je przeszukiwać, natomiast klient może jedynie przeszukiwać tytuły i sprawdzać egzemplarze wybranych tytułów.

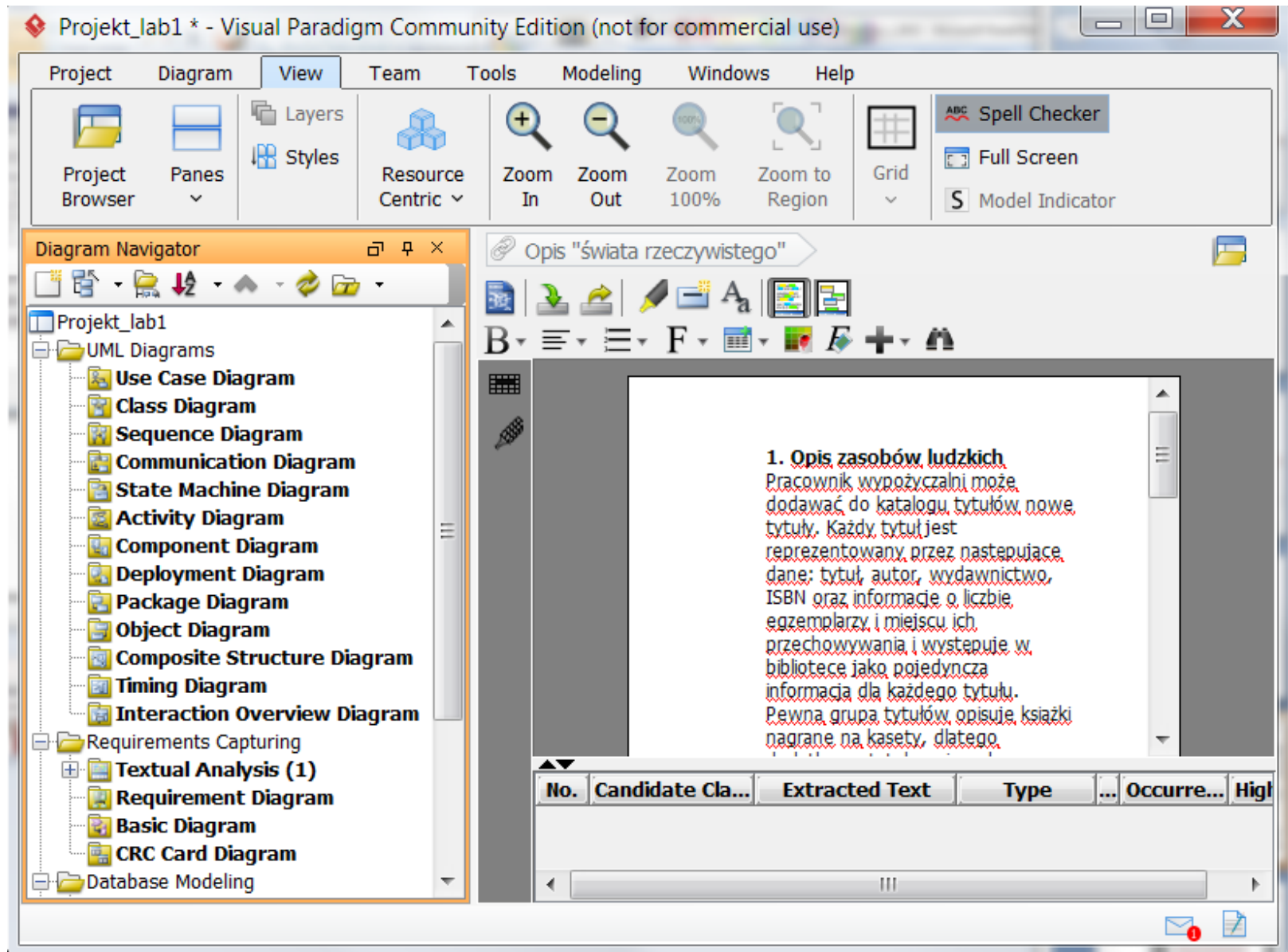
2. **Przepisy**

Pracownik ponosi odpowiedzialność za poprawność danych - odpowiada materialnie za niezgodność danych ze stanem wypożyczalni.

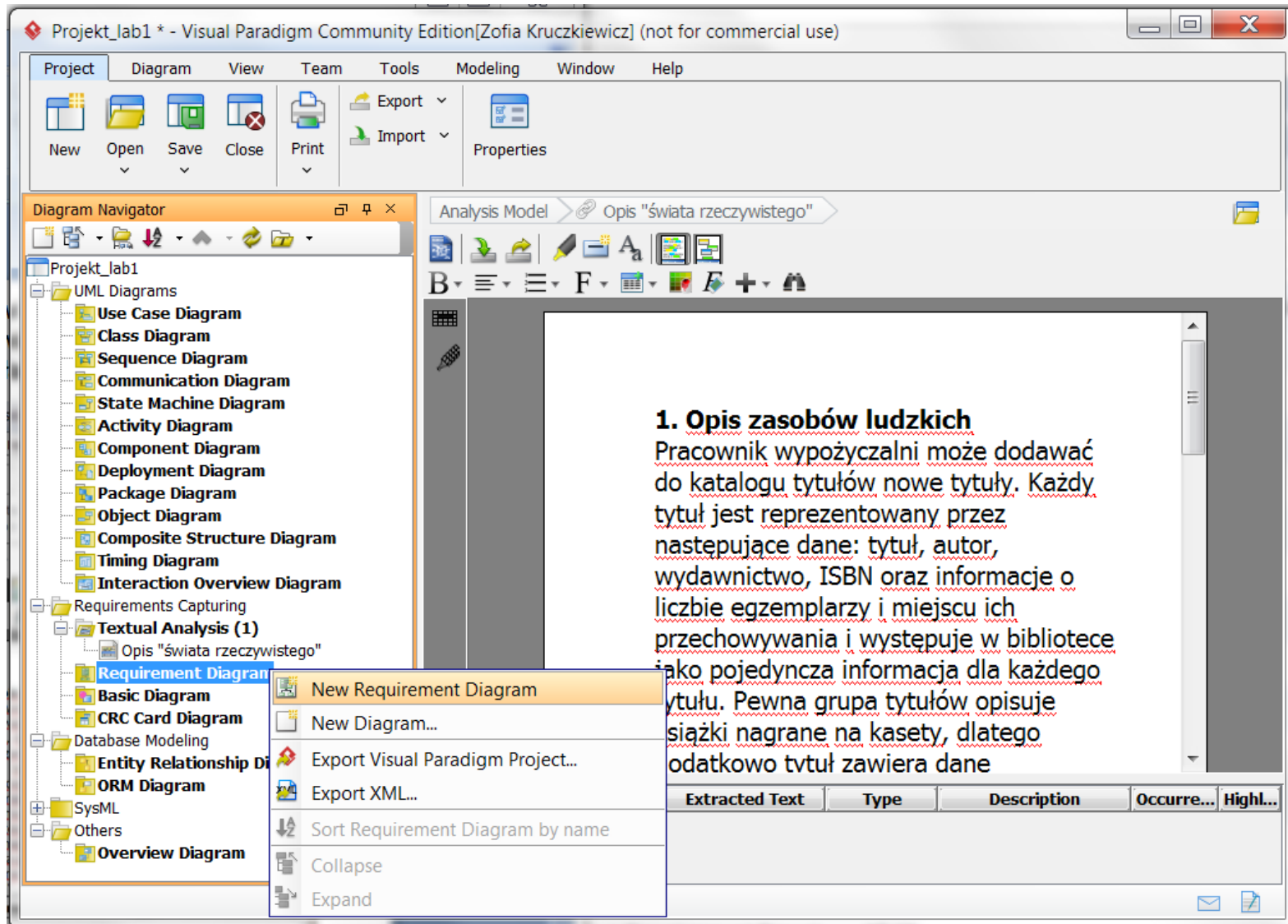
3. **Dane techniczne**

Klient może przeglądać dane wypożyczalni za pośrednictwem strony internetowej lub bezpośrednio za pomocą specjalnego programu. Zakłada się, że klientów jednocześnie przeglądających dane wypożyczalni może być ponad 1000 oraz wypożyczalnia może zawierać kilkadziesiąt tysięcy tytułów oraz przynajmniej dwukrotnie więcej egzemplarzy. Biblioteka składa się z kilku ośrodków w różnych miastach na terenie kraju (lista miast jest dołączona do umowy). Zaleca się stosowanie technologii Java.

7.3. Okno diagramu – należy nadać nazwę diagramowi np. Opis „świata rzeczywistego” i należy wprowadzić tekst reprezentujący opis „świata rzeczywistego” projektu (następny slajd zawiera przykład opisu)



8/8.1. Należy wyspecyfikować wymagania funkcjonalne i niefunkcjonalne za pomocą diagramu wymagań (*Requirement Diagram*). W tym celu należy dodać ten diagram klikając prawym klawiszem myszy na pozycję *Requirement Diagram/ New Requirement Diagram*.



8.2. Należy zdefiniować wymagania aplikacji

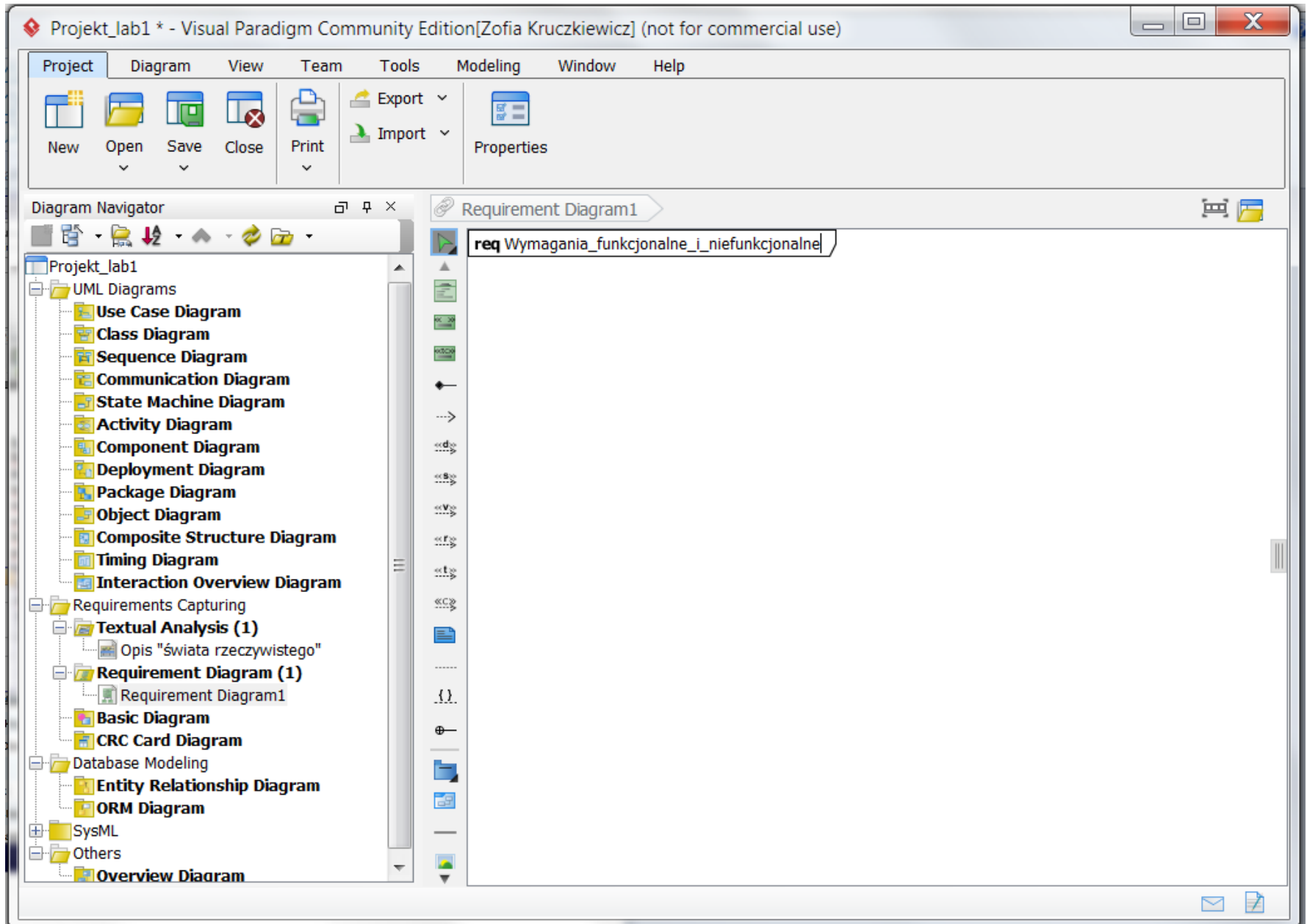
Wymagania funkcjonalne

- Biblioteka wypożycza podane książki i czasopisma osobom zarejestrowanym, o ile je posiada
- Biblioteka dokonuje zakupu nowych książek, przy czym popularne książki kupuje w kilku egzemplarzach. Usuwa zniszczone książki i czasopisma.
- Bibliotekarz jest pracownikiem biblioteki, komunikuje się z wypożyczającym. Jego praca jest wspierana za pomocą systemu
- Wypożyczający może zarezerwować książkę lub czasopismo, które nie jest dostępne w danej chwili, W momencie, kiedy zamówione rzeczy są dostępne- albo po zwrocie lub dzięki zakupowi, można je wypożyczyć i usunąć rezerwację. Rezerwację można usunąć niezależnie.
- Biblioteka może łatwo utworzyć, zmienić i usunąć informację o tytułach, wypożyczających, wypożyczeniach i rezerwacjach

Wymagania нефunkcjonalne

- System powinien pracować w popularnych systemach (UNIX, Windows, OS/2) i powinien mieć nowoczesny graficzny interfejs użytkownika
- System powinien się rozwijać np. wprowadzenie możliwości zawiadamiania rezerwującego książkę o jej dostępności

8.3. Należy wpisać nazwę diagramu wymagań w lewym górnym rogu diagramu

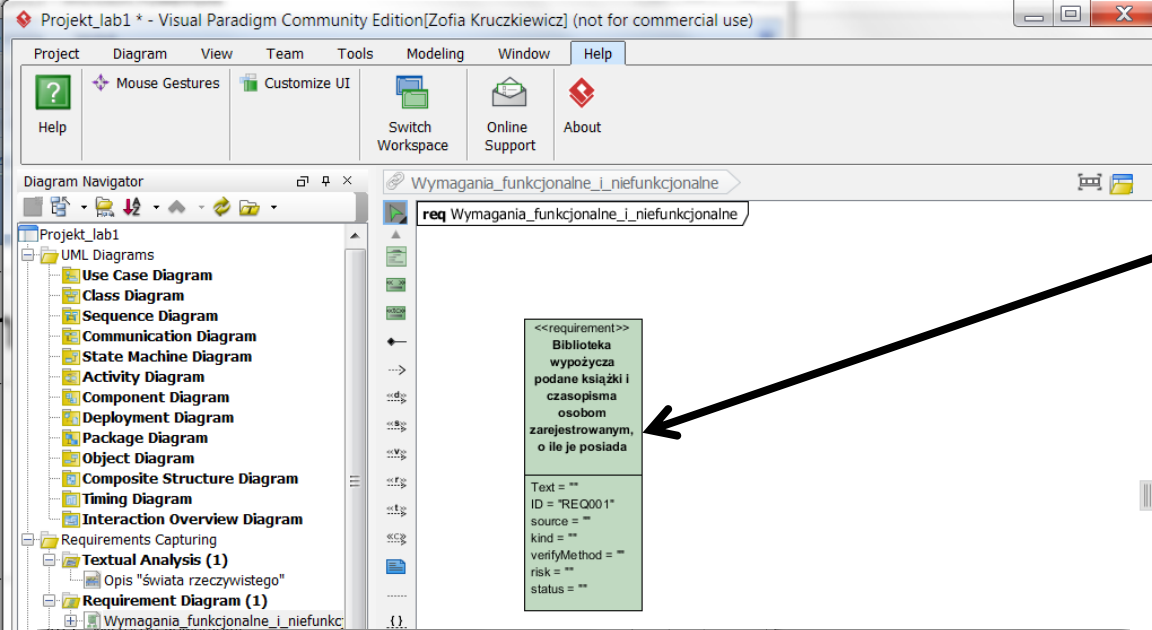


8.4. Należy umieścić na powierzchni diagramu ikony wymagań funkcjonalnych przeciągając je z palety umieszczonej z lewej strony diagramu i upuszczając na powierzchni diagramu

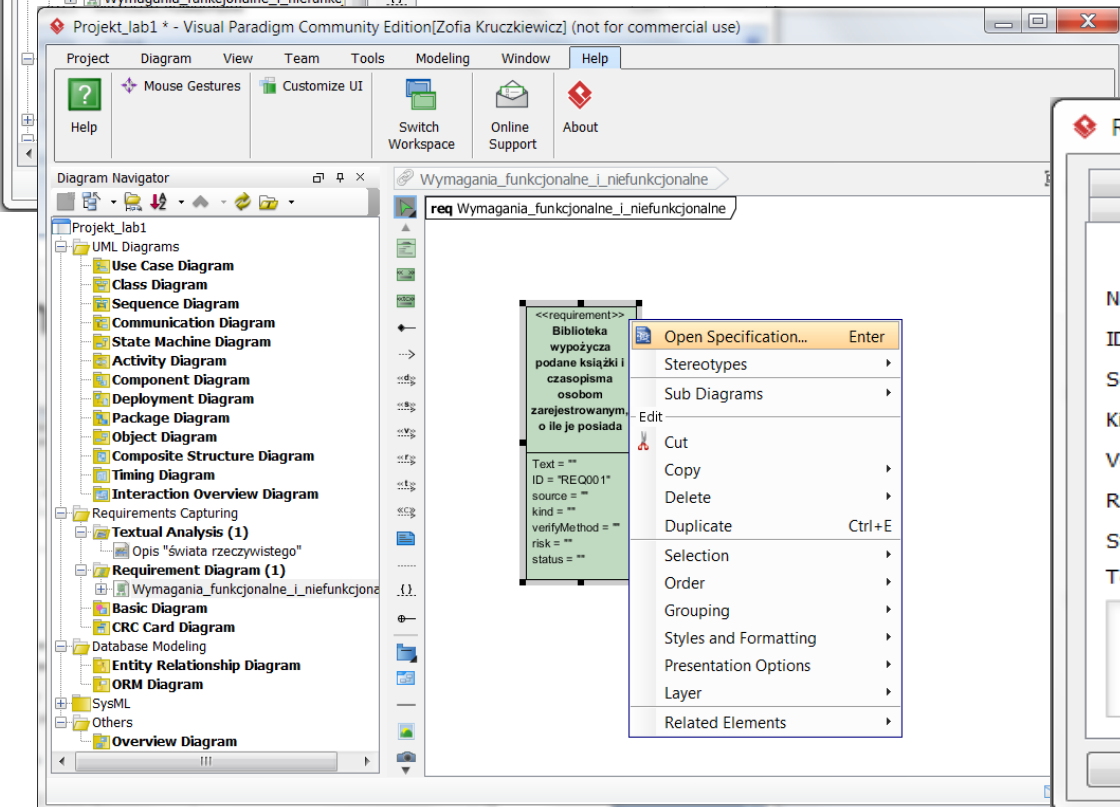
The image consists of two screenshots from the Visual Paradigm Community Edition software, illustrating the process of adding a requirement icon to a diagram.

Top Screenshot: Shows the software interface with the "Diagram Navigator" on the left. The "Requirement Diagram (1)" folder is expanded, showing a palette of requirement icons. A tooltip for the selected icon reads: "Requirement - a system/application function to be satisfied (Q)". A black arrow points from the text "z palety umieszczonej z lewej strony diagramu" in the title to this palette.

Bottom Screenshot: Shows the same software interface, but the requirement icon has been placed on the diagram surface. The icon is a green box with a black border, containing the text: `<<requirement>>`, **Requirement**, and several attributes: `Text = ""`, `ID = "REQ001"`, `source = ""`, `kind = ""`, `verifyMethod = ""`, `risk = ""`, and `status = ""`. A black arrow points from the text "i upuszczając na powierzchni diagramu" in the title to this icon.



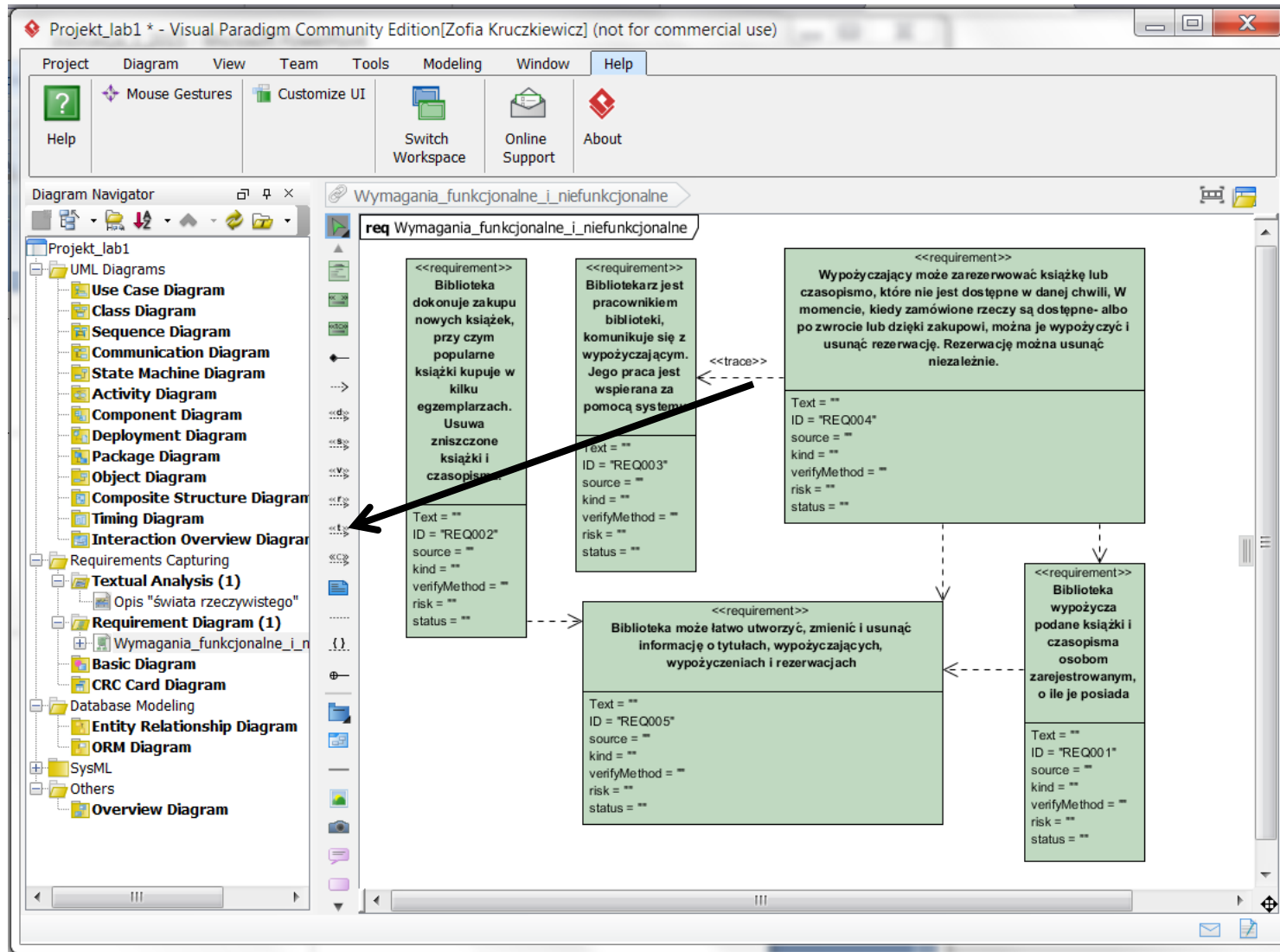
8.5. Po umieszczeniu ikony należy wypełnić puste pola bezpośrednio na ikonie lub po kliknięciu prawym klawiszem myszy wykonać to za pomocą wybranego formularza *Open Specification/Requirement Specification*



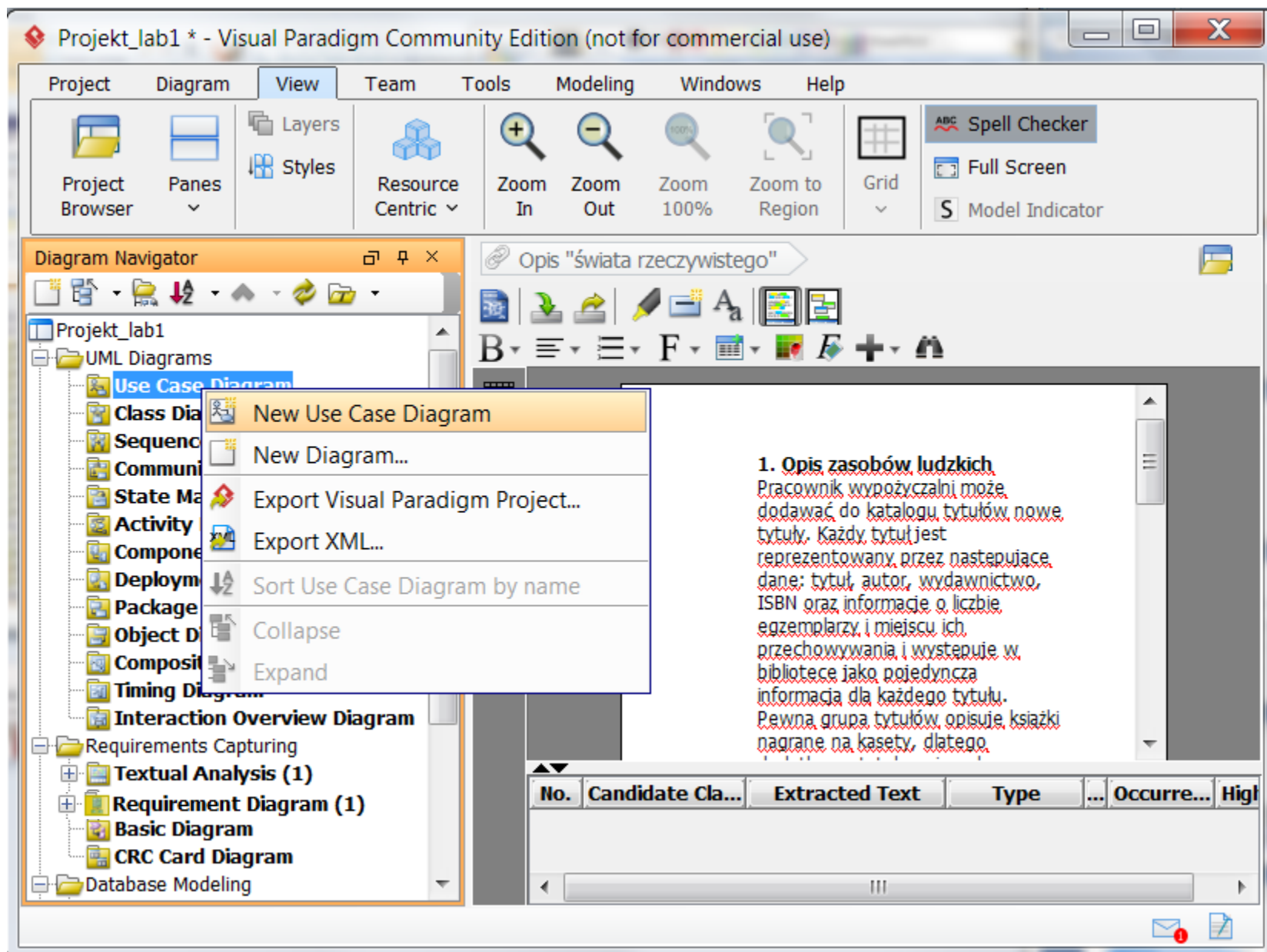
Requirement Specification

Project Management		Quality	Comments
Attributes	Constraints	Diagrams	References
General	Relations	Chart Relations	Stereotypes
Name:	czasopisma osobom zarejestrowanym, o ile je posiada		
ID:	REQ001		
Source:			
Kind:	▼		
Verify Method:	▼		
Risk:	▼		
Status:	▼		
Text:	 		
Reset		OK	Cancel
		Apply	Help

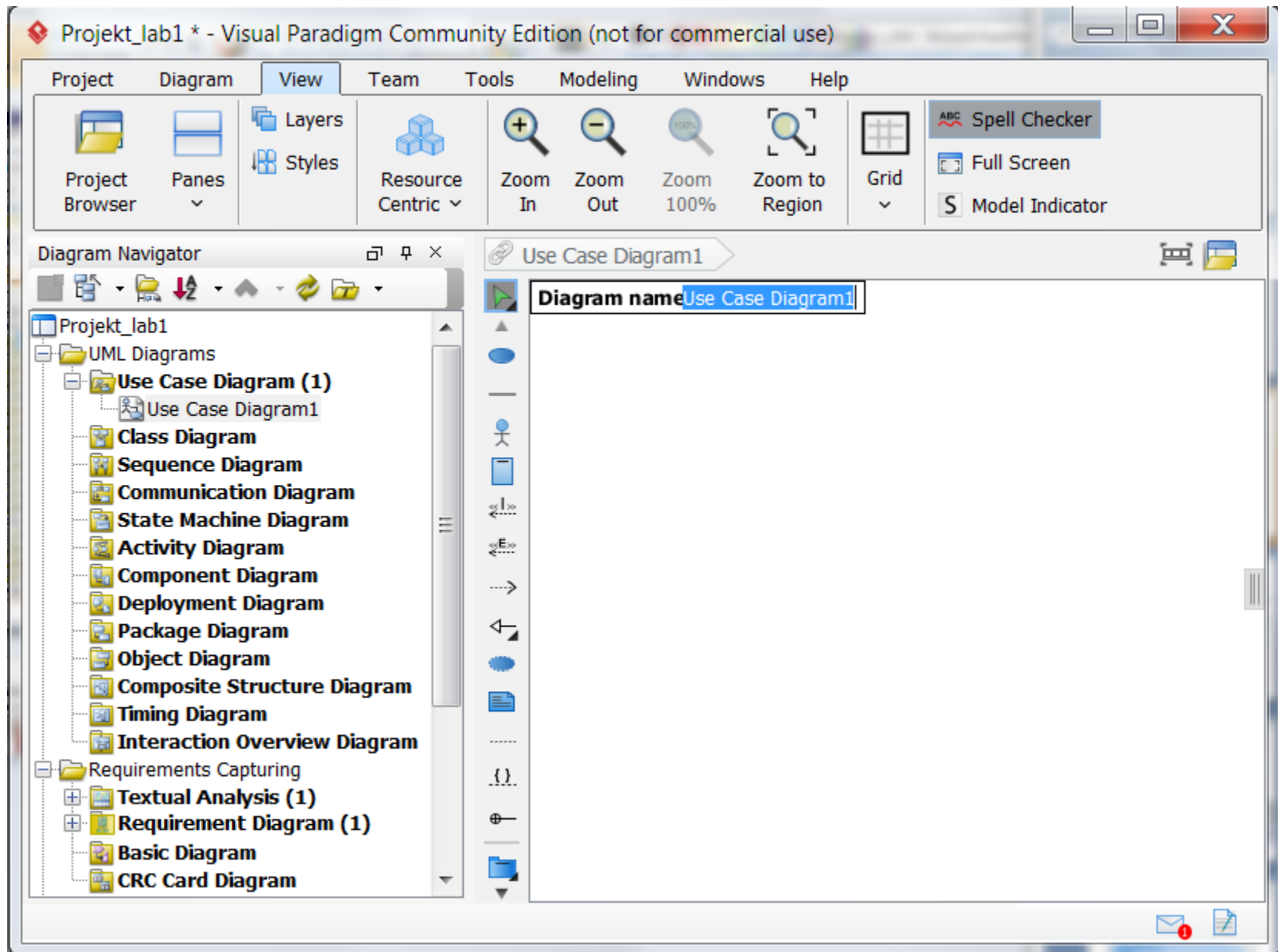
8.6. Rezultat – wykonano powiązania pomiędzy poszczególnymi wymaganiami za pomocą relacji asocjacji oraz <<trace>> przeciągając je z palety z lewej strony diagramu, następnie upuszczając je na powierzchni diagramu i łącząc nimi poszczególne wymagania.



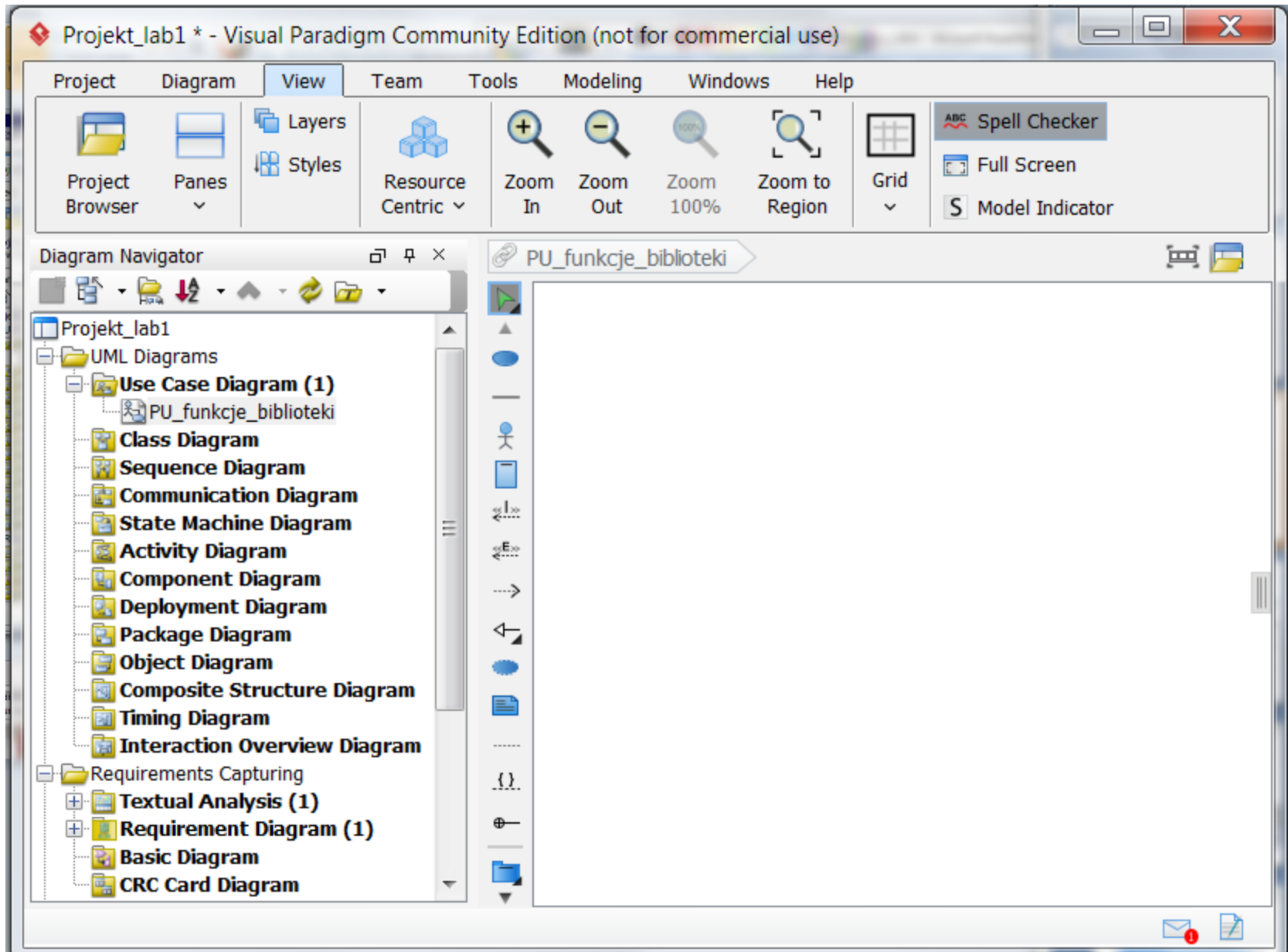
9/9.1. Wstawianie do projektu diagramu typu *Use Case* – po kliknięciu na nazwę diagramu wybór z list: *New Use Case Diagram*

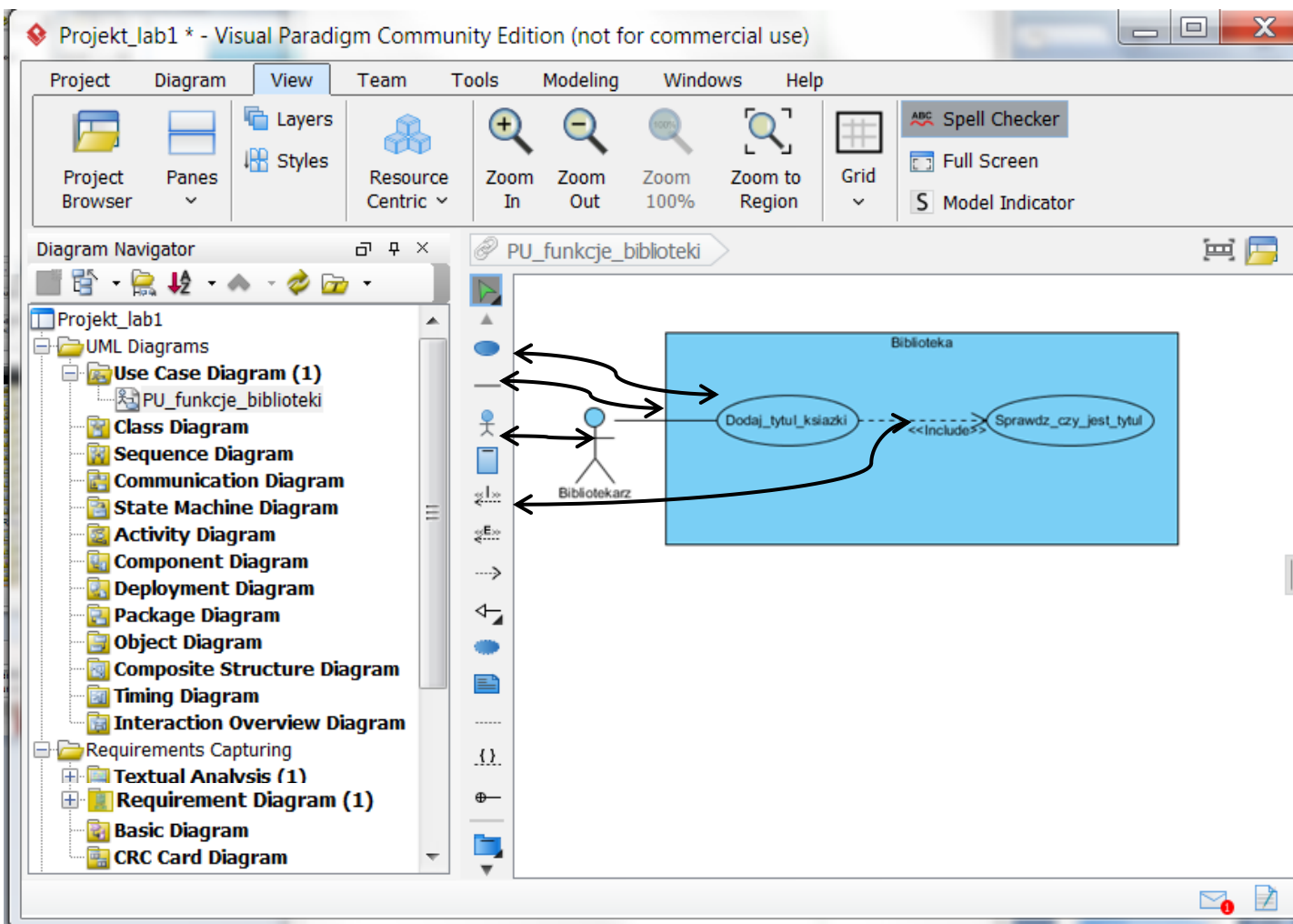


9.2. Wstawianie do projektu diagramu typu *Use Case* – nadanie nazwy diagramowi typu *Use Case*



9.3. Wstawianie do projektu diagramu typu *Use Case* – nadanie nazwy diagramowi typu *Use Case* np..**PU_funkcje_biblioteki**

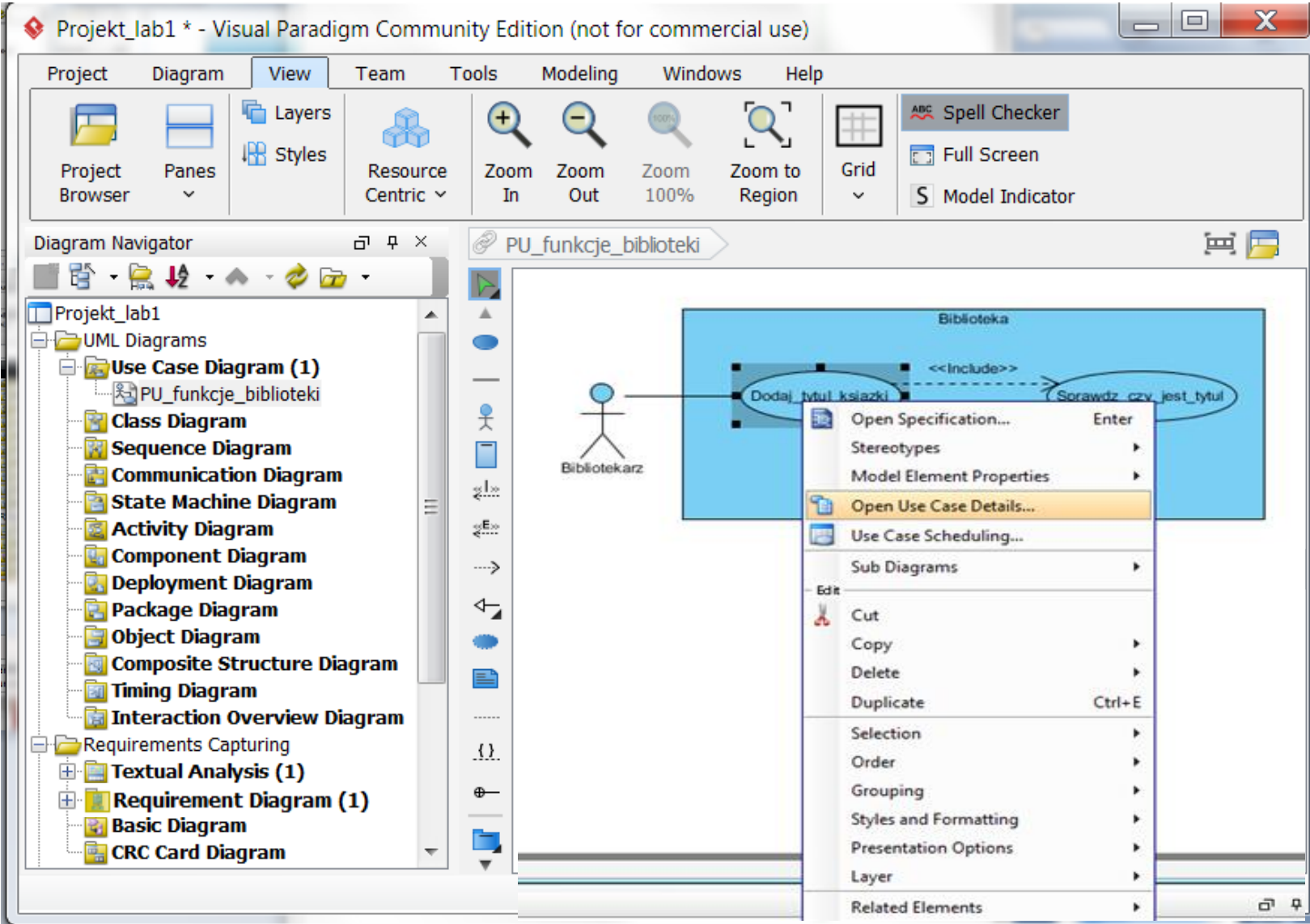




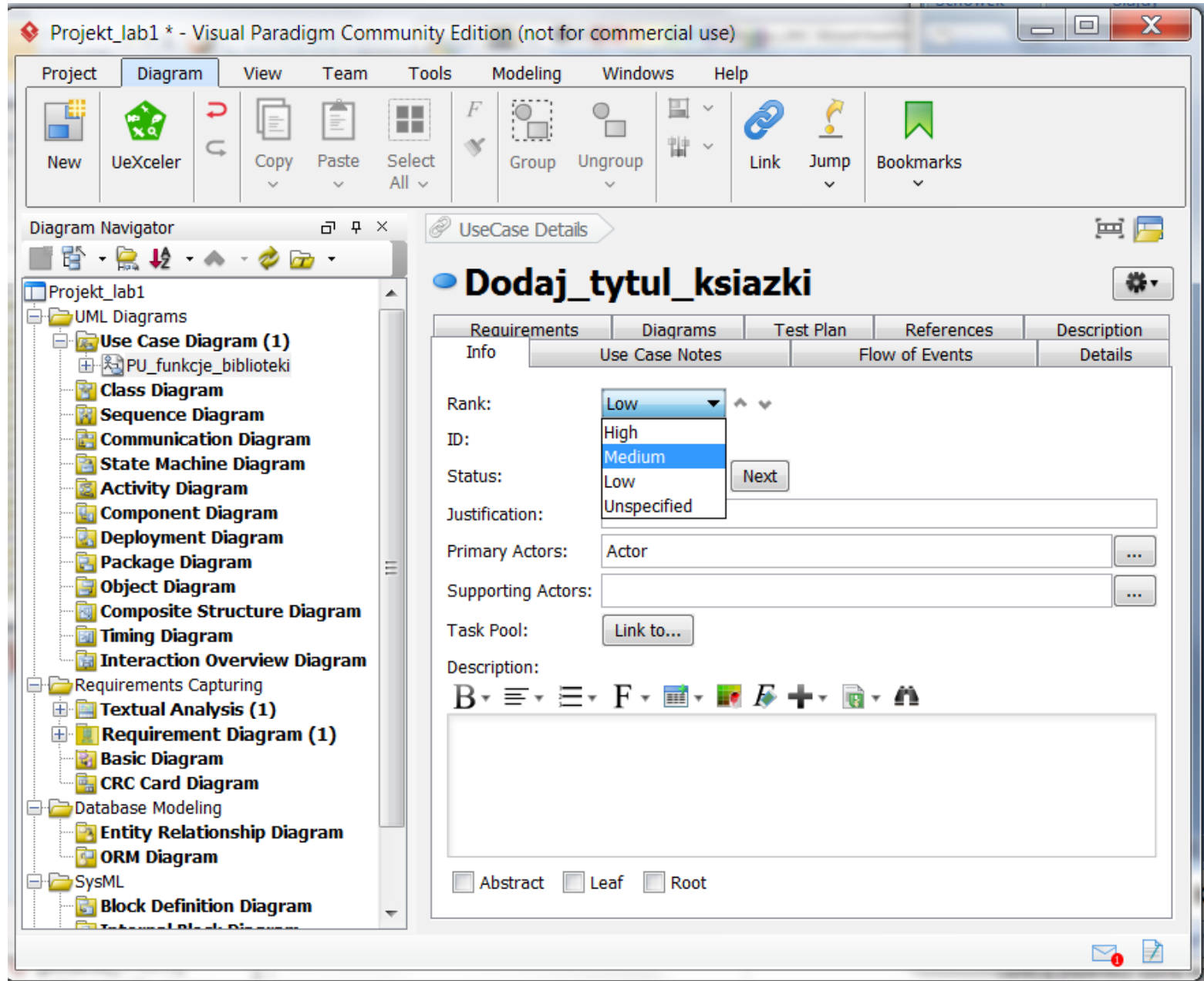
9.4. Wstawianie do projektu diagramu typu *Use Case* – definicja przypadków użycia specyfikujących wymagania stawiane aplikacji (p.8.4) w zakresie wstawiania zasobów biblioteki

Przecignięcie ikon **Actor**, **System**, **Use Case**, pobranych z palety lewym klawiszem myszy i upuszczenie na diagramie PU. Należy nadać im podane nazwy. Następnie, należy połączyć element **Actor** z PU **Dodaj_tytul_ksiazki** relacją **Association**, przeciągając ją z palety (z lewej strony) lewym klawiszem myszy i następnie położyć ją na elemencie **Actor** i przeciągnąć na PU **Dodaj_tytul_ksiazki**. Podobnie należy połączyć PUs relacją **<<Include>>**, przeciągając ją z palety - i następnie należy ją położyć na PU **Dodaj_tytul_ksiazki** i przeciągnąć do PU **Sprawdz_czy_jest_tytul**.

9.5 Definiowanie elementów typu Use Case – po kliknięciu prawym klawiszem myszy na PU *Dodaj_tytul_ksiazki* należy dokonać wyboru z listy opcji *Open Use Case Details*



9.6. Nadanie wagi diagramowi – wybór wartości z listy w polu *Rank*



The screenshot displays the Visual Paradigm Community Edition interface. The main window title is "Projekt_lab1 * - Visual Paradigm Community Edition (not for commercial use)". The menu bar includes Project, Diagram, View, Team, Tools, Modeling, Windows, and Help. The toolbar contains icons for New, UeXceler, Copy, Paste, Select All, Group, Ungroup, Link, Jump, and Bookmarks.

The Diagram Navigator on the left shows a tree structure for "Projekt_lab1" with folders for UML Diagrams, Requirements Capturing, Database Modeling, and SysML. Under "UML Diagrams", the "Use Case Diagram (1)" folder is expanded, showing a list of diagram types including "PU_funkcje_biblioteki", "Class Diagram", "Sequence Diagram", "Communication Diagram", "State Machine Diagram", "Activity Diagram", "Component Diagram", "Deployment Diagram", "Package Diagram", "Object Diagram", "Composite Structure Diagram", "Timing Diagram", "Interaction Overview Diagram", "Textual Analysis (1)", "Requirement Diagram (1)", "Basic Diagram", "CRC Card Diagram", "Entity Relationship Diagram", "ORM Diagram", and "Block Definition Diagram".

The "UseCase Details" panel on the right is titled "Dodaj_tytul_ksiazki". It features a tabbed interface with tabs for Requirements, Diagrams, Test Plan, References, and Description. The "Info" tab is active, showing a form with the following fields:

- Rank: A dropdown menu is open, showing options: Low, High, Medium (highlighted), Low, and Unspecified. A "Next" button is located to the right of the dropdown.
- ID: A text input field.
- Status: A text input field.
- Justification: A text input field.
- Primary Actors: A text input field with a "..." button.
- Supporting Actors: A text input field with a "..." button.
- Task Pool: A "Link to..." button.
- Description: A rich text editor with a toolbar containing icons for Bold (B), Italic (I), Underline (U), Font Color (F), Background Color, Bulleted List, Numbered List, Indent, and Undo.

At the bottom of the "Info" tab, there are three checkboxes: "Abstract", "Leaf", and "Root".

9.7. Wpisanie do podformularza *Info* w części *Description* scenariusza przypadku użycia *Dodaj_tytul*

The screenshot shows the Visual Paradigm Community Edition interface. The main window displays the 'Use Case Details' dialog for the use case 'Dodaj_tytul_książki'. The dialog is divided into several sections:

- Diagram Navigator:** Shows a tree view of the project 'Projekt_lab1' with various UML diagrams and models.
- Use Case Details:** Contains the following fields:
 - Rank:** Medium
 - ID:** UC01
 - Status:** Identify
 - Justification:** (empty)
 - Primary Actors:** Actor
 - Supporting Actors:** (empty)
 - Task Pool:** Link to...
- Description:** Contains a text area with the following text:

Scenariusz:
1. Należy podać dane tytułu: imię i nazwisko autora, tytuł książki, wydawnictwo, ISBN
2. Należy sprawdzić, czy dane wprowadzanego tytułu są unikalne za pomocą wywołania PU Sprawdź, czy jest tytuł, przekazując ISBN tytułu
3. Jeśli tytuł o podanym ISBN istnieje, należy zakończyć przypadek użycia, w przeciwnym razie należy zapisać dane.
- Options:** Includes checkboxes for Abstract, Leaf, and Root.

9.8. Wybór podformularza *Details* związanego z wybranym wcześniej PU

The screenshot displays the Visual Paradigm Community Edition interface. The main window title is 'Projekt_lab1 * - Visual Paradigm Community Edition (not for commercial use)'. The menu bar includes Project, Diagram, View, Team, Tools, Modeling, Windows, and Help. The toolbar contains icons for New, UeXceler, Copy, Paste, Select All, Group, Ungroup, Link, Jump, and Bookmarks.

The Diagram Navigator on the left shows a tree structure for 'Projekt_lab1' with folders for UML Diagrams, Requirements Capturing, Database Modeling, and SysML. Under UML Diagrams, 'Use Case Diagram (1)' is expanded to show 'PU_funkcje_biblioteki'.

The 'UseCase Details' form is open, showing the title 'Dodaj_tytul_książki'. The form has tabs for Requirements, Diagrams, Test Plan, References, and Description. The Description tab is active, showing a table with columns for Info, Use Case Notes, Flow of Events, and Details. The Details column is selected, displaying the following fields:

Info	Use Case Notes	Flow of Events	Details
Level:			<input type="text"/>
Complexity:			<input type="text"/>
Use Case Status:			<input type="text"/>
Implementation Status:			<input type="text"/>
Preconditions:			<input type="text"/>
Post-conditions:			<input type="text"/>
Author:			<input type="text"/>
Assumptions:			<input type="text"/>

9.9. Wybór podformularza *Details* związanego z wybranym wcześniej PU – nadanie wartości poszczególnym polom formularza przez wybór z listy lub wprowadzenie tekstu

The screenshot displays the Visual Paradigm Community Edition interface. The main window title is "Projekt_lab1 * - Visual Paradigm Community Edition (not for commercial use)". The menu bar includes Project, Diagram, View, Team, Tools, Modeling, Windows, and Help. The toolbar contains icons for New, UeXceler, Copy, Paste, Select All, Group, Ungroup, Link, Jump, and Bookmarks. The Diagram Navigator on the left shows a tree structure for "Projekt_lab1" with folders for UML Diagrams, Requirements Capturing, Database Modeling, and SysML. The "UML Diagrams" folder is expanded, showing a "Use Case Diagram (1)" containing a "PU_funkcje_biblioteki" use case. The main workspace displays the "Dodaj_tytul_ksiadzki Details" form, which is a sub-form of the "Dodaj_tytul_ksiadzki" diagram. The form has a tabbed interface with tabs for Requirements, Diagrams, Test Plan, References, and Description. The "Description" tab is active, showing a "Details" sub-tab. The form fields are as follows:

Requirements	Diagrams	Test Plan	References	Description
Info	Use Case Notes	Flow of Events		Details
Level:	User			
Complexity:	Medium			
Use Case Status:	Complete			
Implementation Status:	Partially Complete			
Preconditions:	Uruchomienie aplikacji www luib aplikacji w architekturze typu klient-serwer			
Post-conditions:	Wprowadzenie danych tytułu o unikatowym ISBN			
Author:	Zofia Kruczkiewicz			
Assumptions:				

9.10. Powiązanie *Precondition* z wybranymi wymaganiami z diagramu wymagań (*Requirements Diagram*) – należy kliknąć na powierzchnię pola *Precondition* i kliknąć na przycisk *Insert Requirement...*

The screenshot displays the Visual Paradigm Community Edition interface. The main window title is "Projekt_lab1 * - Visual Paradigm Community Edition[Zofia Kruczkiewicz] (not for commercial use)". The menu bar includes Project, Diagram, View, Team, Tools, Modeling, Window, and Help. The toolbar contains icons for Help, Mouse Gestures, Customize UI, Switch Workspace, Online Support, and About.

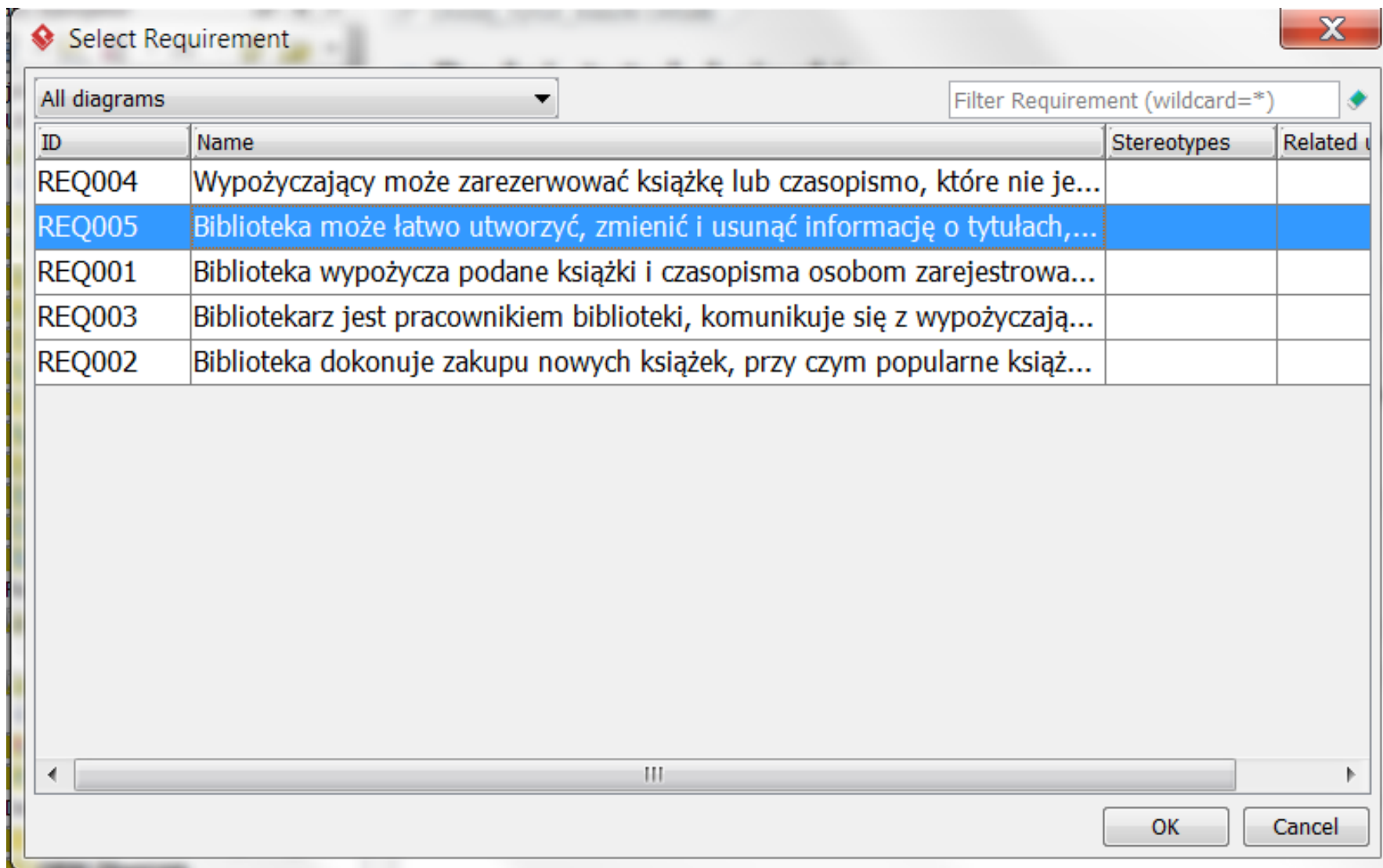
The Diagram Navigator on the left shows a tree structure for "Projekt_lab1" with various UML Diagrams and Requirements Capturing elements. The main diagram area is titled "Dodaj_tytuł_książki Details". The diagram has several tabs: Info, Use Case Notes, Flow of Events, Details (selected), Requirements, Diagrams, Test Plan, References, and Description.

The "Details" tab shows the following fields and values:

- Level: User
- Complexity: Medium
- Use Case Status: Complete
- Implementation Status: Partially Complete
- Preconditions: U uruchomienie aplikacji www lub aplikacji w architekturze typu klient-serwer
- Post-conditions: Wprowadzenie danych tytułu o unikatowym ISBN
- Author: Zofia Kruczkiewicz
- Assumptions: (empty)

Buttons for "Insert Requirement..." and "Insert Use Case..." are visible below the Preconditions field.

9.11. Wybór wymagań



9.12. Rezultat dodania wybranego wymagania do pola *Precondition*

The screenshot displays the Visual Paradigm Community Edition interface. The main window title is "Projekt_lab1 * - Visual Paradigm Community Edition[Zofia Kruczkiewicz] (not for commercial use)". The menu bar includes Project, Diagram, View, Team, Tools, Modeling, Window, and Help. The toolbar contains icons for Help, Mouse Gestures, Customize UI, Switch Workspace, Online Support, and About.

The Diagram Navigator on the left shows a tree structure for "Projekt_lab1" with folders for UML Diagrams, Requirements Capturing, Database Modeling, and SysML. Under UML Diagrams, "Use Case Diagram (1)" is expanded, showing "PU_funkcje_biblioteki". Under Requirements Capturing, "Requirement Diagram (1)" is expanded, showing "Wymagania_funkcjonalne_i_n".

The main workspace displays the configuration for a requirement diagram titled "Dodaj_tytuł_książki". The configuration is organized into tabs: Info, Use Case Notes, Flow of Events, Details, Requirements, Diagrams, Test Plan, References, and Description. The "Details" tab is active, showing the following fields:

- Level: User
- Complexity: Medium
- Use Case Status: Complete
- Implementation Status: Partially Complete
- Preconditions: U uruchomienie aplikacji www lub aplikacji w architekturze typu klient-serwer [Biblioteka może łatwo utworzyć, zmienić i usunąć informacje o tytułach, wypożyczających, wypożyczenia](#)
- Post-conditions: Wprowadzenie danych tytułu o unikatowym ISBN
- Author: Zofia Kruczkiewicz
- Assumptions:

Buttons for "Insert Requirement..." and "Insert Use Case..." are visible below the Preconditions field.

9.13. Po kliknięciu prawym klawiszem myszy na PU *Dodaj_tytul_ksiazki* należy wybrać z listy opcję *Open Specification* – powinien ukazać tekst wprowadzony wcześniej w podformularzu *Info* opcji *Open Use Case Details*

Use Case Specification

Project Management Quality Comments
Tagged Values Constraints Diagrams References
General Extension Points Relations Chart Relations Stereotypes

Name: Dodaj_tytul_ksiazki

ID: UC01

Rank: Medium

Description:

B **F**

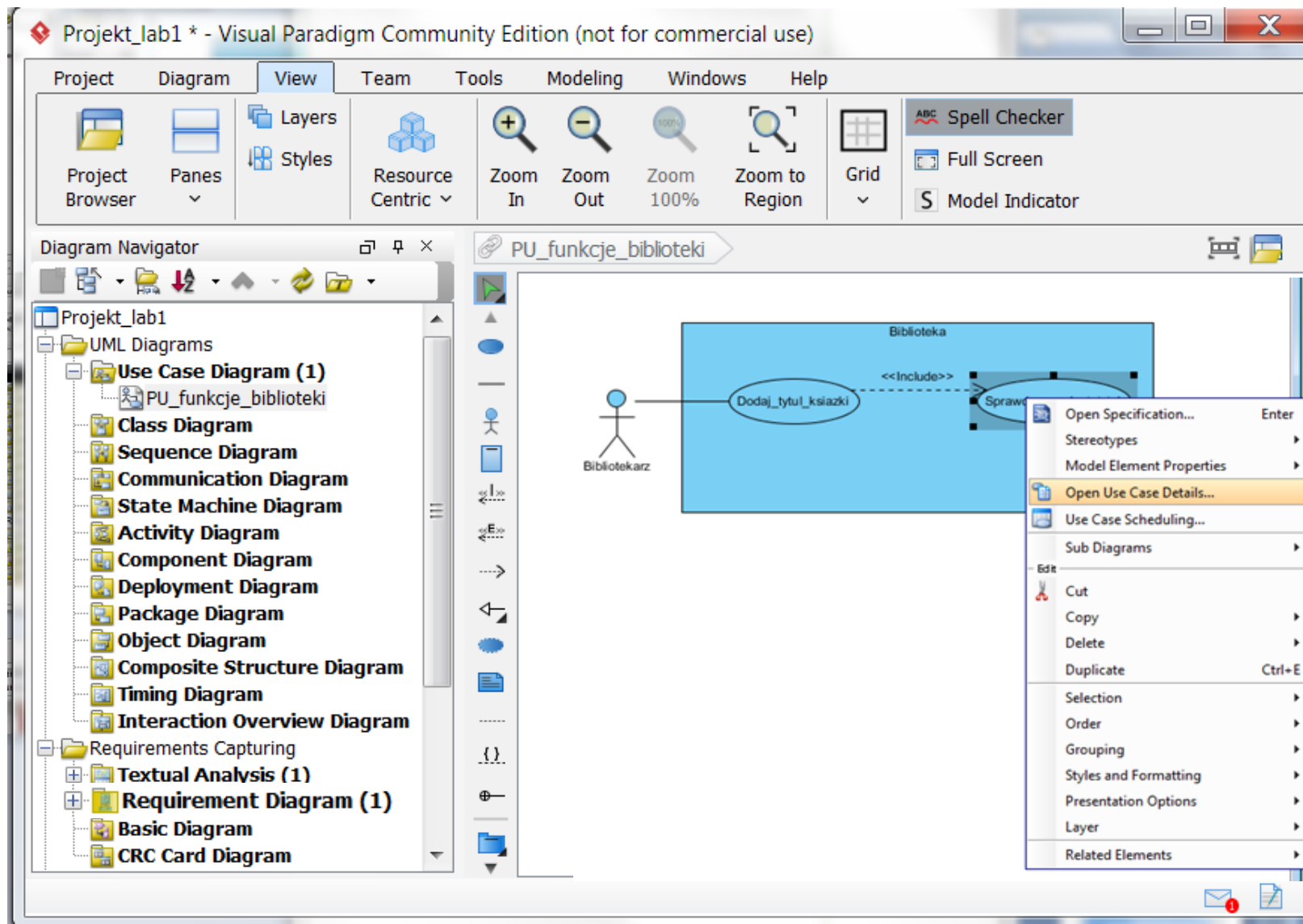
Scenariusz:

1. Należy podać dane tytułu: imię i nazwisko autora, tytuł książki, wydawnictwo, ISBN
2. Należy sprawdzić, czy dane wprowadzanego tytułu są unikalne za pomocą wywołania PU Sprawdź, czy jest tytuł, przekazując ISBN tytułu
3. Jeśli tytuł o podanym ISBN istnieje, należy zakończyć przypadek użycia, w przeciwnym razie należy zapisać dane.

Abstract Leaf Root Business model

Reset OK Cancel Apply Help

9.14. Definiowanie elementów typu Use Case – po kliknięciu prawym klawiszem myszy na PU *Sprawdz_czy_jest_tytul* należy dokonać wyboru z listy opcji *Open Use Case Details*



9.15. Wybór podformularza *Details* związanego z wybranym wcześniej PU – nadanie wartości poszczególnym polom formularza przez wybór z listy lub wprowadzenie tekstu np. po kliknięciu na przycisk *Insert Use Case...* i wyborze przypadku użycia z listy, z którego wywoływany jest *PU Sprawdz_czy_jest_tytul*

The screenshot displays the Visual Paradigm Community Edition interface. The main window shows the 'Sprawdz_czy_jest_tytul Details' form with the following fields:

- Level: Subfunction
- Complexity: Medium
- Use Case Status: Complete
- Implementation Status: Partially Complete
- Preconditions: Dodaj_tytul_książki
- Post-conditions: Zwraca wynik, określający, czy podany ISBN jest unikatowy lub podaje informację, że dany ISBN już istnieje
- Author: Zofia Kruczkiewicz
- Assumptions:

The 'Select Use Case' dialog box is open, showing a table of use cases:

ID	Name	Primary actors	Supportin
UC01	Dodaj_tytul_książki	Bibliotekarz	
UC02	Sprawdz_czy_jest_tytul		

Arrows indicate the flow of information: one arrow points from the 'Dodaj_tytul_książki' entry in the dialog to the 'Preconditions' field in the main form, and another arrow points from the 'Insert Use Case...' button in the main form to the dialog box.

9.16. Dodanie wymagań do pola *Precondition*

The screenshot shows the Visual Paradigm Community Edition interface. The main window displays the details for a use case titled "Sprawdz_czy_jest_tytul". The interface includes a menu bar (Project, Diagram, View, Team, Tools, Modeling, Window, Help) and a toolbar with icons for Help, Mouse Gestures, Customize UI, Switch Workspace, Online Support, and About. The Diagram Navigator on the left shows a project tree with folders for UML Diagrams, Requirements Capturing, Database Modeling, SysML, and Others. The main area has tabs for Info, Use Case Notes, Flow of Events, Details, Requirements, Diagrams, Test Plan, References, and Description. The Details tab is active, showing the following fields:

- Level: Subfunction
- Complexity: Medium
- Use Case Status: Complete
- Implementation Status: Partially Complete
- Preconditions: Dodaj tytuł książki
- Post-conditions: Zwraca wynik, określający, czy podany ISBN jest unikatowy lub podaje informację, czy dany ISBN już istnieje
- Author: Zofia Kruczkiewicz
- Assumptions: (empty)

Buttons for "Insert Requirement..." and "Insert Use Case..." are visible below the Preconditions field.

9.17. Rezultat

The screenshot displays the Visual Paradigm Community Edition interface. The main window title is "Projekt_lab1 * - Visual Paradigm Community Edition[Zofia Kruczkiewicz] (not for commercial use)". The menu bar includes Project, Diagram, View, Team, Tools, Modeling, Window, and Help. The toolbar contains icons for Help, Mouse Gestures, Customize UI, Switch Workspace, Online Support, and About.

The Diagram Navigator on the left shows a project tree for "Projekt_lab1" with a folder "UML Diagrams" containing a "Use Case Diagram (1)". The diagram is titled "Sprawdz czy jest tytul". The details view for this use case is shown on the right, with tabs for Info, Use Case Notes, Flow of Events, Details, Requirements, Diagrams, Test Plan, References, and Description. The "Details" tab is active, showing the following information:

- Level: Subfunction
- Complexity: Medium
- Use Case Status: Complete
- Implementation Status: Partially Complete
- Preconditions:
 - [Dodaj tytul książki](#)
 - [Biblioteka może łatwo utworzyć, zmienić i usunąć informacje o tytułach, wypożyczających, wypożyczeniach](#)
- Post-conditions: Zwraca wynik, określający, czy podany ISBN jest unikatowy lub podaje informację, czy dany ISBN już istnieje
- Author: Zofia Kruczkiewicz
- Assumptions:

Buttons for "Insert Requirement..." and "Insert Use Case..." are visible below the preconditions section.

9.18. Wpisanie do podformularza *Info* w części *Description* scenariusza przypadku użycia PU *Sprawdz_czy_jest_tytul*

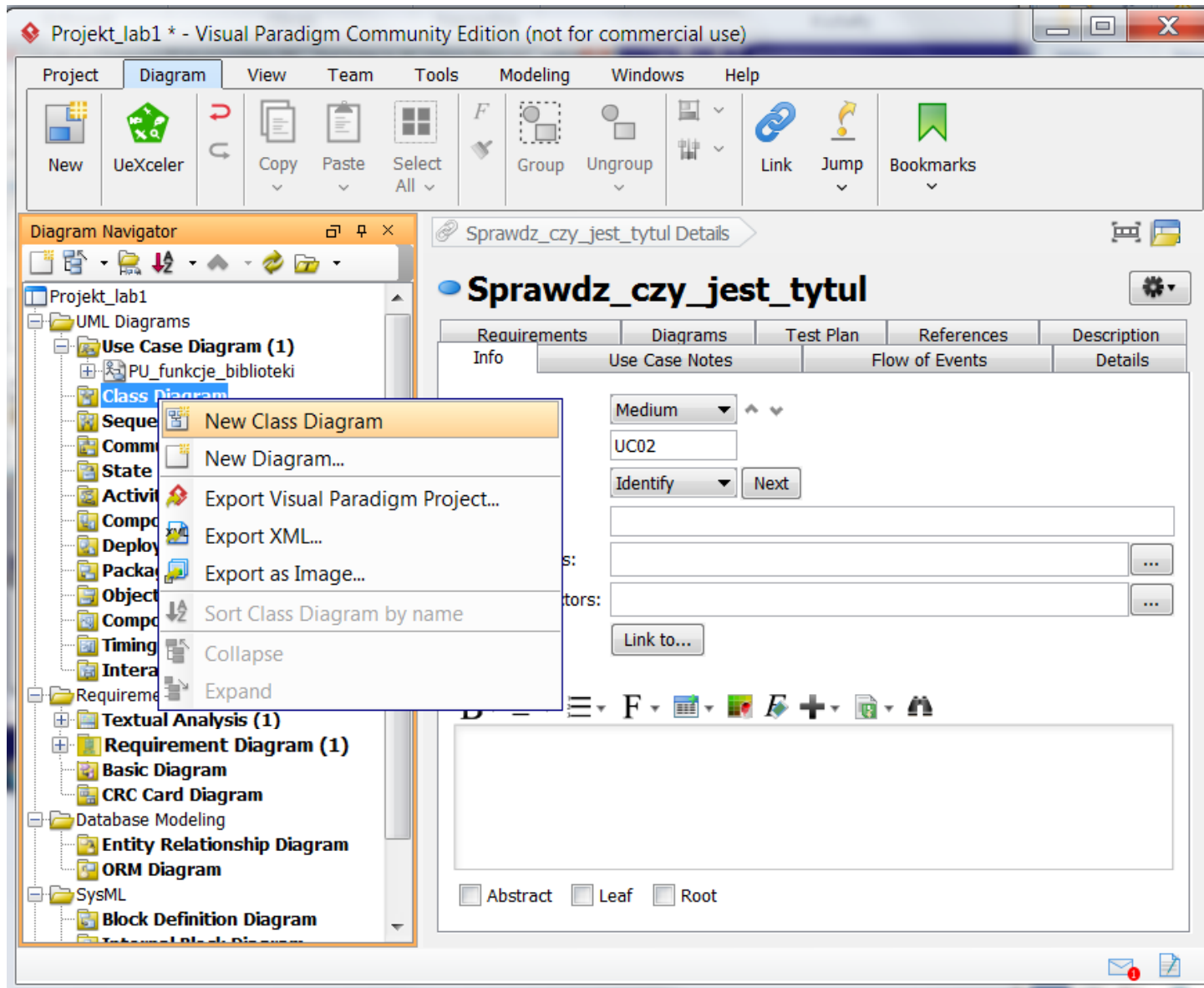
The screenshot displays the Visual Paradigm Community Edition interface. The main window title is 'Projekt_lab1 * - Visual Paradigm Community Edition (not for commercial use)'. The menu bar includes Project, Diagram, View, Team, Tools, Modeling, Windows, and Help. The toolbar contains icons for New, UeXcelers, Copy, Paste, Select All, Group, Ungroup, Link, Jump, and Bookmarks. The Diagram Navigator on the left shows a tree structure for 'Projekt_lab1' with folders for UML Diagrams, Requirements Capturing, Database Modeling, and SysML. The 'UML Diagrams' folder is expanded, showing 'Use Case Diagram (1)' with a sub-entry 'PU_funkcje_biblioteki'. The main workspace shows the 'Sprawdz_czy_jest_tytul Details' window. The 'Info' tab is active, displaying the following details:

Requirements	Diagrams	Test Plan	References	Description
Info	Use Case Notes	Flow of Events	Details	

Rank: Medium
ID: UC02
Status: Identify Next
Justification:
Primary Actors:
Supporting Actors:
Task Pool: Link to...
Description:
1. Porównuje ISBN podanego tytułu książki i numerami ISBN pozostałych tytułów książek, przechowywanych w bibliotece.
2. W przypadku znalezienia tytułu o takim samym numerze ISBN PU kończy przeglądanie numerów ISBN pozostałych tytułów książek i zwraca znaleziony tytuł książki.
3. W przypadku braku tytułu książki o podanym numerze ISBN, po przejrzaniu tytułów książek, zwracany jest wynik negatywny.

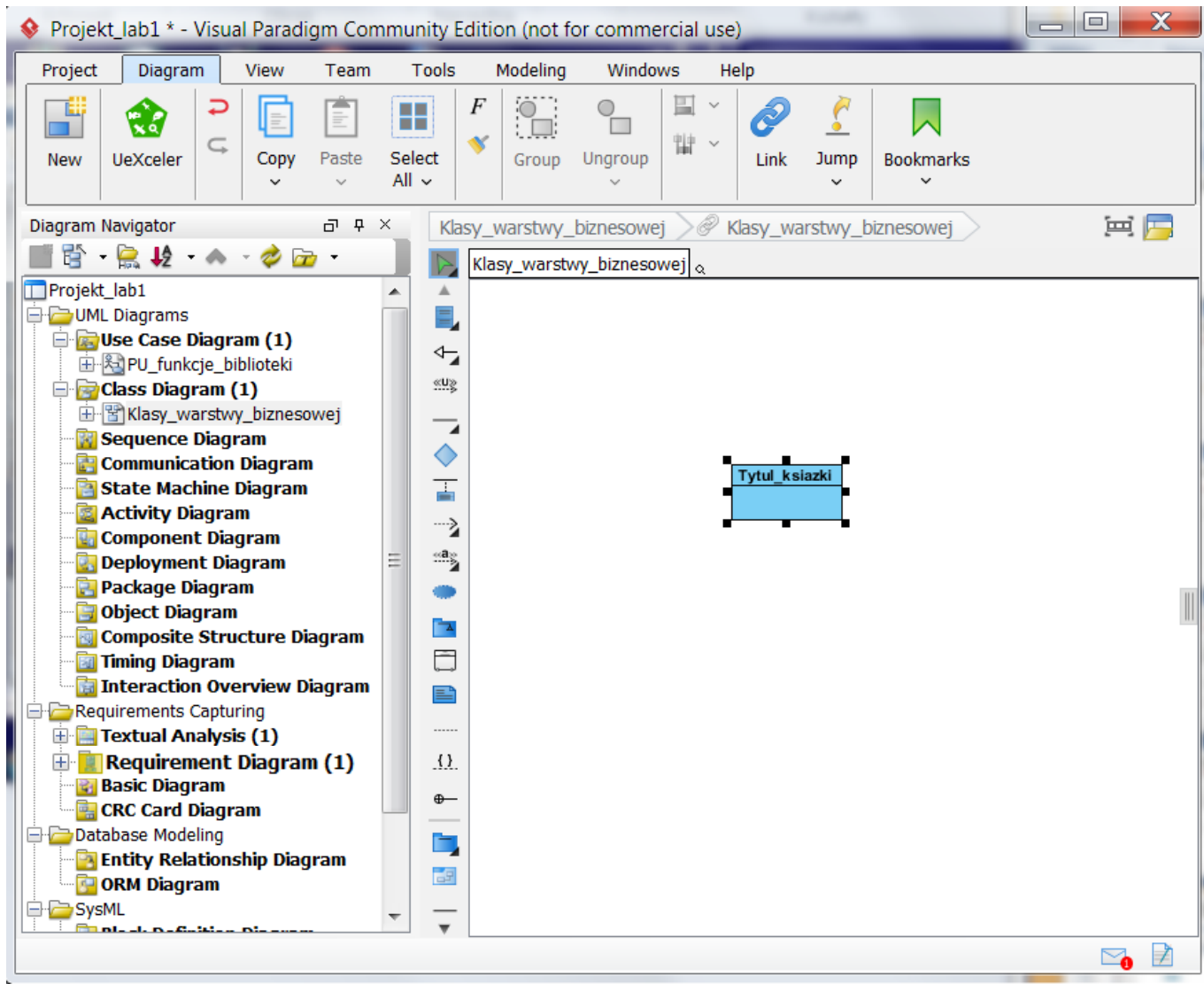
Abstract Leaf Root

10./10.1. Dodanie diagramu klas do projektu – należy kliknąć prawym klawiszem na nazwę diagramu w okienku *Diagram Navigator* i wybrać z list *New Class Diagram*

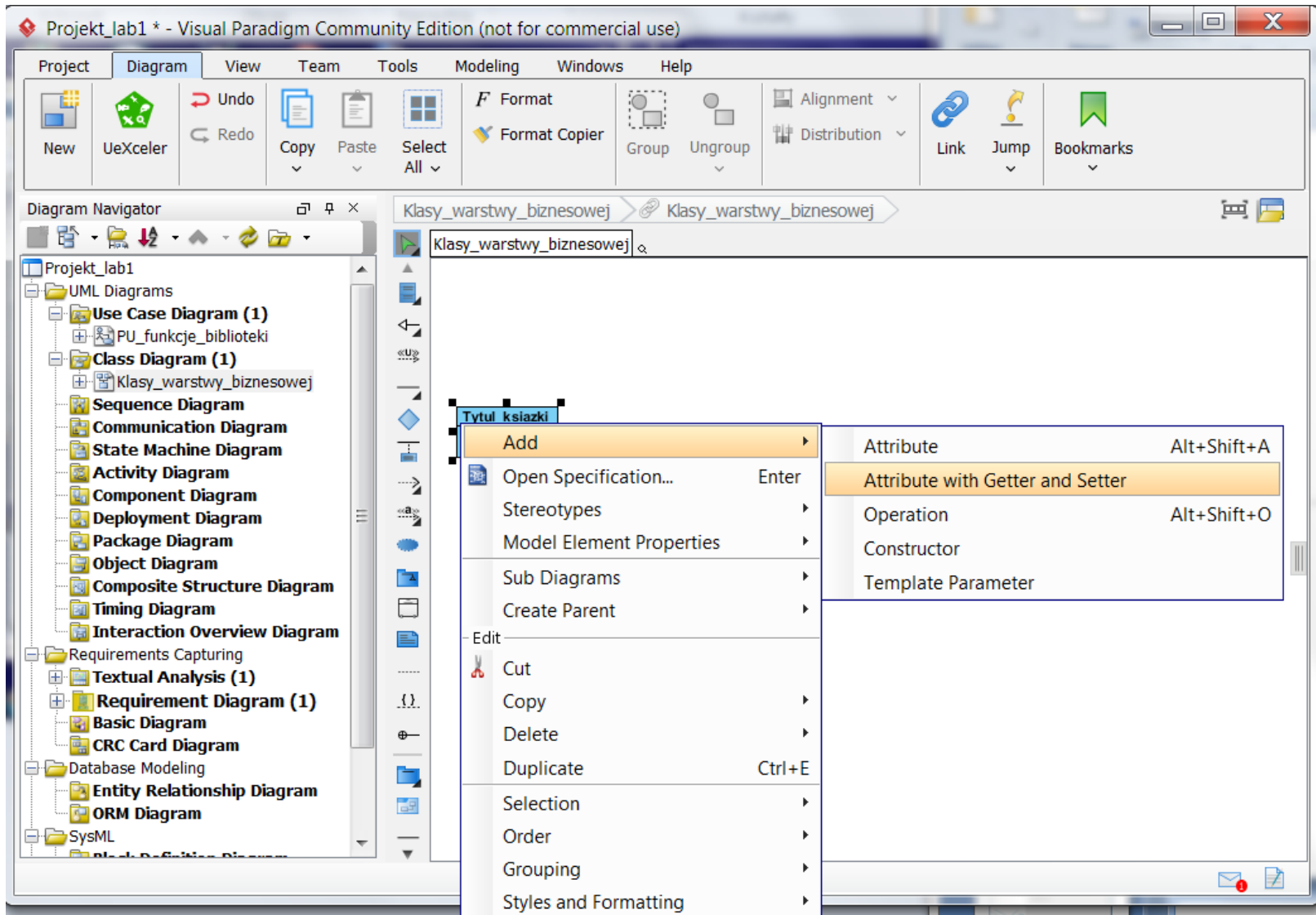


10.2. Po nadaniu nazwy diagramowi klas warstwy biznesowej jako

Klasy_warstwy_biznesowej należy zdefiniować klasy zidentyfikowane na podstawie scenariuszy przypadków użycia. Pierwsza definiowana klasa zawiera dane tytułu książki – należy przeciągnąć ikonę klasy z palety lewym klawiszem myszy i położyć na diagramie i nadać jej nazwę **Tytuł_książki**



10.3. Zdefiniowanie atrybutów i metod – po kliknięciu prawym klawiszem na klasę należy wybrać z listy pozycje *Attribute* do definiowania nowych atrybutów lub *Operation* do definiowania metod. Zdefiniowanie atrybutów i metod dostępu do atrybutów – po kliknięciu prawym klawiszem na klasę należy wybrać z listy pozycję *Attribute with Getter and Setter*



10.4. Dodano prywatne atrybuty i publiczne metody dostępu do atrybutów typu getter i setter klasie **Tytul_książki**

The screenshot displays the Visual Paradigm Community Edition interface. The main workspace shows a class diagram for the class **Tytul_książki**. The class is represented by a blue box with a title bar. The attributes are listed in the top section, and the methods are listed in the bottom section.

Attributes:

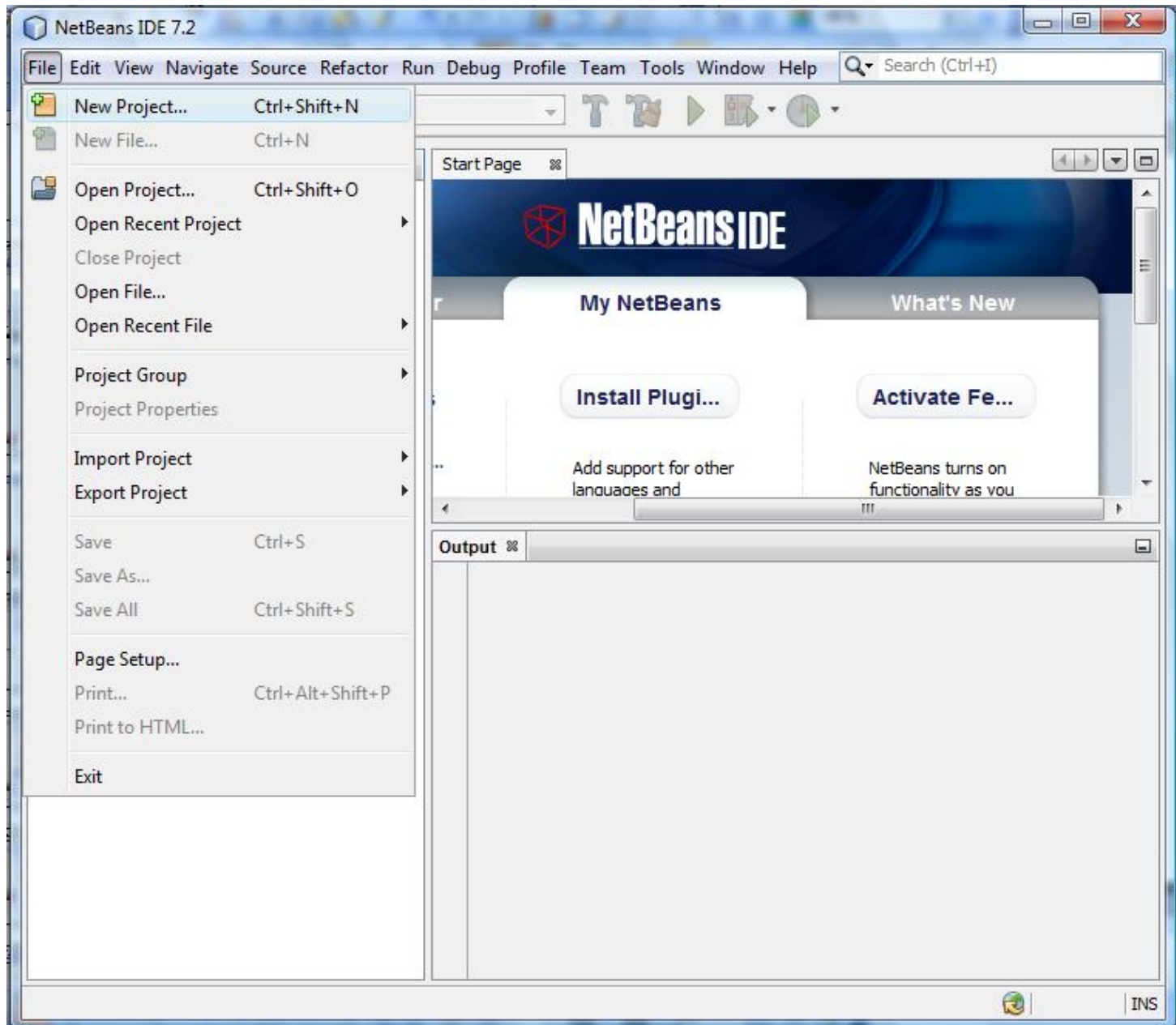
- tytul
- imie
- nazwisko
- wydawnictwo
- ISBN

Methods:

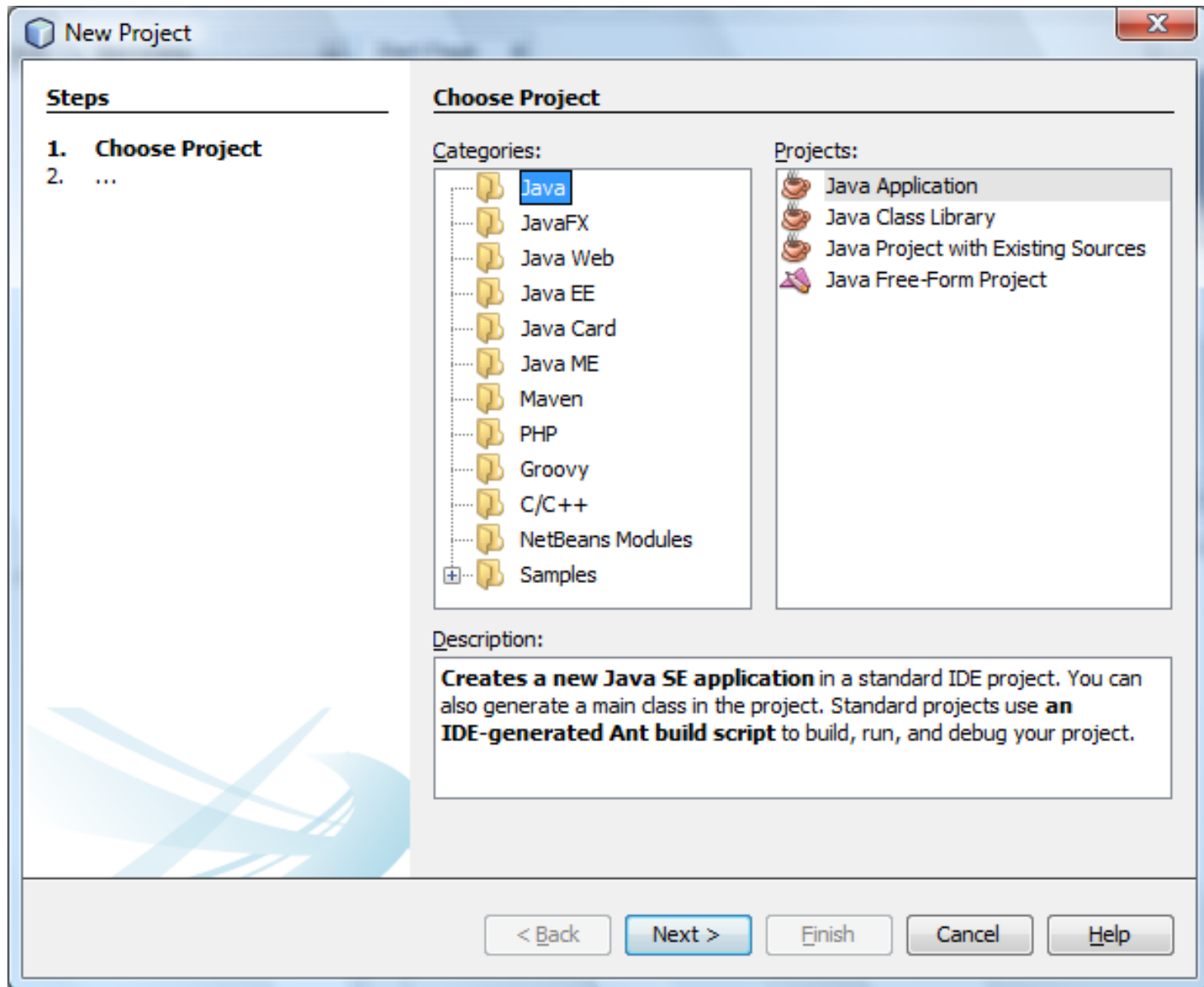
- +getTytul()
- +setTytul(tytul) : void
- +getImie()
- +setImie(imie) : void
- +getNazwisko()
- +setNazwisko(nazwisko) : void
- +getWydawnictwo()
- +setWydawnictwo(wydawnictwo) : void
- +getISBN()
- +setISBN(ISBN) : void

The Diagram Navigator on the left shows the project structure, including the package **Klasy_warstwy_biznesowej** containing the **Tytul_książki** class. The top toolbar contains various icons for creating, editing, and navigating diagrams.

11/11.1. Wykonanie projektu typu aplikacja Javy w środowisku typu NetBeans – *File/New Project*



11.2. Wykonanie projektu typu *Java/ Java Application* – wybór w kolumnie *Categories* pozycji *Java* oraz pozycji *Java Application* w kolumnie *Projects*



11.3. Wykonanie projektu typu *Java/ Java Application* – nadanie nazwy projektowi w polu *Project Name* oraz lokalizacji za pomocą klawiasza *Browse...* w polu *Project Location* bez ustawienia opcji *Create Main Class*

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name:

Project Location:

Project Folder:

Use Dedicated Folder for Storing Libraries

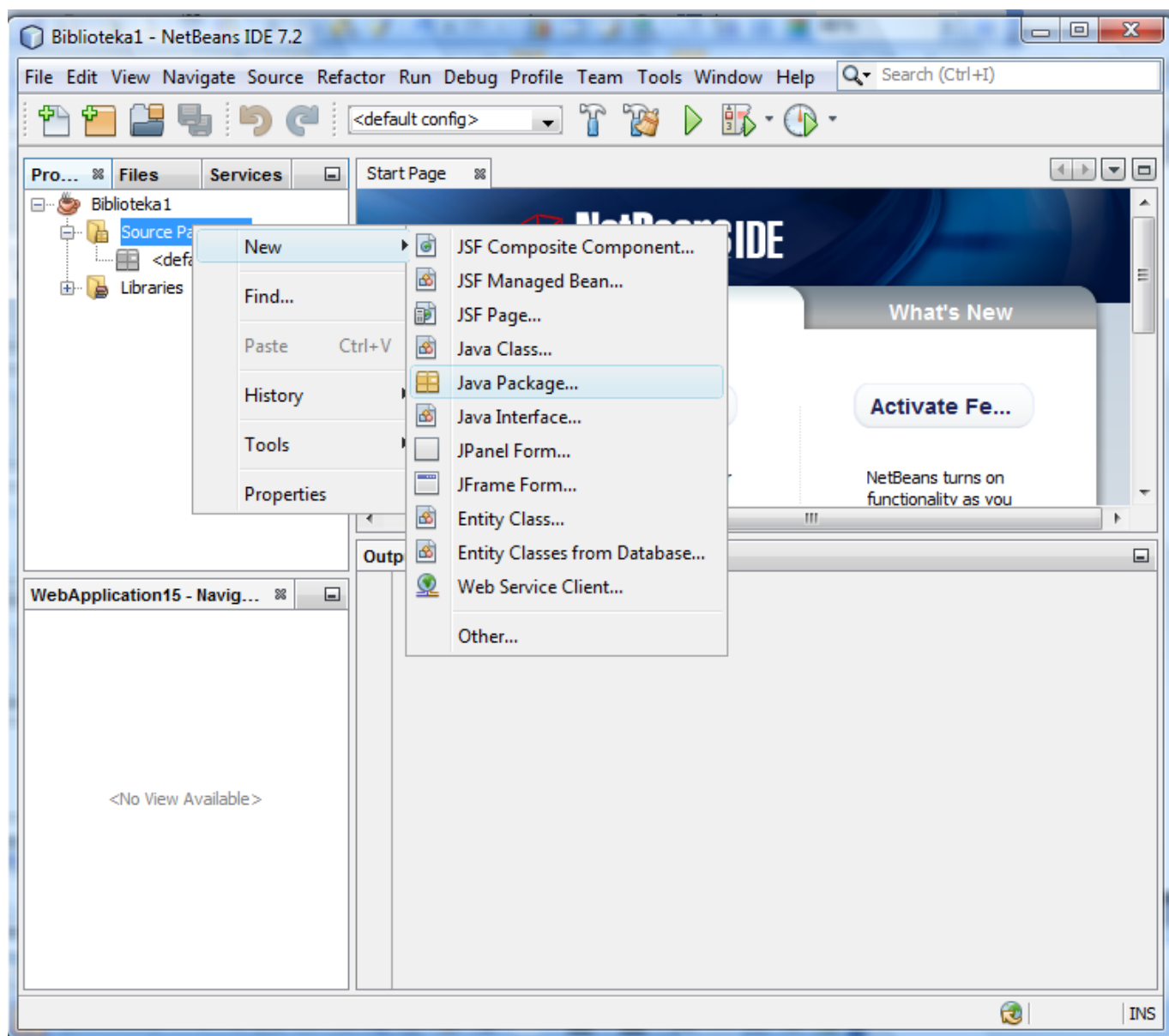
Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

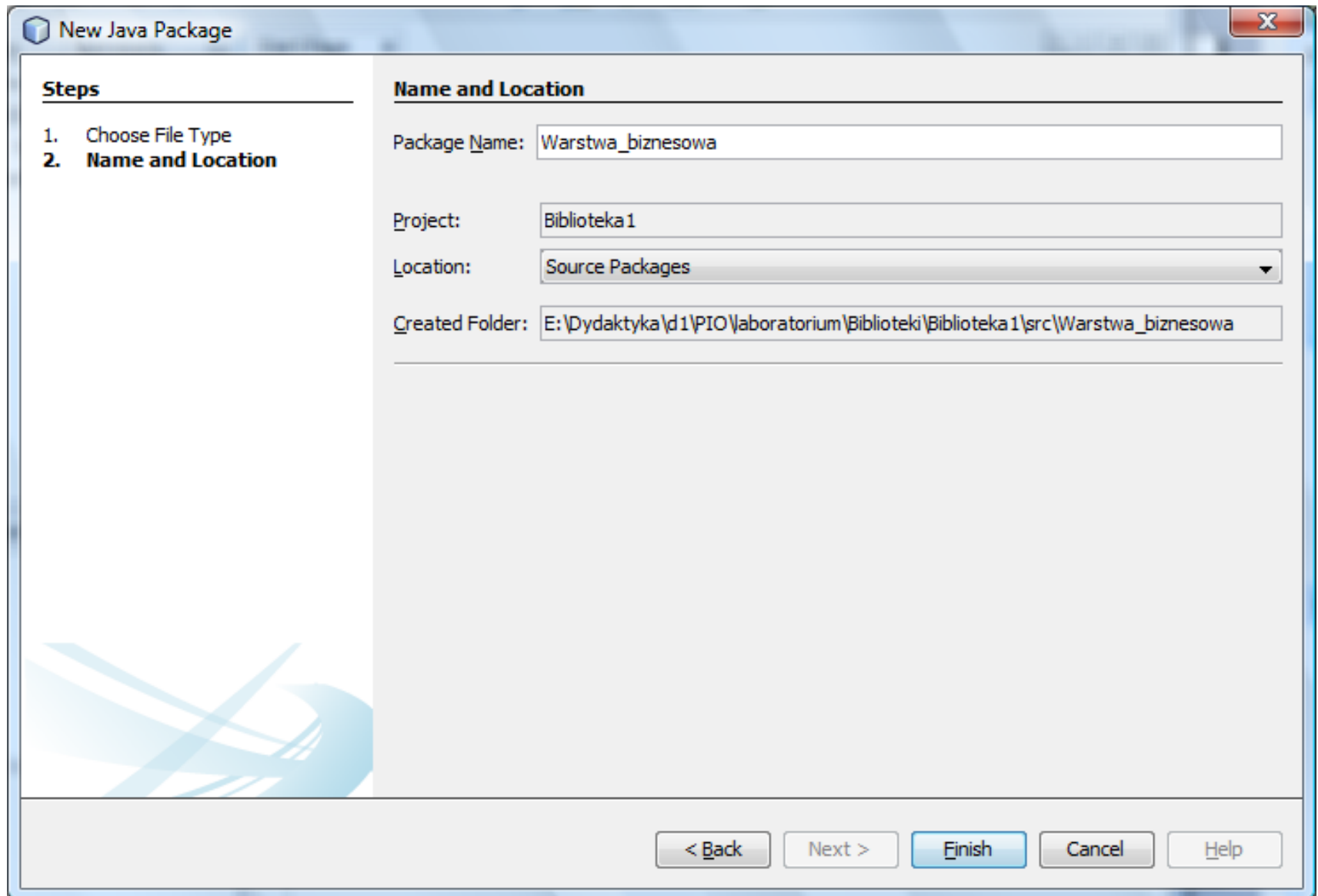
Create Main Class

< Back Next > Finish Cancel Help

11.4. Wstawienie nowego pakietu do projektu – prawym klawiszem należy kliknąć na pozycję *Source Package* w okienku *Projects* i wybrać z listy pozycję *Java Package* (lub *Other*, jeśli nie ma takiej pozycji na liście)



11.5. Wstawienie nowego pakietu do projektu – nadanie nazwy pakietowi **Warstwa_biznesowa** w polu *Package Name*



New Java Package

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Package Name:

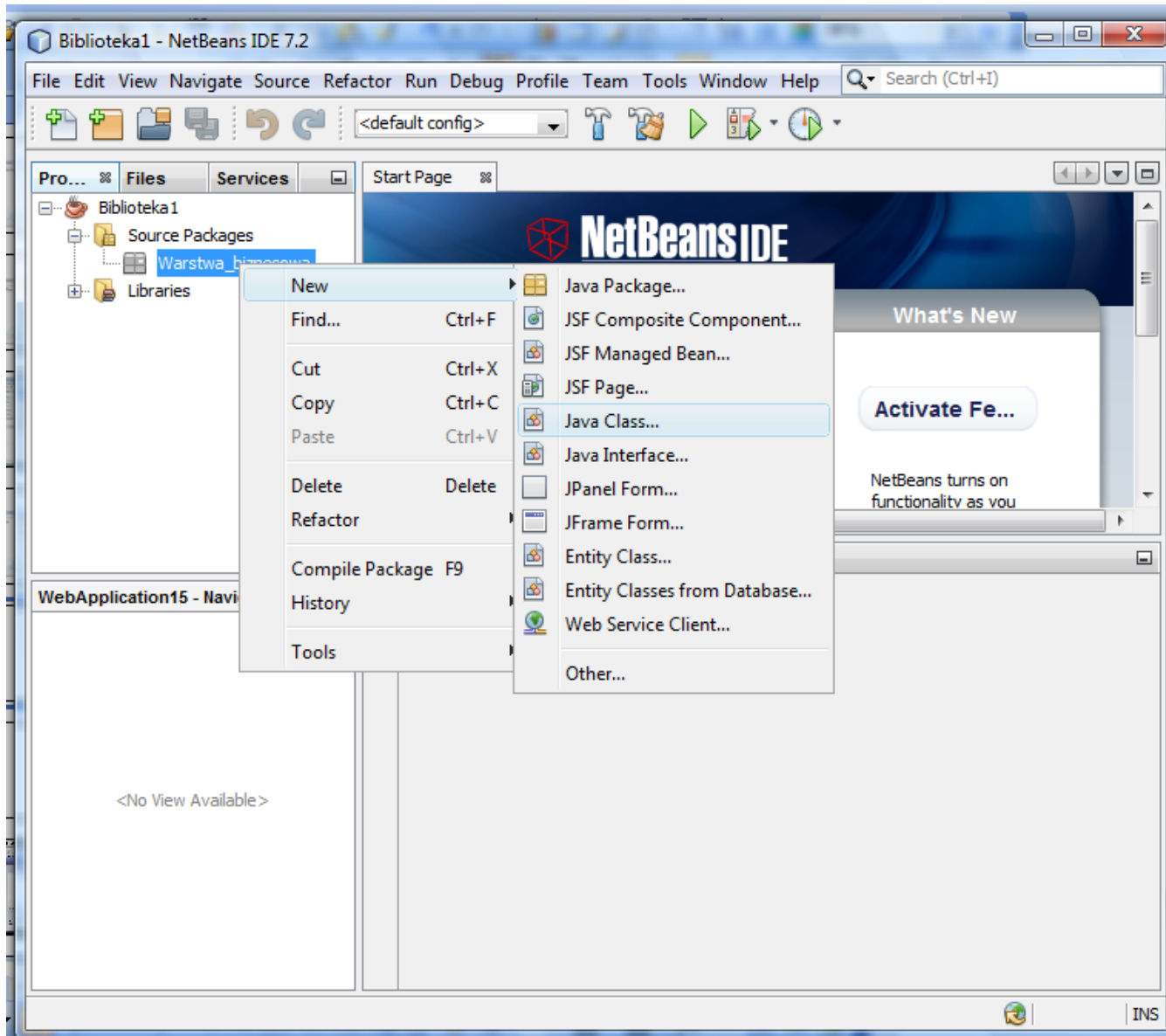
Project:

Location:

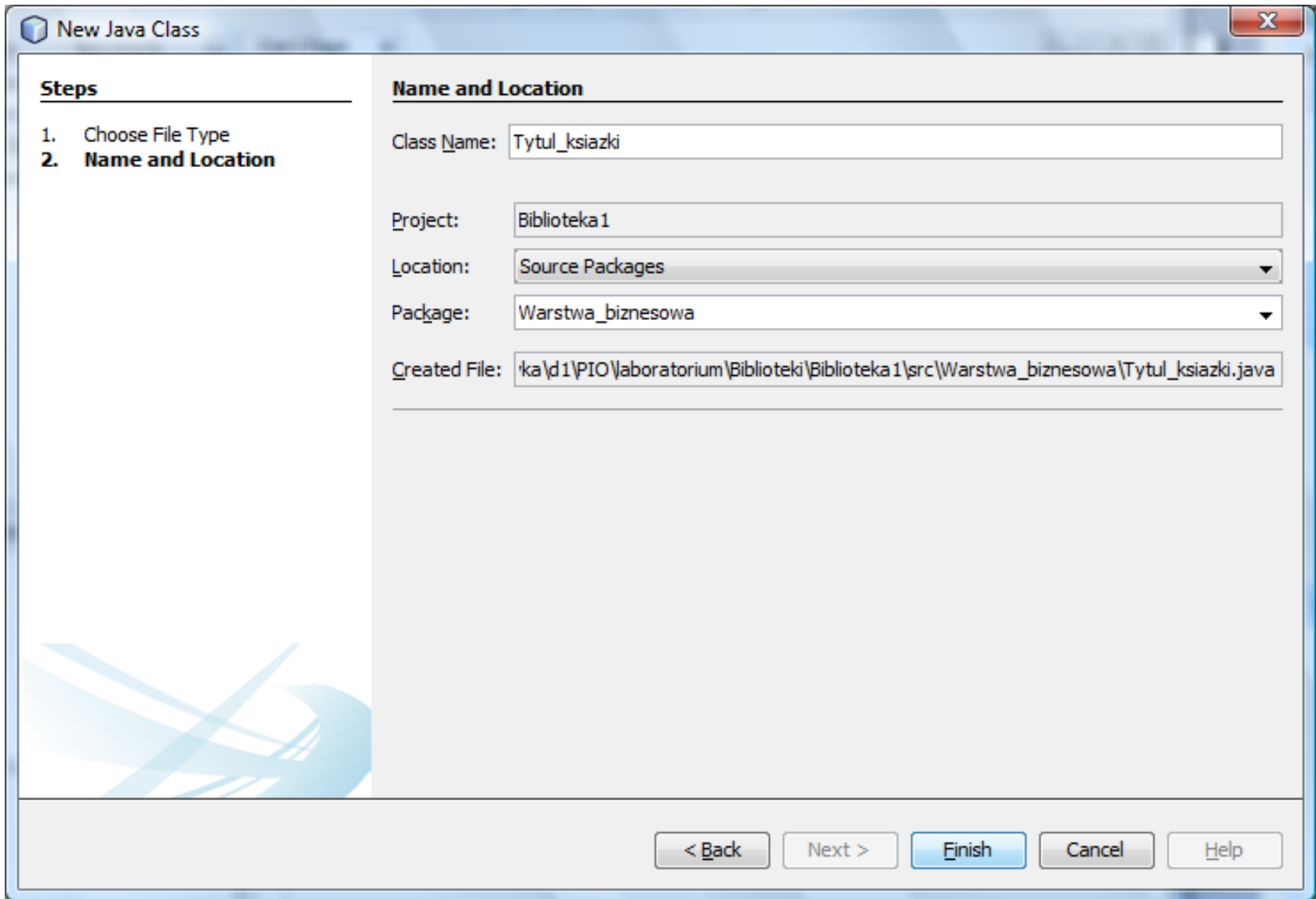
Created Folder:

< Back Next > **Finish** Cancel Help

11.6. Wstawienie do pakietu **Warstwa_biznesowa** nowej klasy – należy kliknąć prawym klawiszem myszy na nazwę pakietu i wybrać z listy pozycję *Java Class* (lub *Other*, jeśli nie ma takiej pozycji na liście)



11.7. Nadanie nazwy nowej klasie **Tytul_książki** w polu *Class Name*



New Java Class

Steps

1. Choose File Type
- 2. Name and Location**

Name and Location

Class Name: Tytul_książki

Project: Biblioteka1

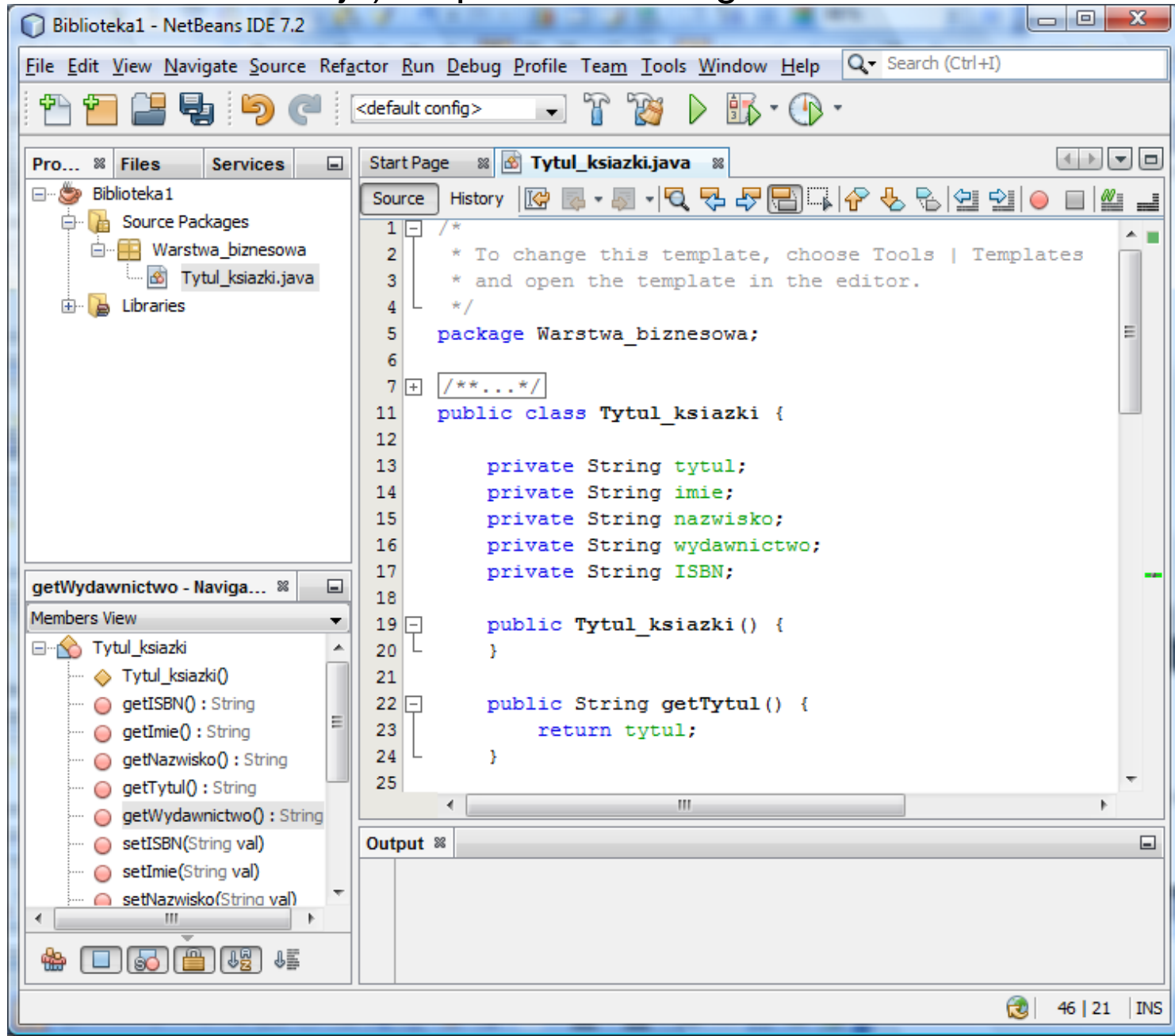
Location: Source Packages

Package: Warstwa_biznesowa

Created File: 'ka\d1\PIO\laboratorium\Biblioteki\Biblioteka1\src\Warstwa_biznesowa\Tytul_książki.java

< Back Next > **Finish** Cancel Help

11.8. Zdefiniowanie kodu klasy **Tytul_ksiazki** (kod klasy zawiera następnny slajd) na podstawie diagramu klas



The screenshot displays the NetBeans IDE 7.2 interface. The main editor window shows the source code for the `Tytul_ksiazki.java` file. The code is as follows:

```
1  /*
2   * To change this template, choose Tools | Templates
3   * and open the template in the editor.
4   */
5  package Warstwa_biznesowa;
6
7  /**...*/
11 public class Tytul_ksiazki {
12
13     private String tytul;
14     private String imie;
15     private String nazwisko;
16     private String wydawnictwo;
17     private String ISBN;
18
19     public Tytul_ksiazki() {
20     }
21
22     public String getTytul() {
23         return tytul;
24     }
25 }
```

The left sidebar shows the project structure for "Biblioteka1", including "Source Packages", "Warstwa_biznesowa", and "Tytul_ksiazki.java". The "Members View" for the `Tytul_ksiazki` class is also visible, listing methods such as `getISBN() : String`, `getImie() : String`, `getNazwisko() : String`, `getTytul() : String`, `getWydawnictwo() : String`, `setISBN(String val)`, `setImie(String val)`, and `setNazwisko(String val)`.

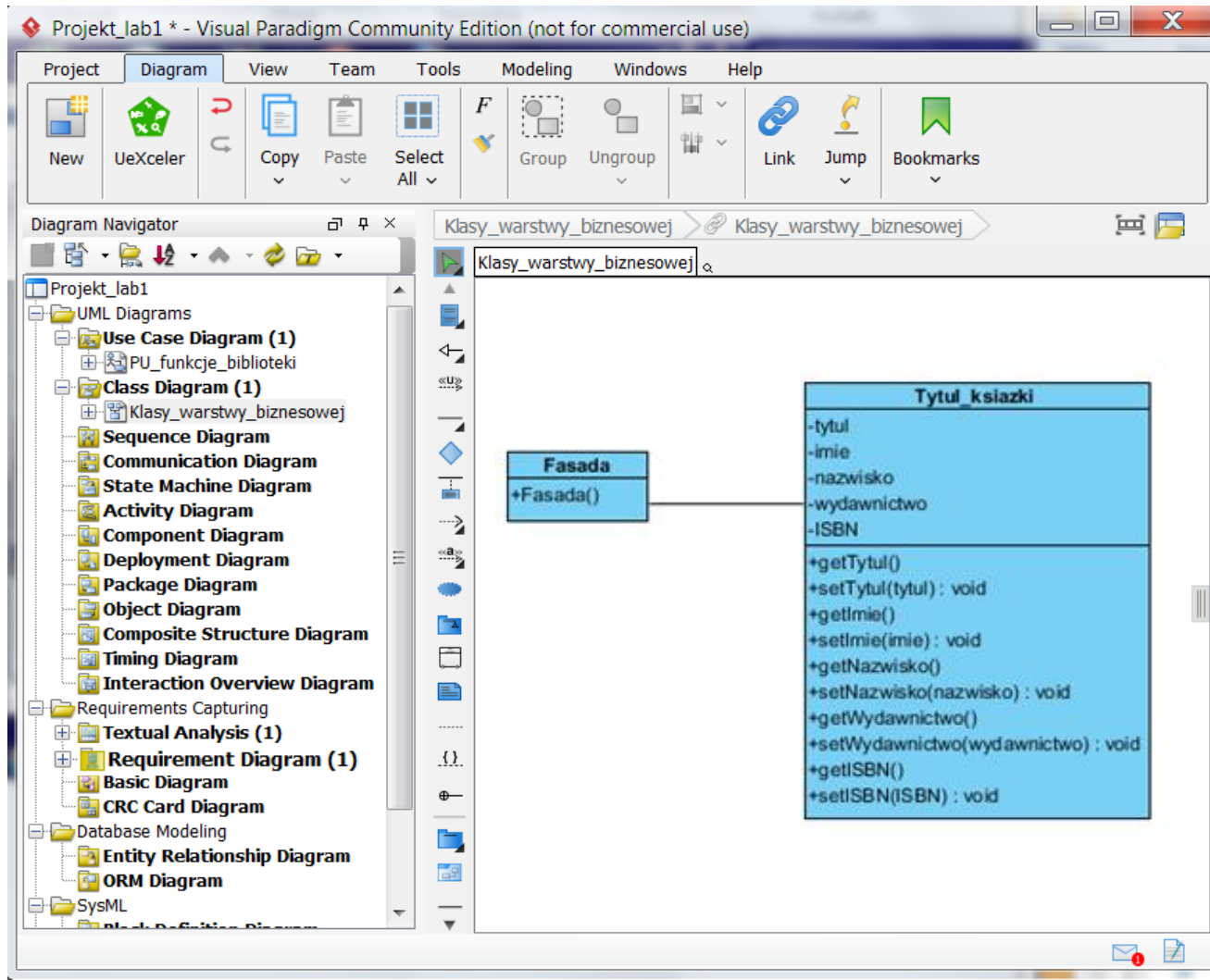
The bottom right corner of the IDE shows the "Output" window and the system tray with the text "46 | 21 | INS".

11.9. Kod klasy Tytul_ksiazki

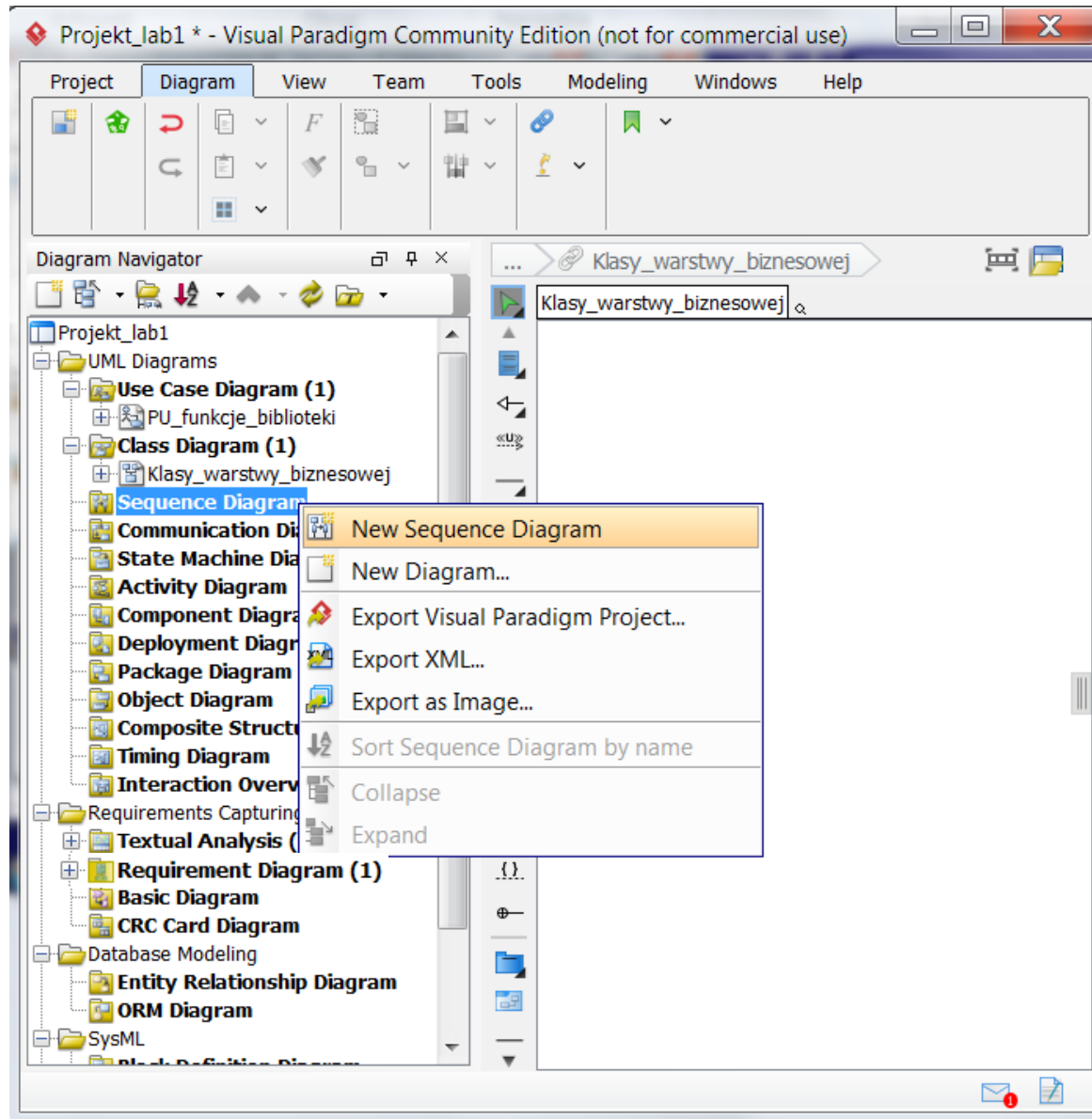
```
public class Tytul_ksiazki {  
    private String tytul;  
    private String imie;  
    private String nazwisko;  
    private String wydawnictwo;  
    private String ISBN;  
  
    public String getTytul()  
    public void setTytul(String val)           { return tytul; }  
    public String getImie()                   { this.tytul = val; }  
    public void setImie(String val)           { return imie; }  
    public String getNazwisko()                { this.imie = val; }  
    public void setNazwisko(String val)       { return nazwisko; }  
    public String getWydawnictwo()            { this.nazwisko = val; }  
    public void setWydawnictwo(String val)   { return wydawnictwo; }  
    public String getISBN()                   { this.wydawnictwo = val; }  
    public void setISBN(String val)          { return ISBN; }  
    public void setISBN(String val)          { this.ISBN = val; }  
}
```

Relacja jeden do jeden

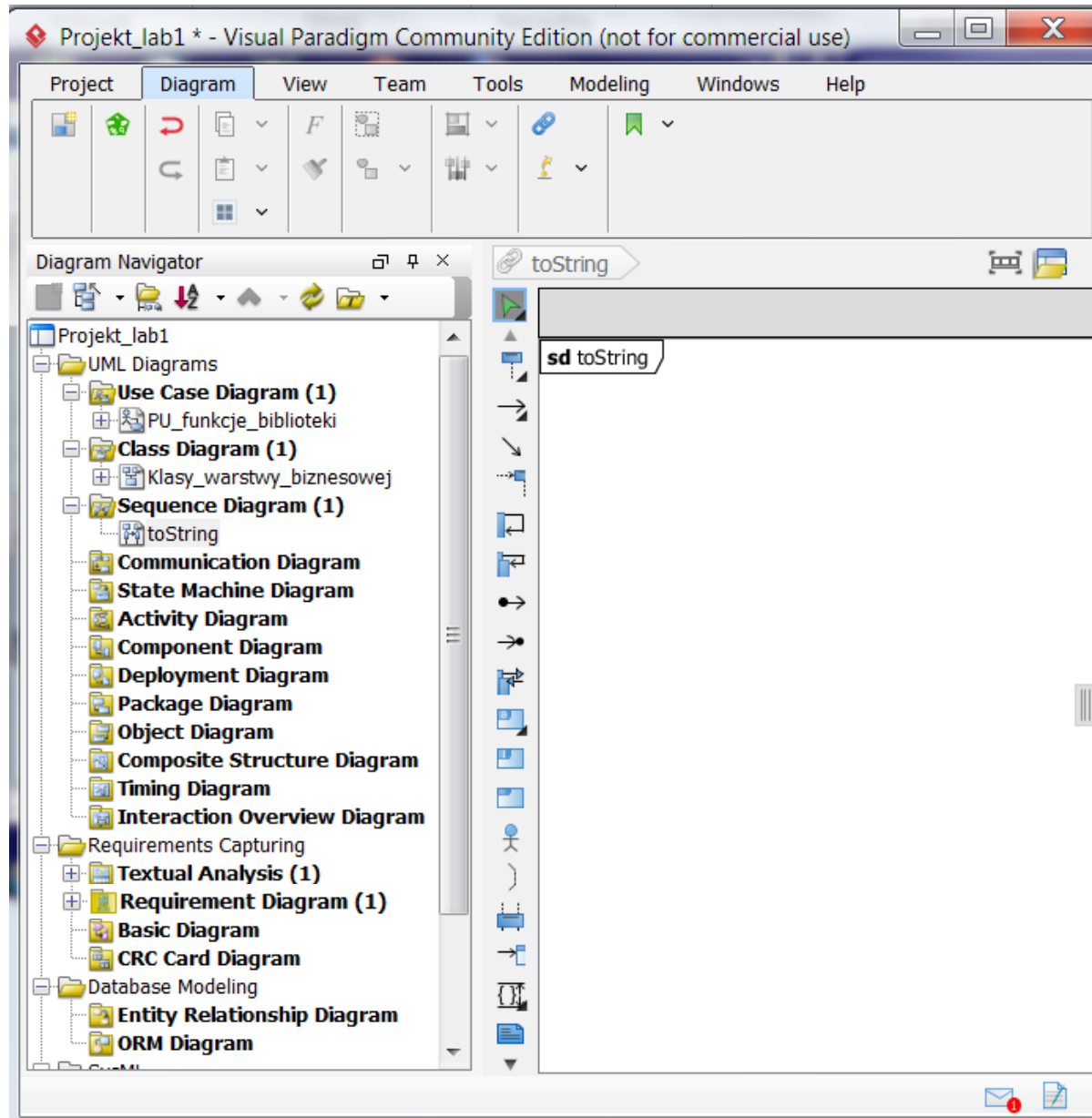
12/12.1 Wstawienie nowej klasy **Fasada** (podobnie jak klasę Tytuł_książki) powiązanej za pomocą relacji *Association* 1..0 z klasą typu **Tytuł_książki** – relację należy wybrać z palety z lewej strony lewym klawiszem myszy oraz położyć ją na klasie **Fasada** i przeciągnąć na klasę **Tytuł_książki**. Klasa ta reprezentuje wzorzec projektowy Fasada - będzie zastosowana do obsługi wywołań przypadków użycia przez warstwę interfejsu graficznego użytkownika.



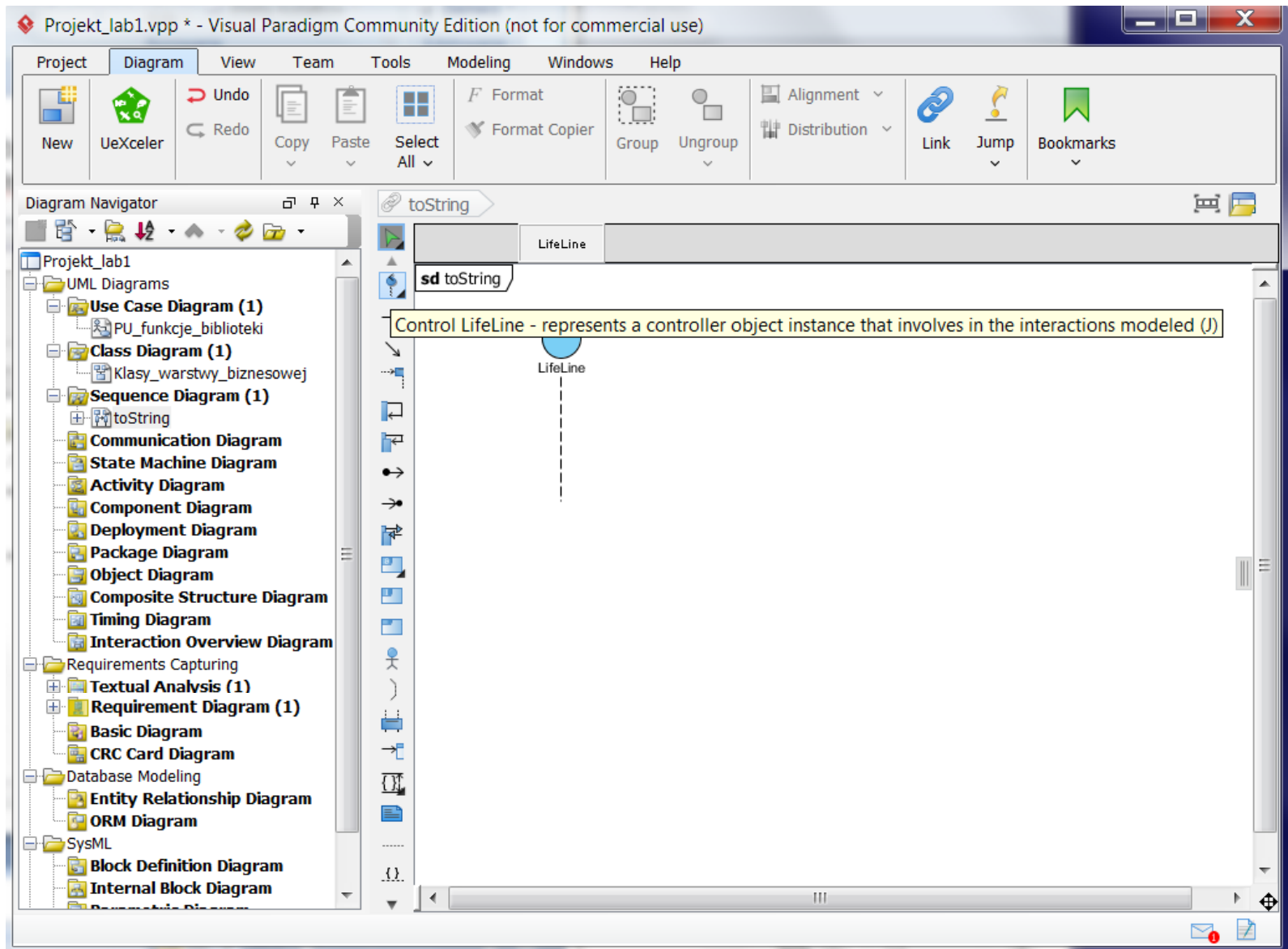
12.2. Wstawianie diagramu sekwencji – należy kliknąć prawym klawiszem myszy na nazwę modelu w okienku Model Explorer i wybrać z listy opcję *Diagram/UML Diagram/Sequence Diagram*



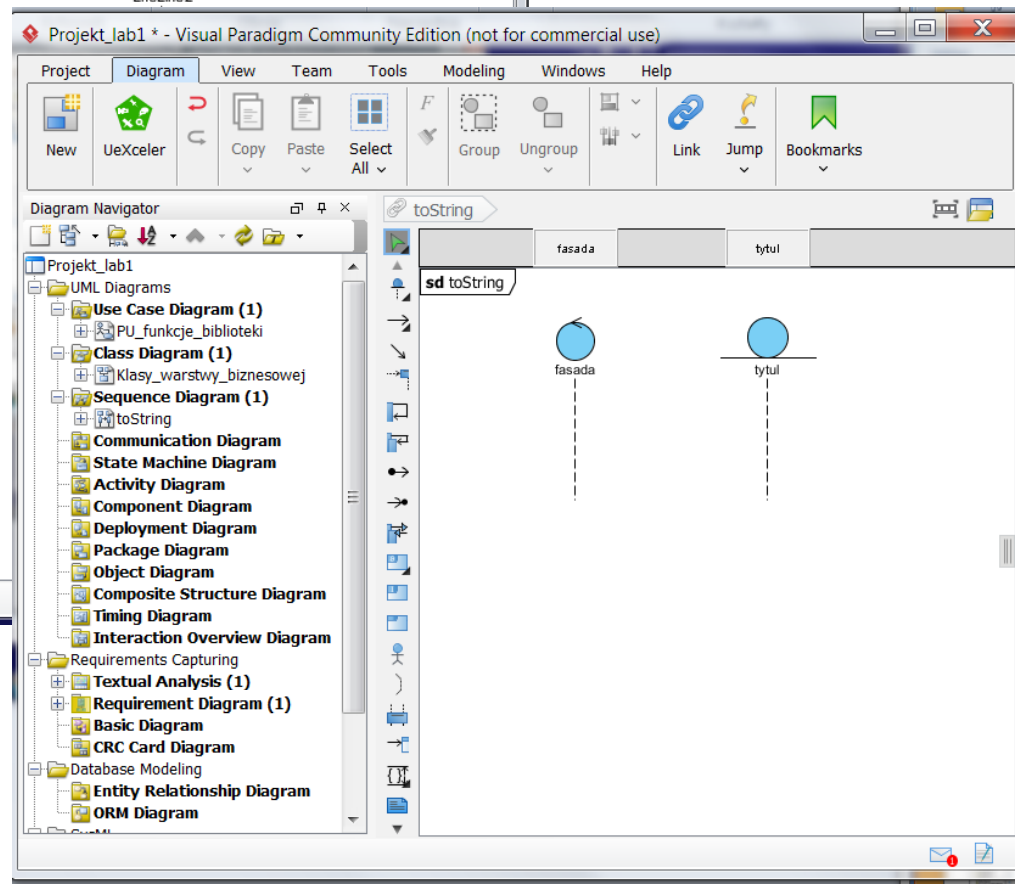
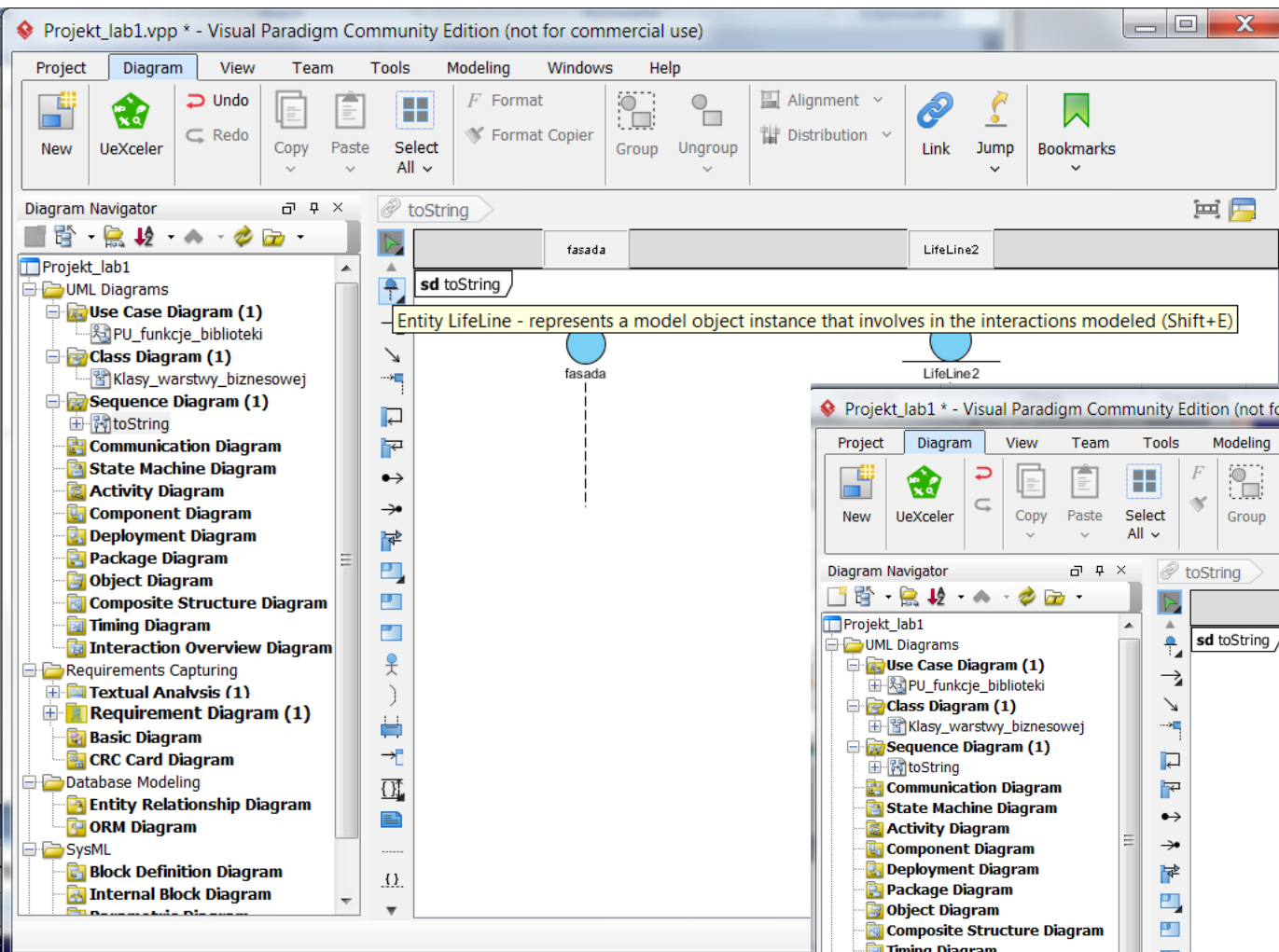
12.3. Należy nadać nazwę **toString** diagramowi sekwencji – diagram będzie zawierał definicję metody **toString** w klasie **Tytuł_książki**



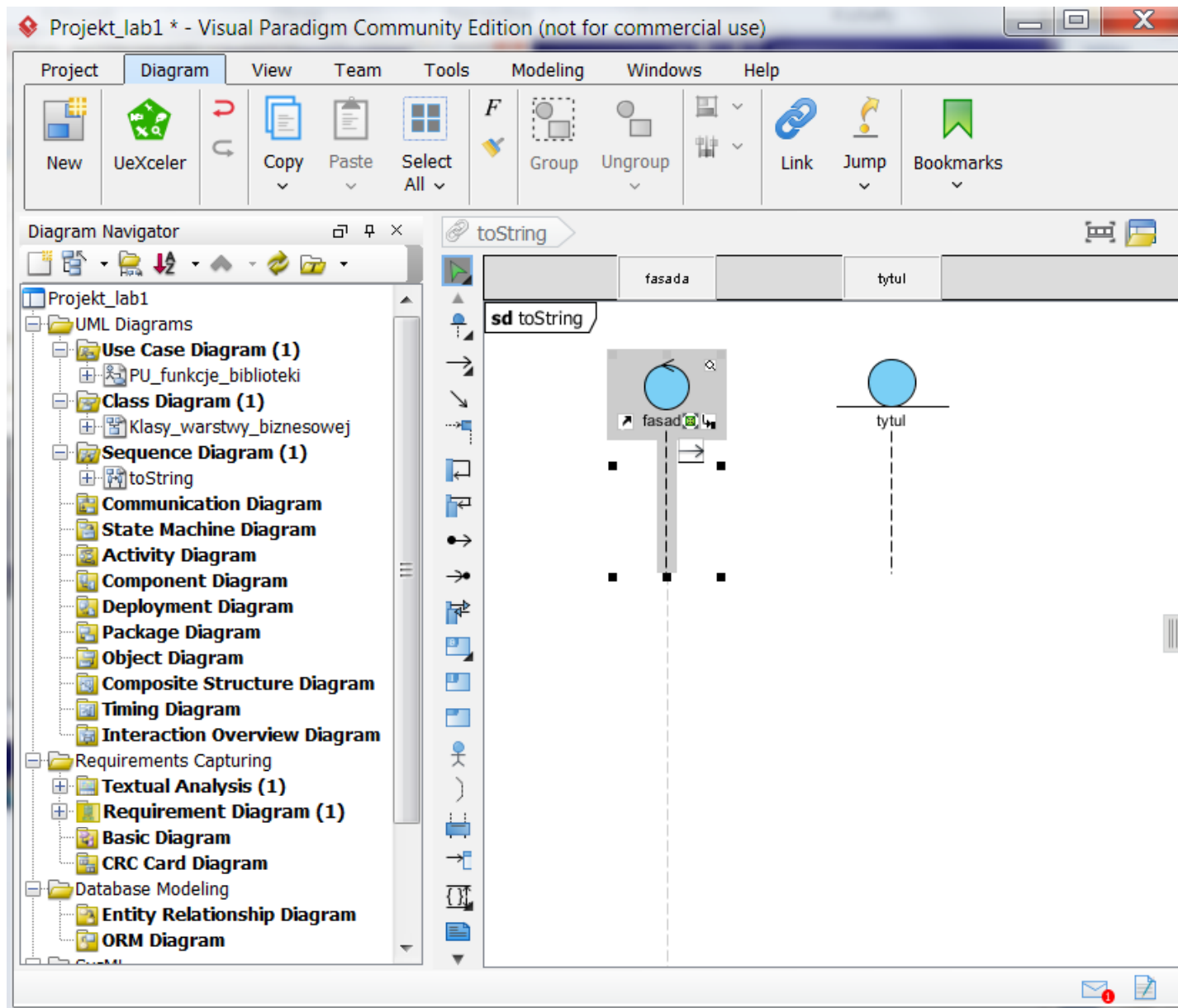
12.4. Należy z palety z lewej strony wybrać z listy *LifeLine* linię życia typu *Control LifeLine*



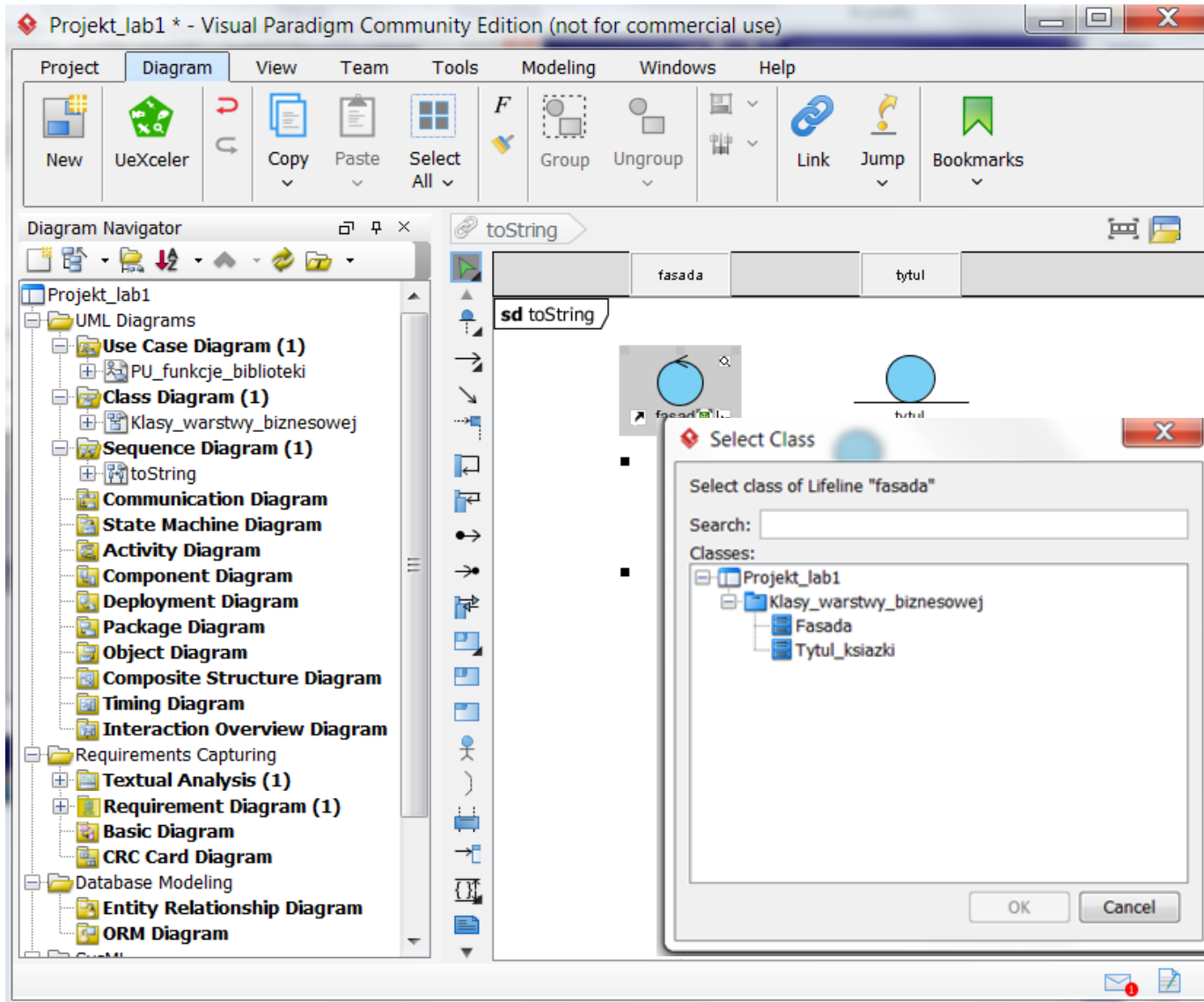
12.5. Należy z palety z lewej strony wybrać z listy Lifeline linię życia typu *Entity* Lifeline i nadać nazwę tytuł



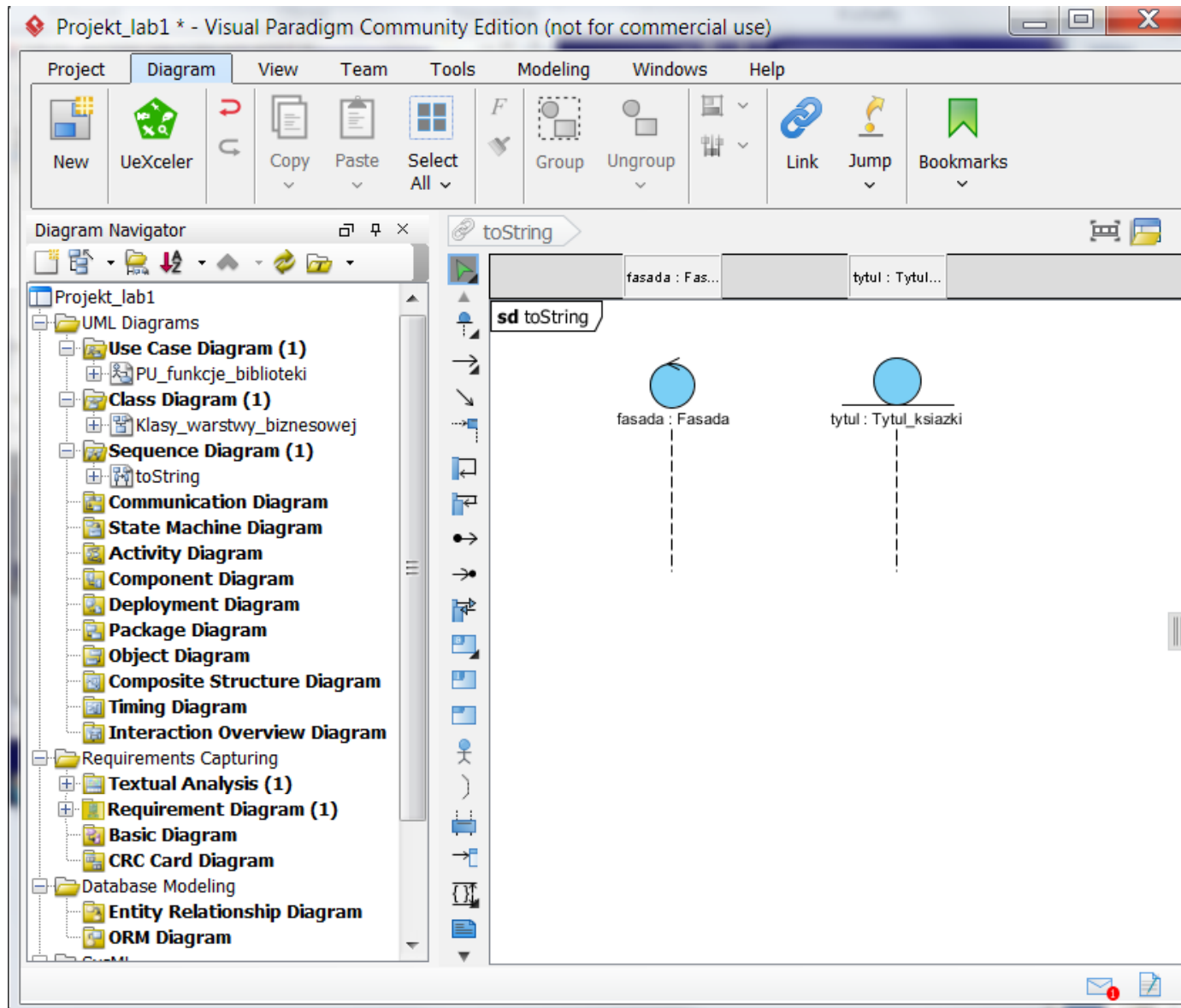
12.6. Należy linie życia obiektów powiązać z klasami z diagramu klas – po wybraniu linii życia **fasada** należy kliknąć prawym klawiszem myszy i wybrać z listy opcję *Select Class/Select Class*



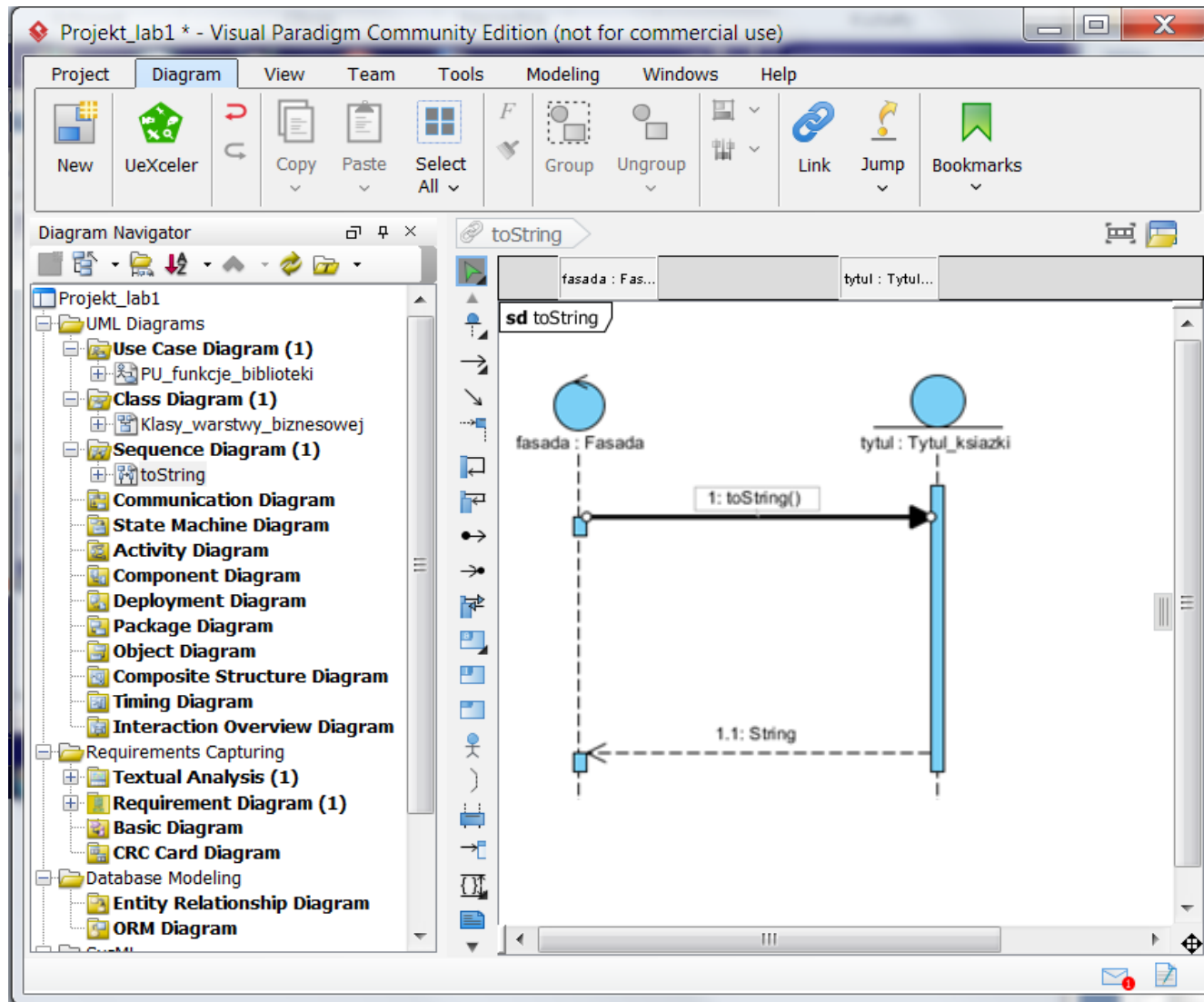
12.7. W formularzu *Select Class* należy w polu *Search* wpisać fragment nazwy klasy **Fasada**. W ukazanym okienku wybrać właściwą klasę i nacisnąć klawisz *OK*. Podobnie należy powiązać linię życia **tytul**.



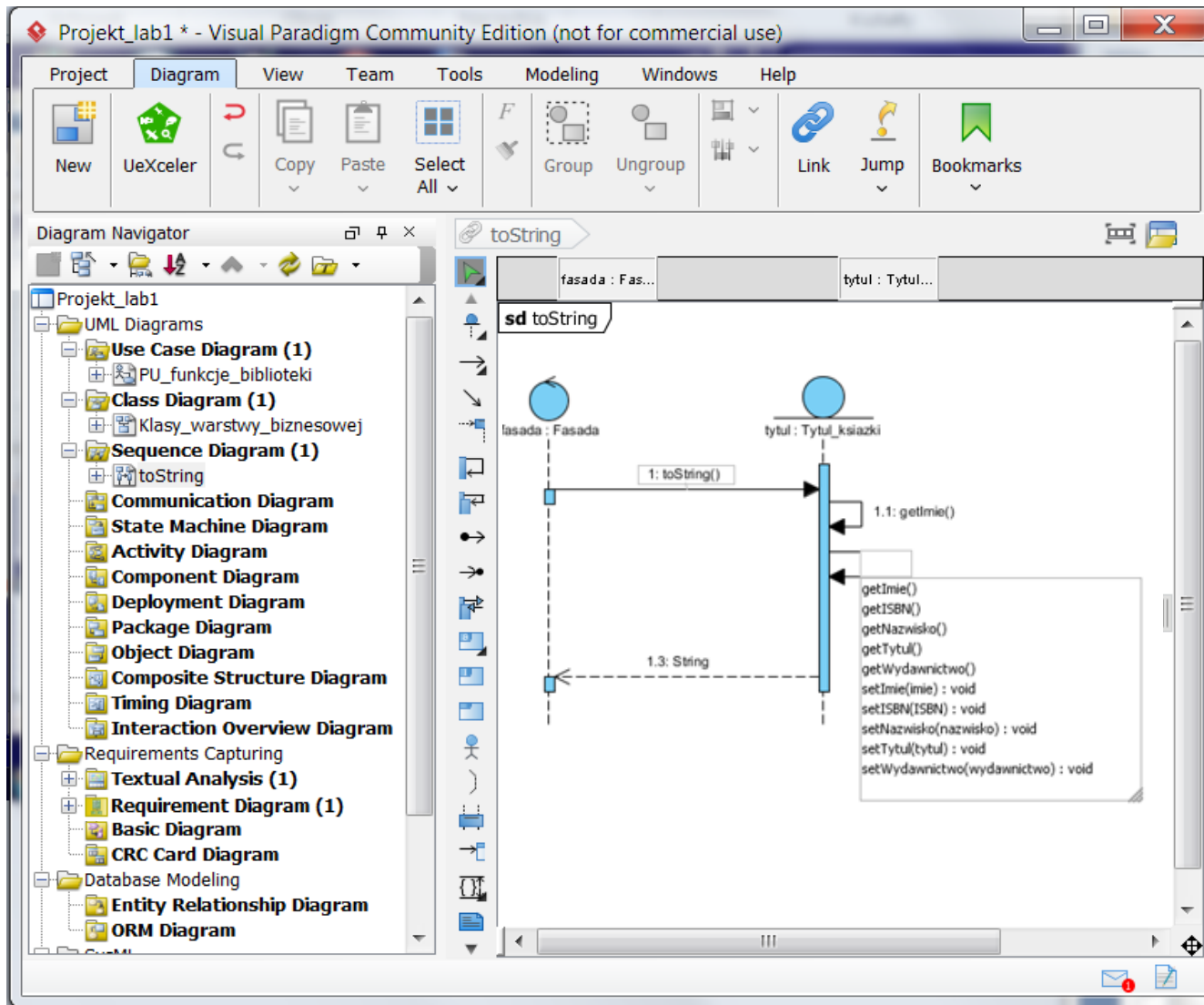
12.8. Należy wybrać z listy *Message* typ metody *Call Message* i przeciągnąć ją kładąc na linii życia **fasada** i przeciągając położyć na linii życia **tytul**. Podobnie należy zrobić z wiadomością typu *Return Message*.



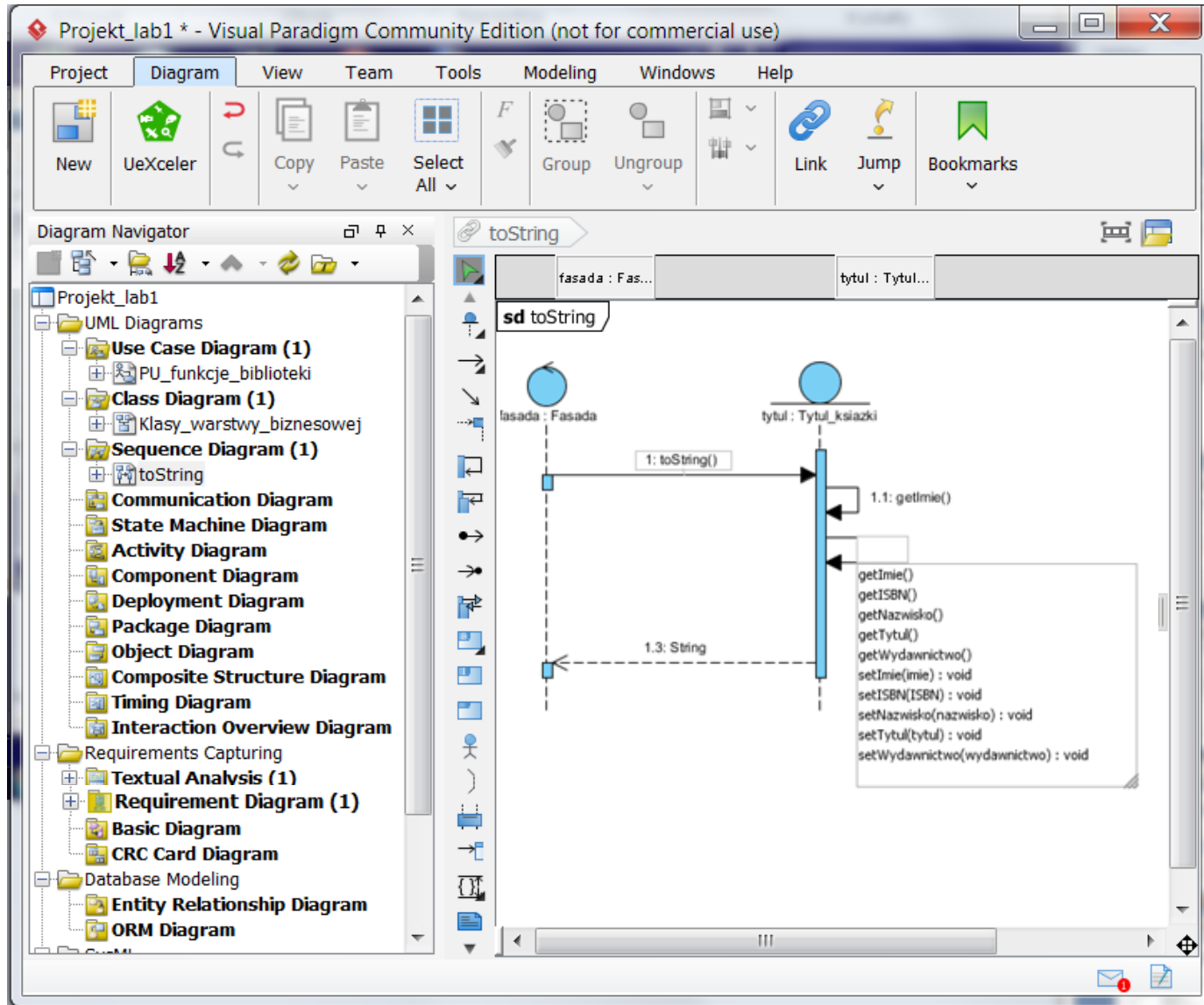
12.9. Należy wybrać z listy *Message* typ metody *Call Message* i przeciągnąć ją kładąc na linii życia **fasada** i przeciągając położyć na linii życia **tytul**. Podobnie należy zrobić z wiadomością typu *Return Message*, która powinna wychodzić z linii życia **tytul** i wchodzić do linii życia **fasada**.



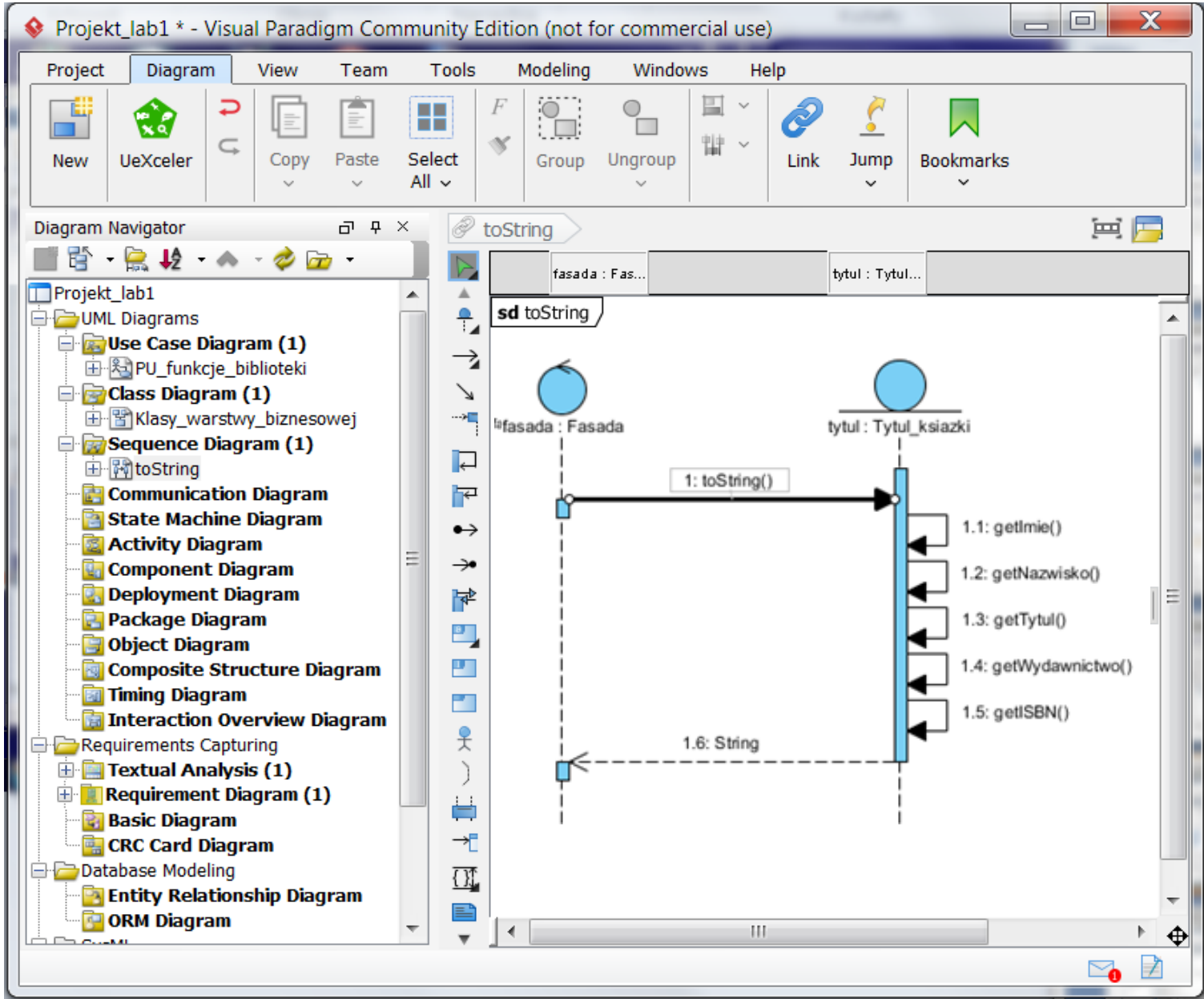
12.10. Należy zdefiniować ciało metody **toString** w klasie **Tytul_książki** za pomocą wiadomości *Self Message*, przeciąganych z palety z lewej strony. Podczas wstawiania wiadomości pokazuje się lista metod klasy **Tytul_książki** zdefiniowanych podczas tworzenia diagramu klas. Należy dokonać wyboru właściwej metody typu `get` z listy metod klasy **Tytul_książki**.



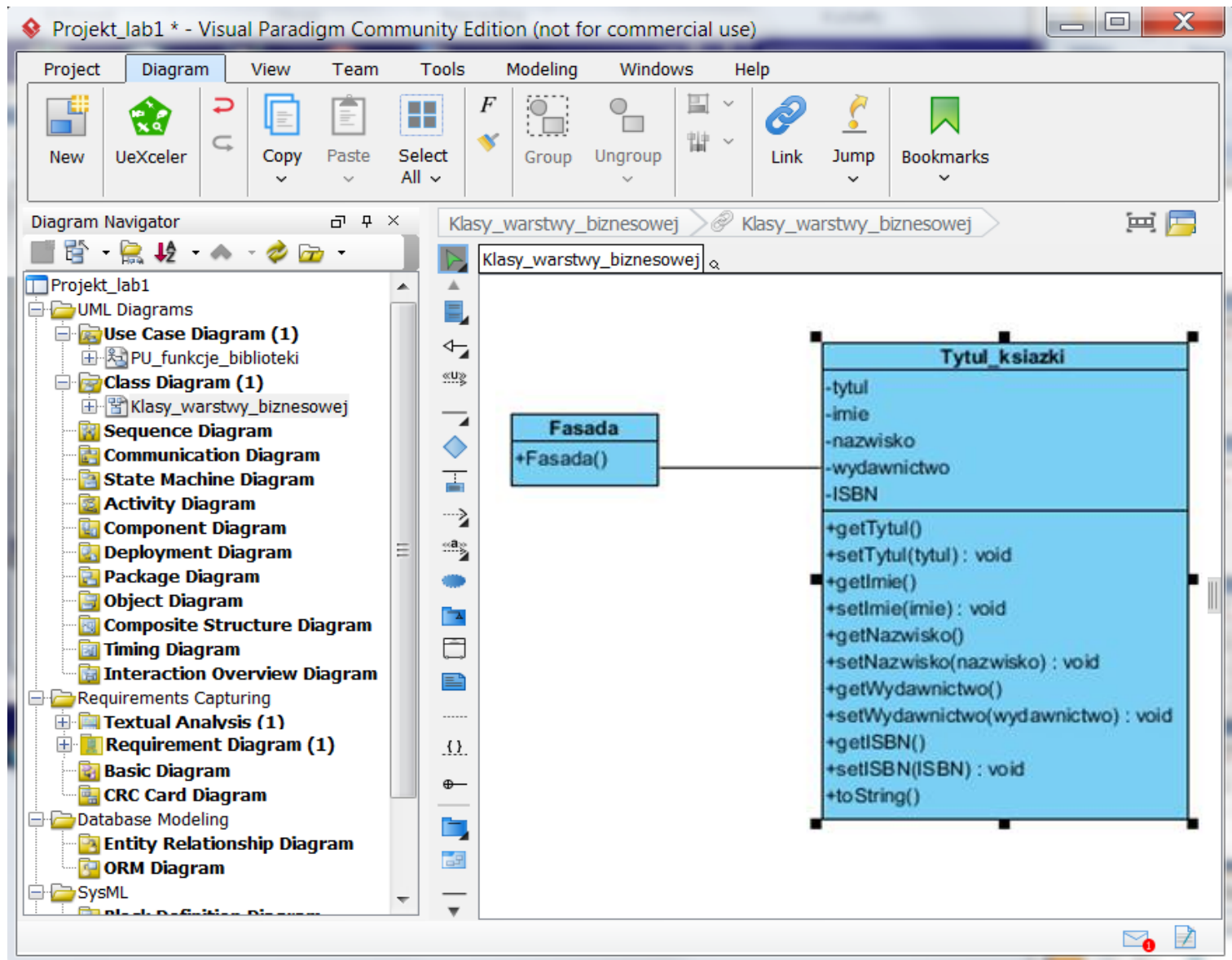
12.11. Włączenie zdefiniowanej metody do klasy **Tytuł_książki** – po wybraniu wiadomości o nazwie **toString** klikając prawym klawiszem myszy należy wybrać z listy pozycje *Select Operation/Select Operation „toString()”*



12.12. Rezultat wykonania scenariusza metody *toString* w klasie **Tytuł_książki**.



12.13. Metoda `toString` zdefiniowana na diagramie sekwencji pojawiła się w klasie `Tytul_książki`



12.14. Definicja kodu metody `toString` w klasie `Tytul_książki` zdefiniowana na podstawie diagramu sekwencji (p. 11.12)

The screenshot displays the NetBeans IDE 7.2 interface. The main editor window shows the source code for the `Tytul_książki.java` class. The `toString` method is highlighted, showing its implementation. The method concatenates the title, author, ISBN, and publisher information into a single string.

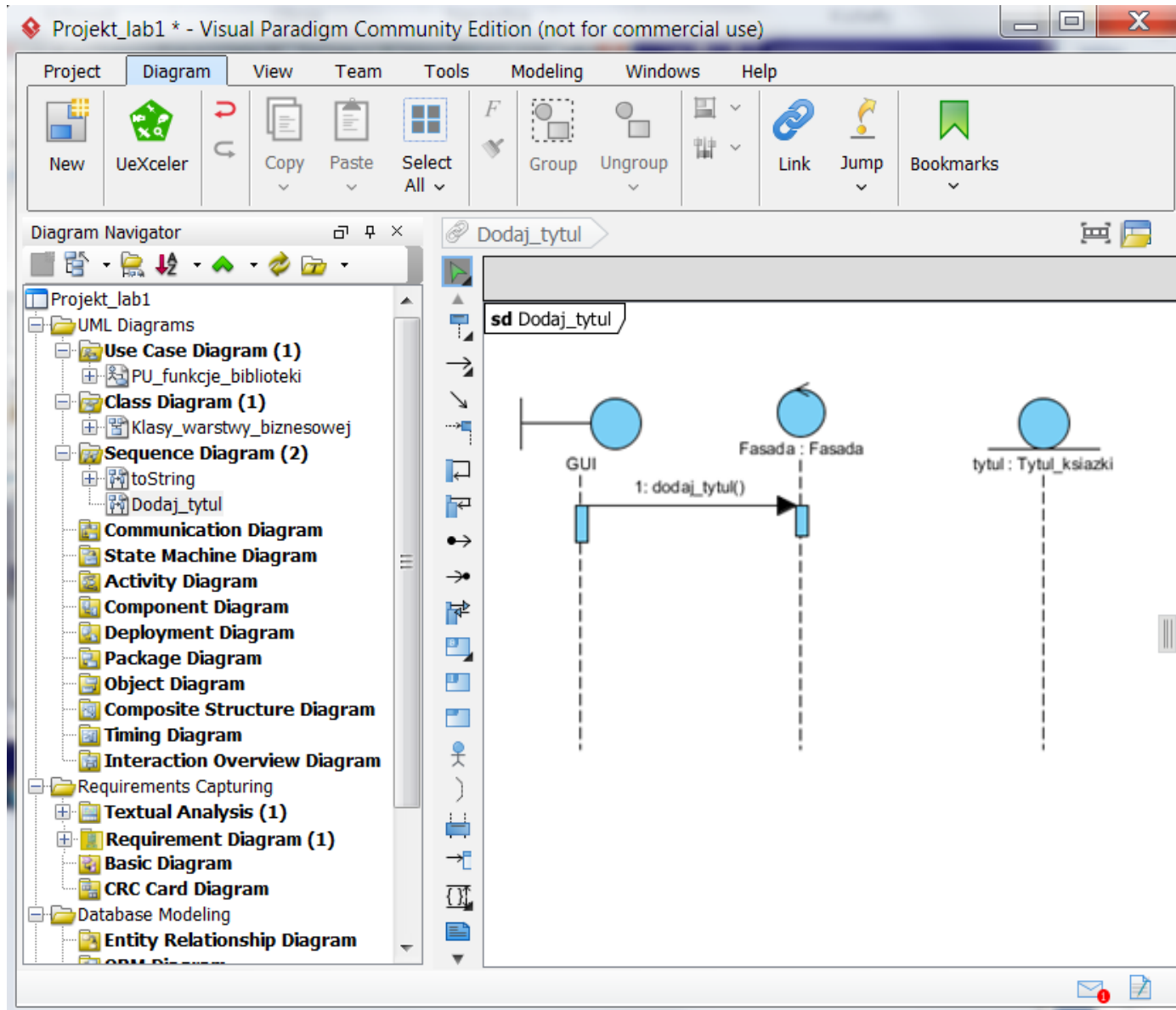
```
54 public String getISBN() {
55     return ISBN;
56 }
57
58 public void setISBN(String val) {
59     this.ISBN = val;
60 }
61
62 @Override
63 public String toString()
64 {
65     String pom = "Tytuł: " + getTytuł();
66     pom += " Autor:" + getNazwisko() + " " + getImię();
67     pom += " ISBN: " + getISBN();
68     pom += " Wydawnictwo:" + getWydawnictwo();
69     return pom;
70 }
71 }
72 }
73 }
```

The `Members View` on the left shows the following methods:

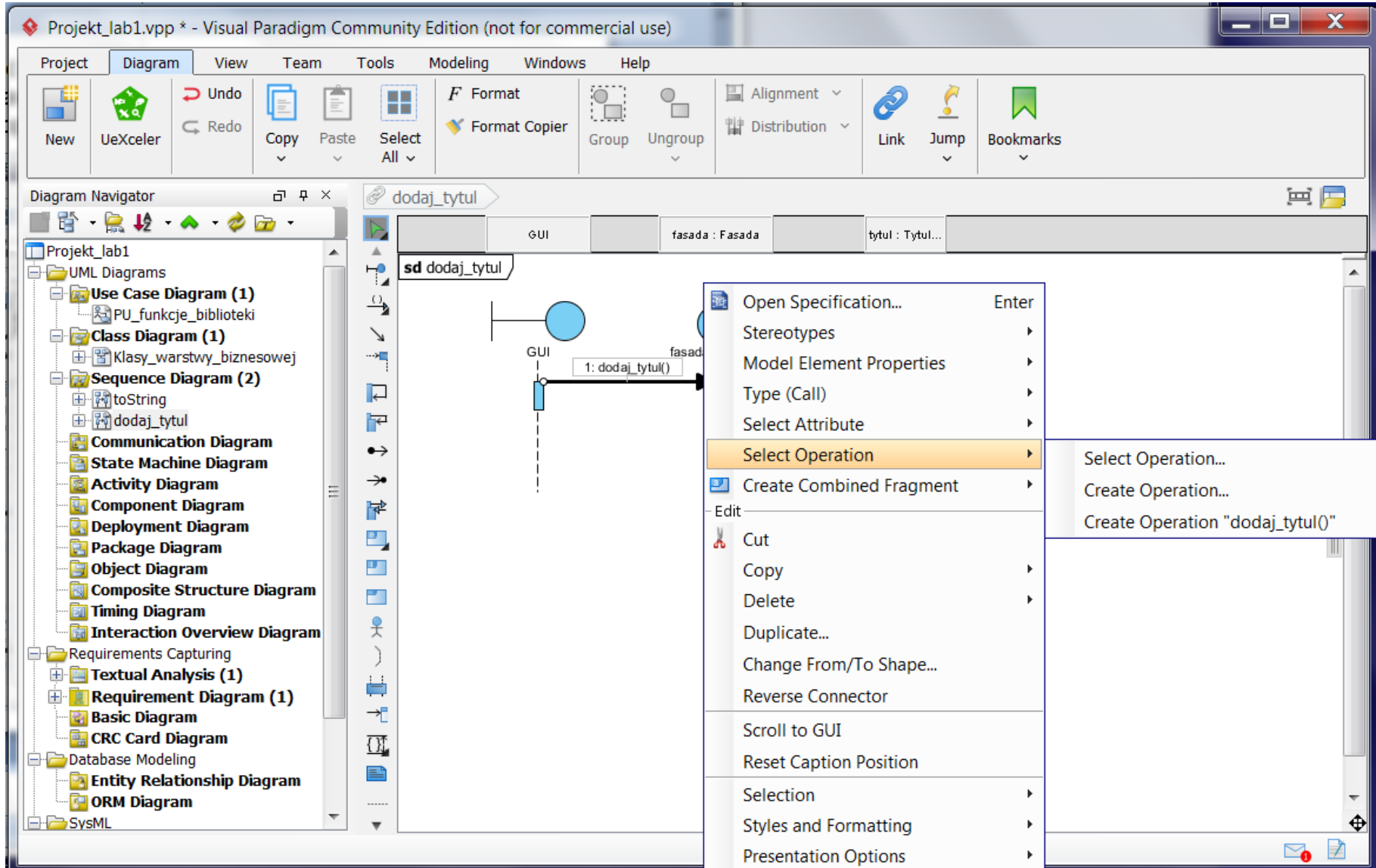
- `getWydawnictwo(): String`
- `setISBN(String val)`
- `setImię(String val)`
- `setNazwisko(String val)`
- `setTytuł(String val)`
- `setWydawnictwo(String val)`
- `toString(): String`

The status bar at the bottom right shows the page number 63 | 12 and the text INS.

12.15. Dodanie nowego diagramu sekwencji o nazwie **Dodaj_tytul** reprezentującego definicję przypadku użycia **Dodaj_tytul**.



12.16. Należy wiadomość **Dodaj_tytul** z klasy przypisać do klasy **Fasada** – należy kliknąć prawym klawiszem myszy na strzałkę reprezentującą metodę i wybrać z listy opcje *Select Operation/Create Operation*



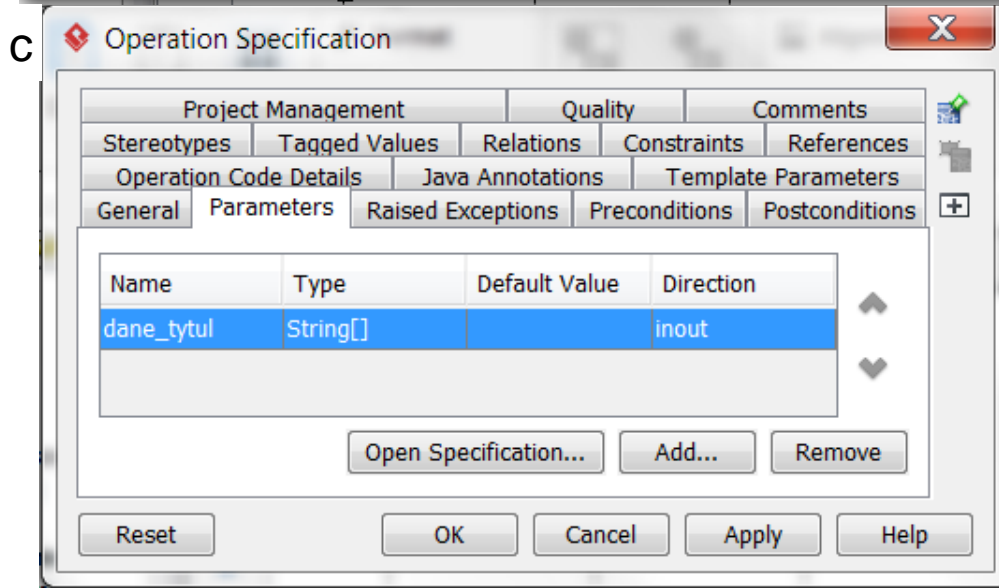
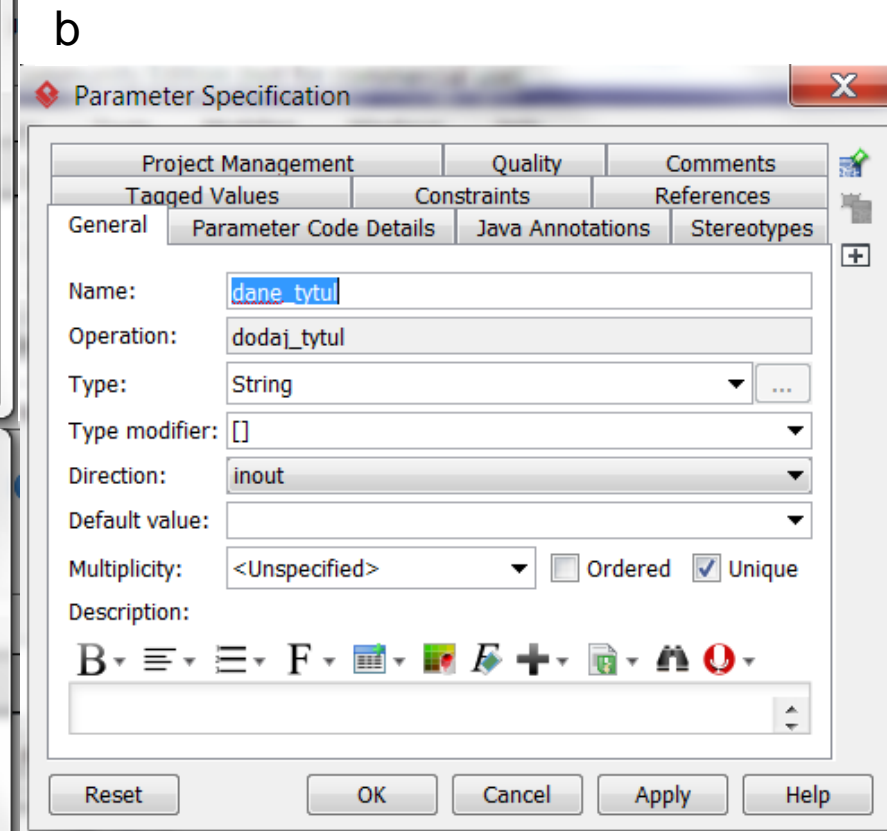
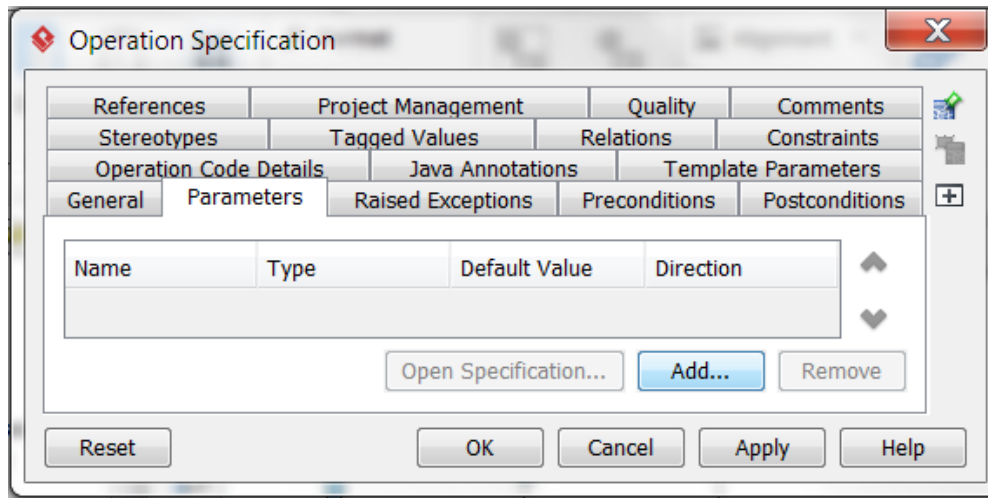
12.17. Na formularzu *Operation Specification* należy wypełnić następujące pola: *Name* (nazwa metody), *Classifier* (Nazwę klasy), *Return type* (wynik zwracany przez metodę), *Visibility* (określanie zasięgu metody). Następnie należy zdefiniować listę parametrów metody wybierając kolejny formularz w zakładce *Parameters*

The screenshot shows the 'Operation Specification' dialog box with the following fields and values:

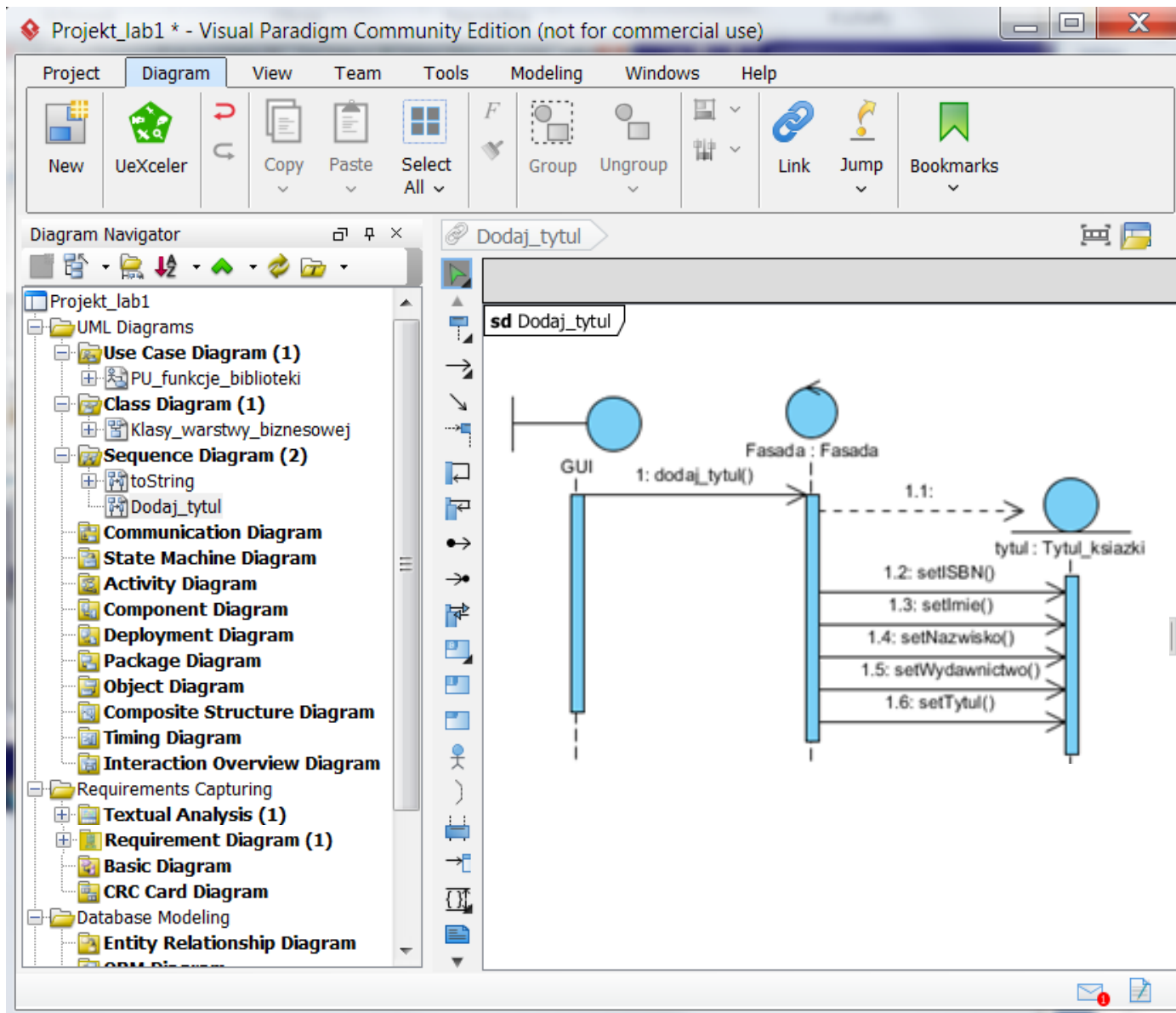
- Name:** `dodaj_tytul`
- Classifier:** `Klasy_warstwy_biznesowej.Fasada`
- Return type:** `void`
- Type modifier:** `<Unspecified>`
- Visibility:** `public`
- Scope:** `instance`
- Lower:** (empty)
- Upper:** (empty)
- Body condition:** (empty)
- Description:** (empty)
- Options:** Abstract, Leaf, Query, Ordered, Unique

Buttons at the bottom: Reset, OK, Cancel, Apply, Help.

12.18. Zdefiniowanie parametru metody **Dodaj_tytul** w oknie *Parameters*: w okienku *a* należy kliknąć klawisz *Add*. W formularzu *b* w polu *Name* należy wpisać nazwę parametru **dane_tytul**, w polu *Operation* nazwę metody, w polu *Type* typ parametru, w polu *Type modifier* uzupełnienie definicji typu parametru. Po kliknięciu na klawisz *OK* następuje powrót do okna *Parameters*, gdzie uwidoczny jest nowy parametr



12.19. Pełna definicja metody **dodaj_tytul** w klasie **Fasada** - linia życia **tytul** typu **Tytul_ksiazki** została zdefiniowana jako *Create Message* przez przeciągnięcie na linię życia typu *Entity Lifeline* ikony *Create Message* z palety z lewej strony.

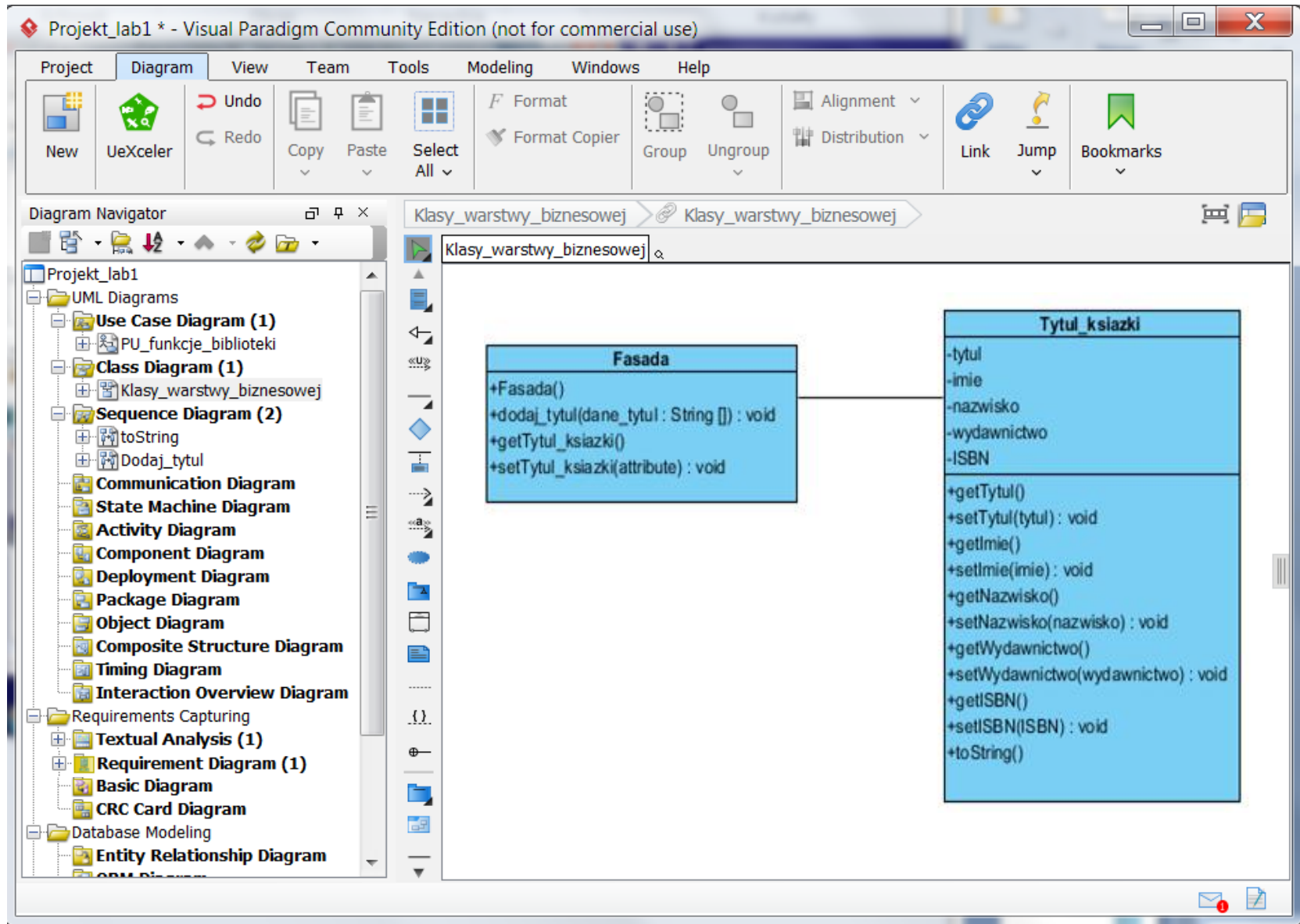


12.20. Uwidocznienie nowej metody **dodaj_tytul** z listą parametrów w klasie **Fasada**, zdefiniowanej na diagramie sekwencji z p.11.19

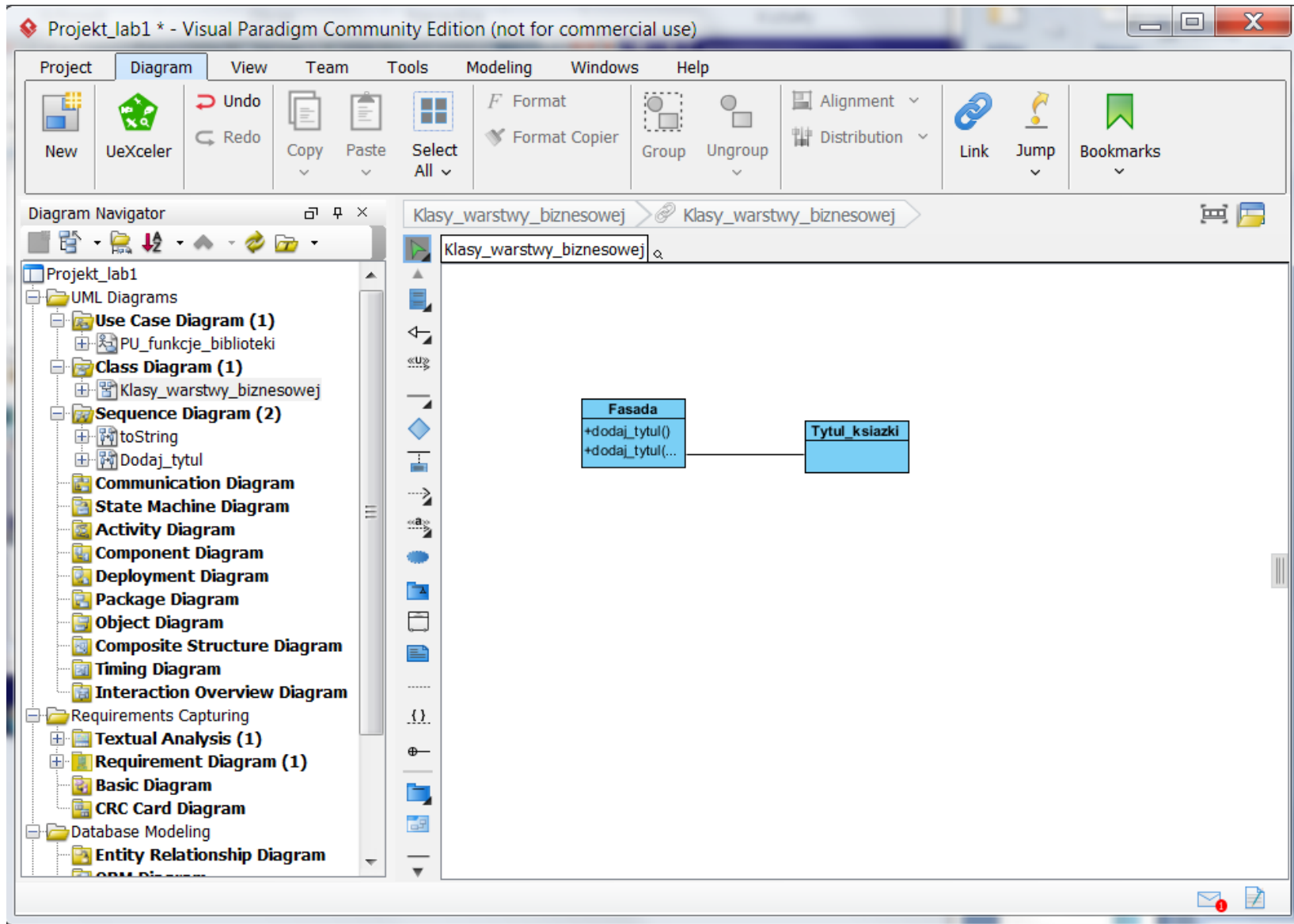
The screenshot displays the Visual Paradigm Community Edition interface. The main workspace shows a UML Class Diagram with two classes: **Fasada** and **Tytul_książki**. The **Fasada** class has two methods: `+Fasada()` and `+dodaj_tytul(dane_tytul : String []) : void`. The **Tytul_książki** class has private attributes: `-tytul`, `-imie`, `-nazwisko`, `-wydawnictwo`, and `-ISBN`. It also has public methods: `+getTytul()`, `+setTytul(tytul) : void`, `+getImie()`, `+setImie(imie) : void`, `+getNazwisko()`, `+setNazwisko(nazwisko) : void`, `+getWydawnictwo()`, `+setWydawnictwo(wydawnictwo) : void`, `+getISBN()`, `+setISBN(ISBN) : void`, and `+toString()`. A solid line connects the two classes, indicating an association.

The Diagram Navigator on the left shows the project structure for **Projekt_lab1**, including UML Diagrams, Class Diagram (1), and Sequence Diagram (2). The Class Diagram (1) contains the **Klasy_warstwy_biznesowej** package, which contains the **Fasada** class.

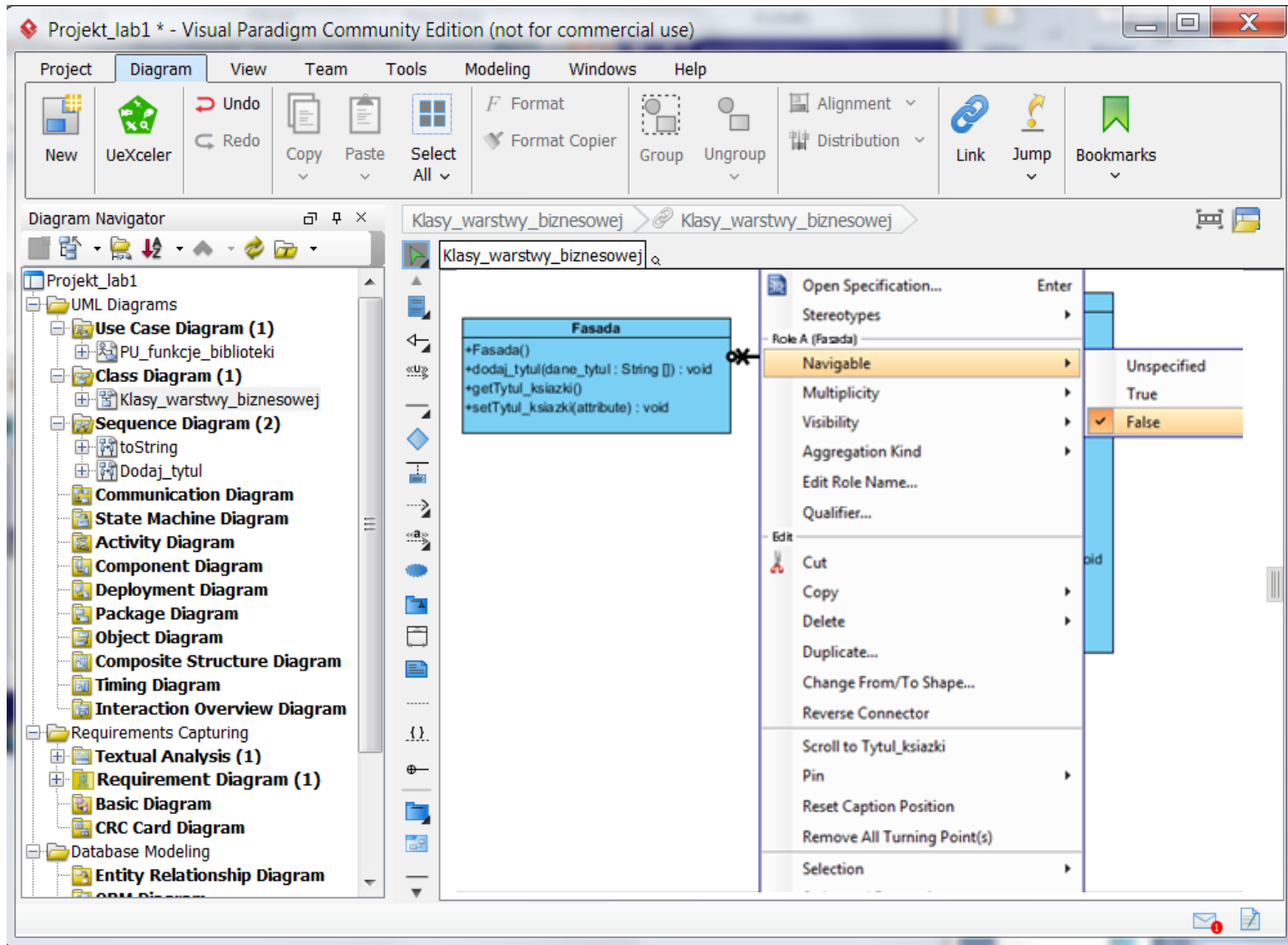
12.21. Dodanie dwóch operacji typu **get** i **set**, reprezentujących implementację relacji *Association* w klasie **Fasada** (obiekt typu **Fasada** będzie miał dostęp do jednego obiektu typu **Tytul_książki**)



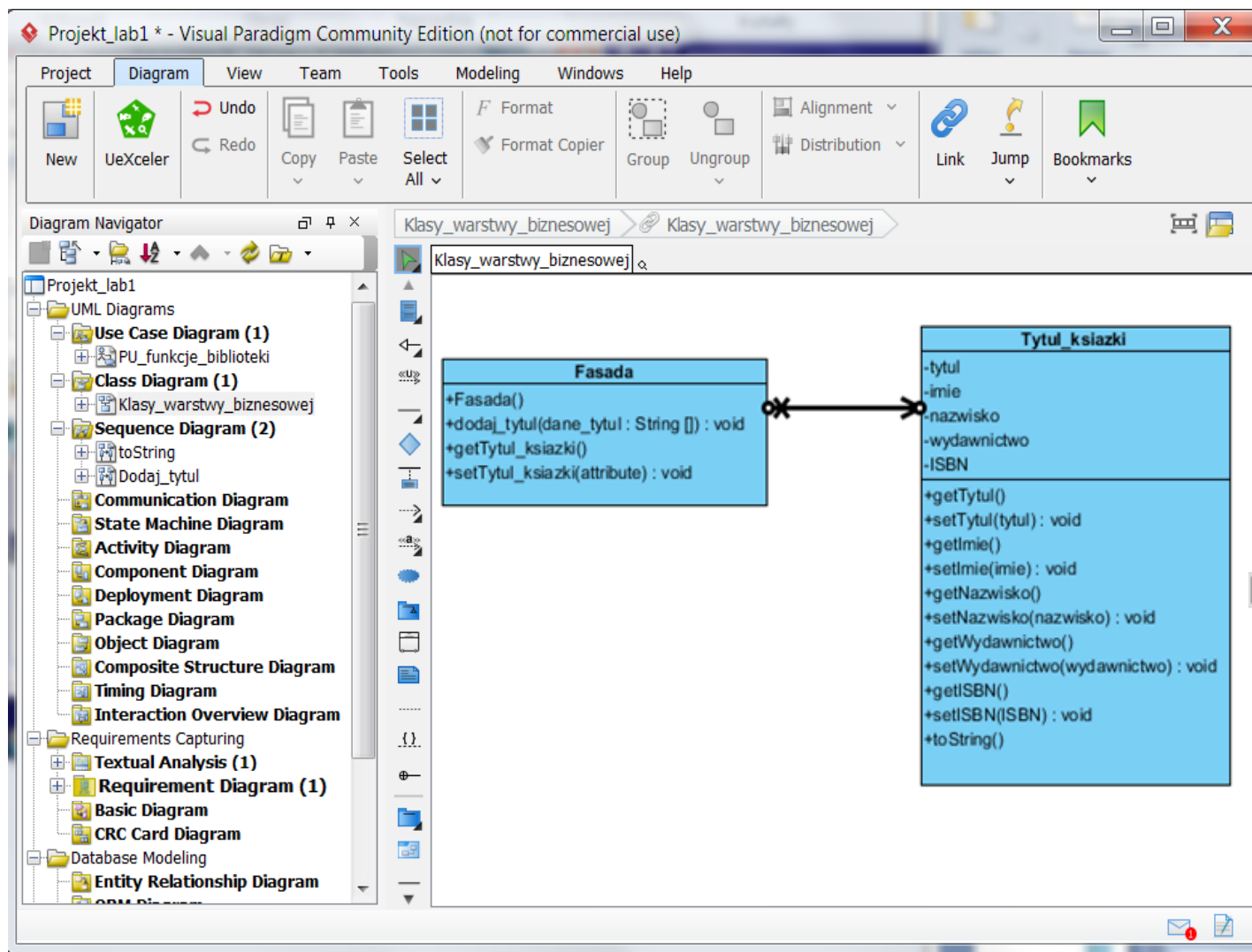
12.22. Definicja powiązania jednokierunkowego typu *Association* – należy kliknąć prawym klawiszem na koniec relacji powiązanej z klasą **Tytul_książki** i wybrać z listy opcje *Navigable/True*



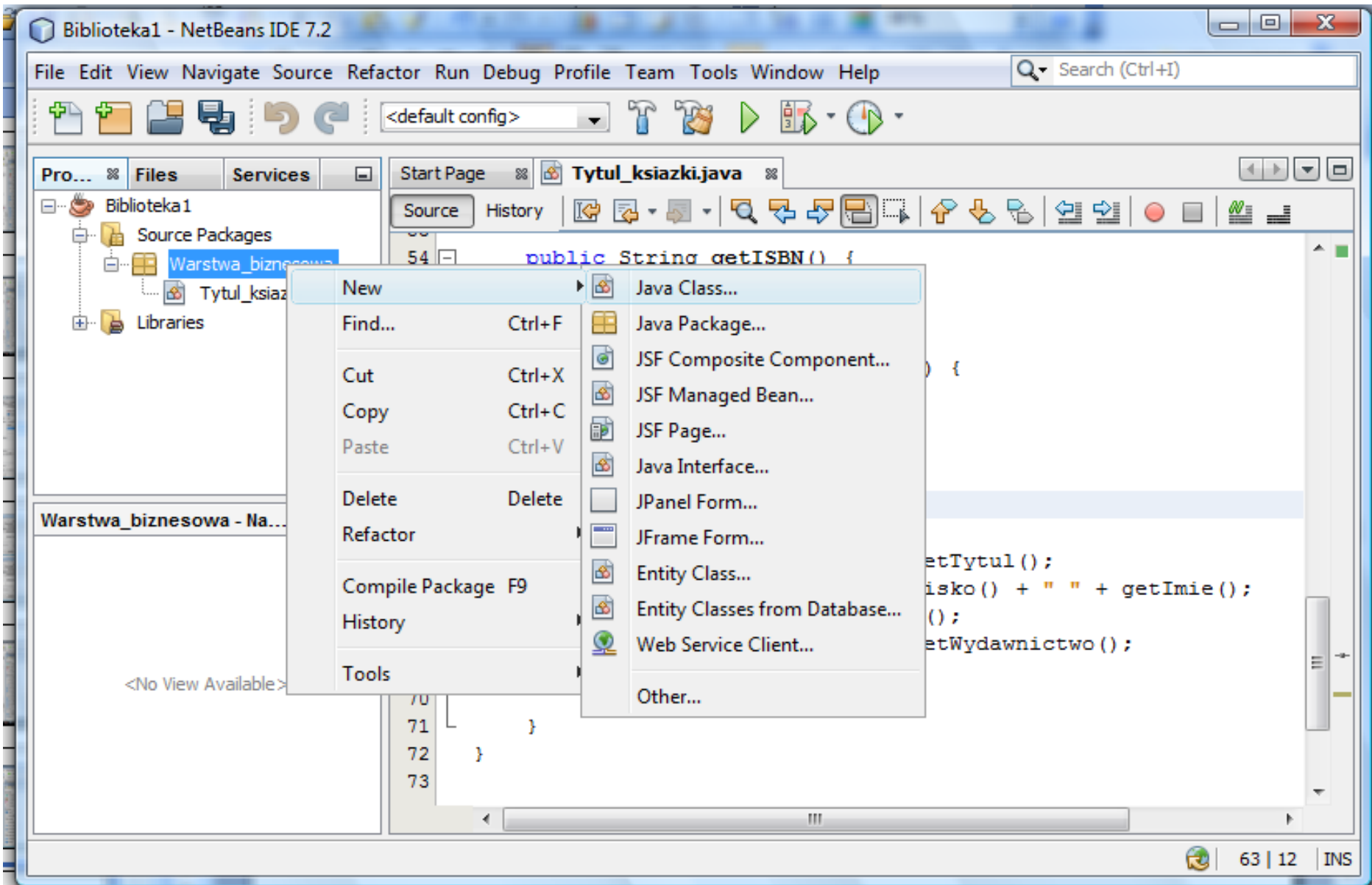
12.23. Definicja powiązania jednokierunkowego typu *Association* – należy kliknąć prawym klawiszem na koniec relacji powiązanej z klasą **Fasada** i wybrać z listy opcje *Navigable/False*



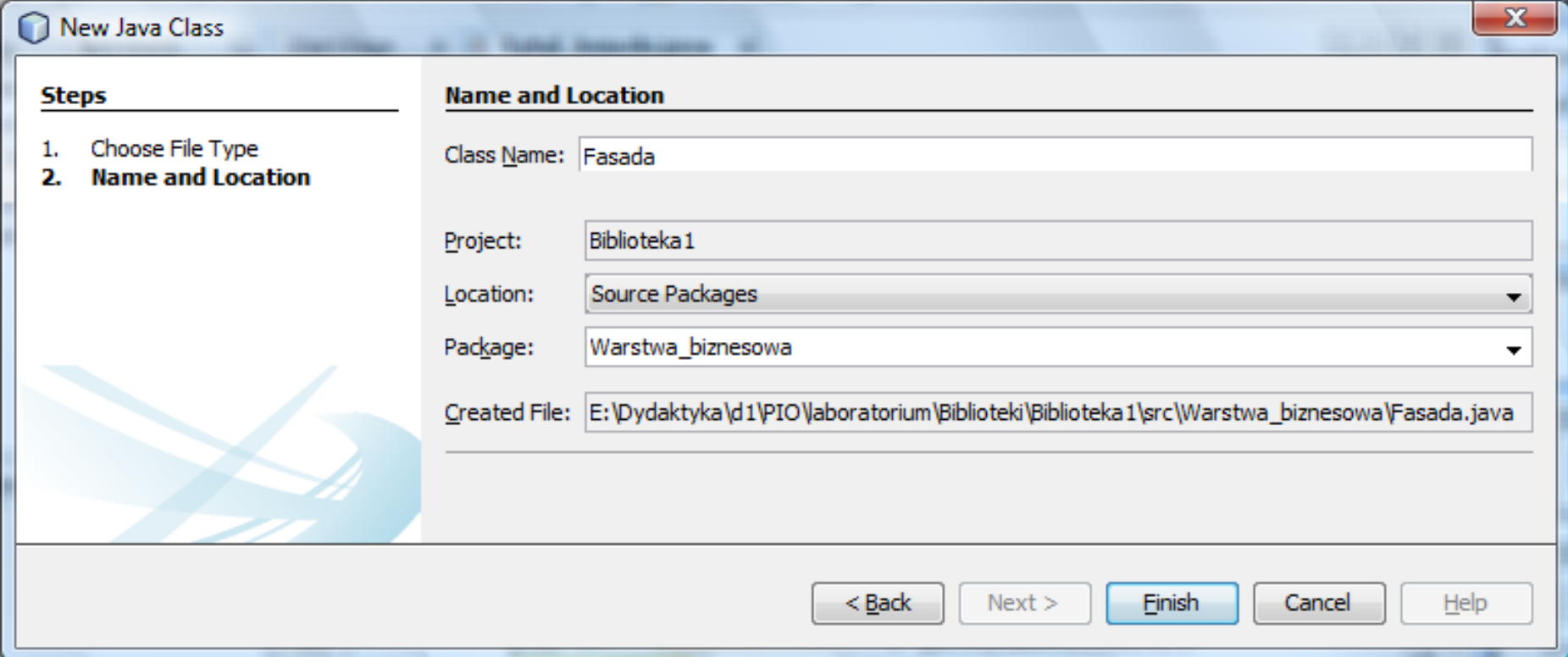
12.24. Uzyskano jednokierunkową relację typu *Association* – oznacza ona, że obiekt typu **Fasada** zawiera referencję do obiektu typu **Tytul_książki**, natomiast obiekt typu **Tytul_książki** nie ma dostępu do obiektu klasy **Fasada** (definicja relacji jednokierunkowej typu 1..0..1)



13/13.1. Uzupełnienie kodu projektu **Java Application** zawierającego implementację diagramów sekwencji **dodaj_tytul** – należy dodać nową klasę typu **Fasada** (podobnie jak **Tytul_książki**)



13.2. Nadanie nazwy nowej klasie w polu *Class Name*



New Java Class

Steps

1. Choose File Type
- 2. Name and Location**

Name and Location

Class Name: Fasada

Project: Biblioteka 1

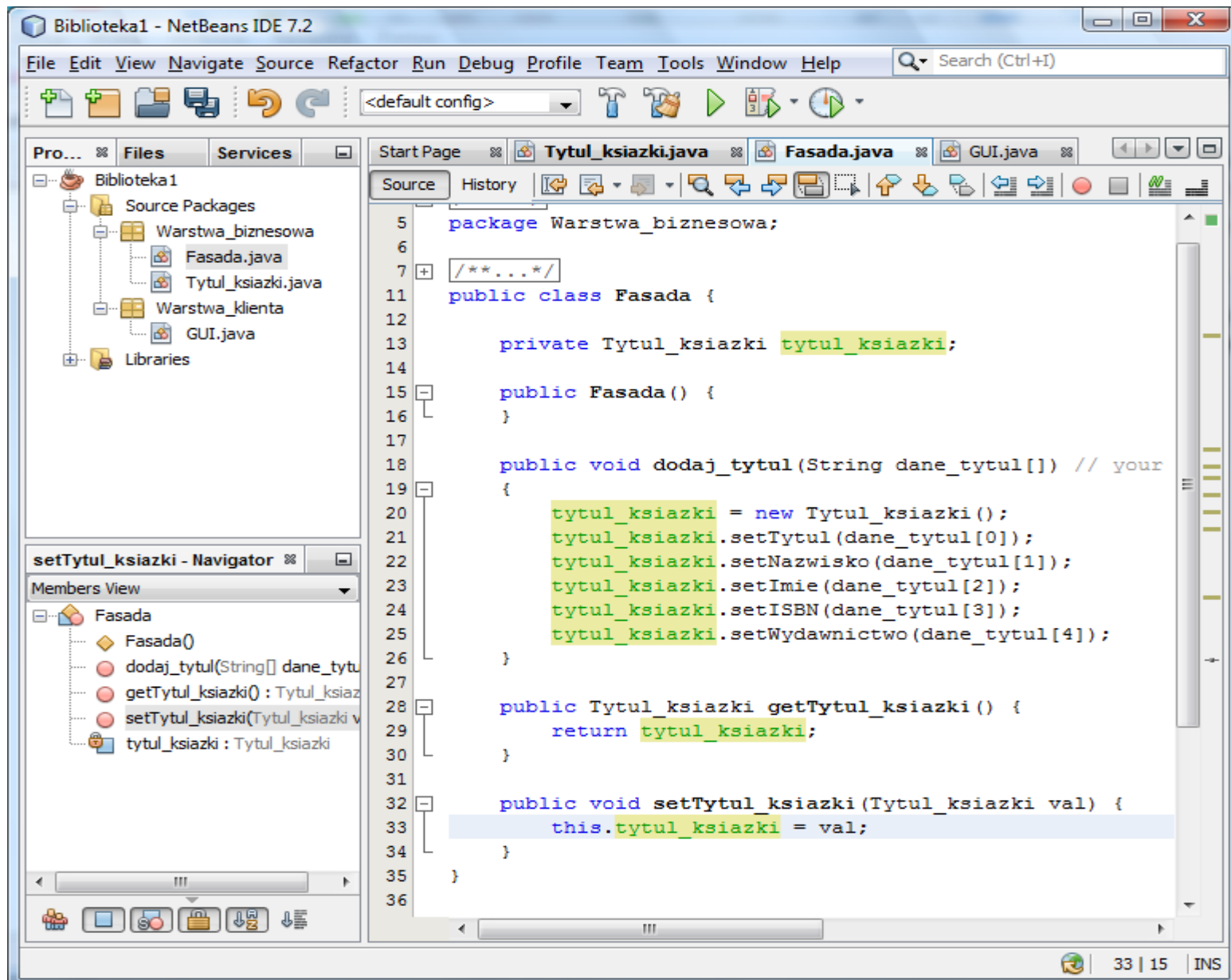
Location: Source Packages

Package: Warstwa_biznesowa

Created File: E:\Dydaktyka\d1\PIO\laboratorium\Biblioteki\Biblioteka 1\src\Warstwa_biznesowa\Fasada.java

< Back Next > **Finish** Cancel Help

13.3. Dodanie kodu metody `dodaj_tytul` do klasy `Fasada` wg diagramu z p.11.19 oraz metody typu setter i getter dla referencji typu `Tytul_książki`



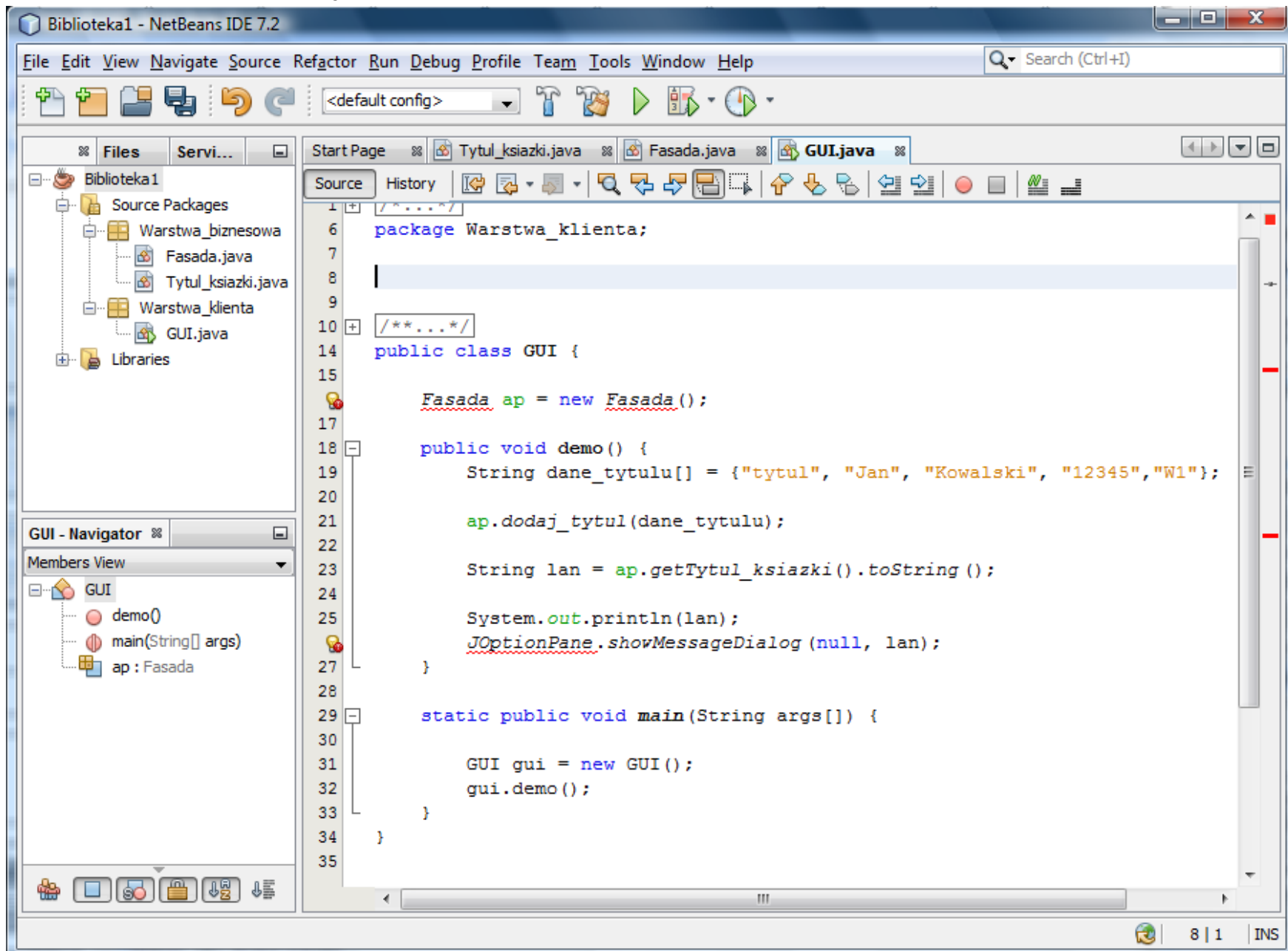
13.4. Kod klasy Fasada

```
package Warstwa_biznesowa;

public class Fasada {
    private Tytul_książki tytul_książki;
    public Fasada() {
    }
    public void dodaj_tytul(String dane_tytul[]) // your code here
    {
        tytul_książki = new Tytul_książki(); //implementacja Create Message
        tytul_książki.setTytul(dane_tytul[0]);
        tytul_książki.setNazwisko(dane_tytul[1]);
        tytul_książki.setImie(dane_tytul[2]);
        tytul_książki.setISBN(dane_tytul[3]);
        tytul_książki.setWydawnictwo(dane_tytul[4]);
    }
    public Tytul_książki getTytul_książki() {
        return tytul_książki;
    }
    public void setTytul_książki(Tytul_książki val) {
        this.tytul_książki = val;
    }
}
```

Referencja do obiektu klasy Tytul_książki reprezentuje relację 1 do 1 po stronie klasy Fasada, która jest „właścicielem” relacji

13.5. Kod klasy **GUI** reprezentujący warstwę klienta (interfejs graficzny użytkownika) z pakietu **Warstwa_klienta**

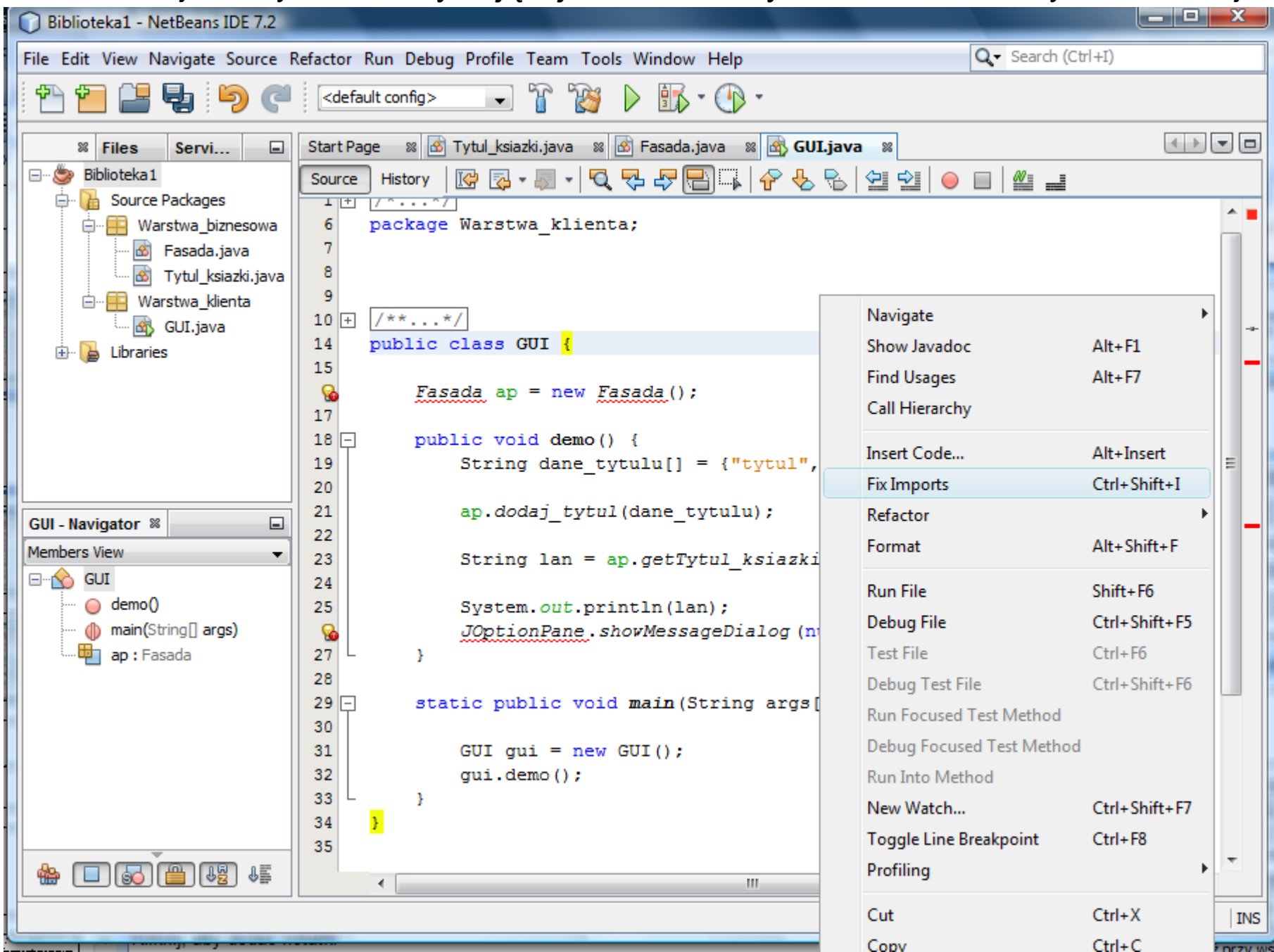


The screenshot displays the NetBeans IDE 7.2 interface. The main window shows the source code for the `GUI` class in the `Warstwa_klienta` package. The code is as follows:

```
1  /**...*/
2
3
4
5
6  package Warstwa_klienta;
7
8  |
9
10 /**...*/
11
12
13
14 public class GUI {
15
16
17     Fasada ap = new Fasada ();
18
19     public void demo () {
20         String dane_tytulu[] = {"tytul", "Jan", "Kowalski", "12345","W1"};
21
22         ap.dodaj_tytul(dane_tytulu);
23
24         String lan = ap.getTytul_ksiazki().toString ();
25
26         System.out.println(lan);
27         JOptionPane.showMessageDialog (null, lan);
28     }
29
30     static public void main(String args[]) {
31
32         GUI gui = new GUI ();
33         gui.demo ();
34     }
35 }
```

The IDE interface includes a menu bar (File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help), a toolbar, and a project explorer on the left showing the project structure: Biblioteka1 > Source Packages > Warstwa_biznesowa > Fasada.java, Tytul_ksiazki.java; Warstwa_klienta > GUI.java; and Libraries. The GUI - Navigator window shows the class hierarchy with members `demo()`, `main(String[] args)`, and `ap: Fasada`.

13.6. Definicja klasy **GUI** korzystającej z metod klasy **Fasada** z warstwy biznesowej



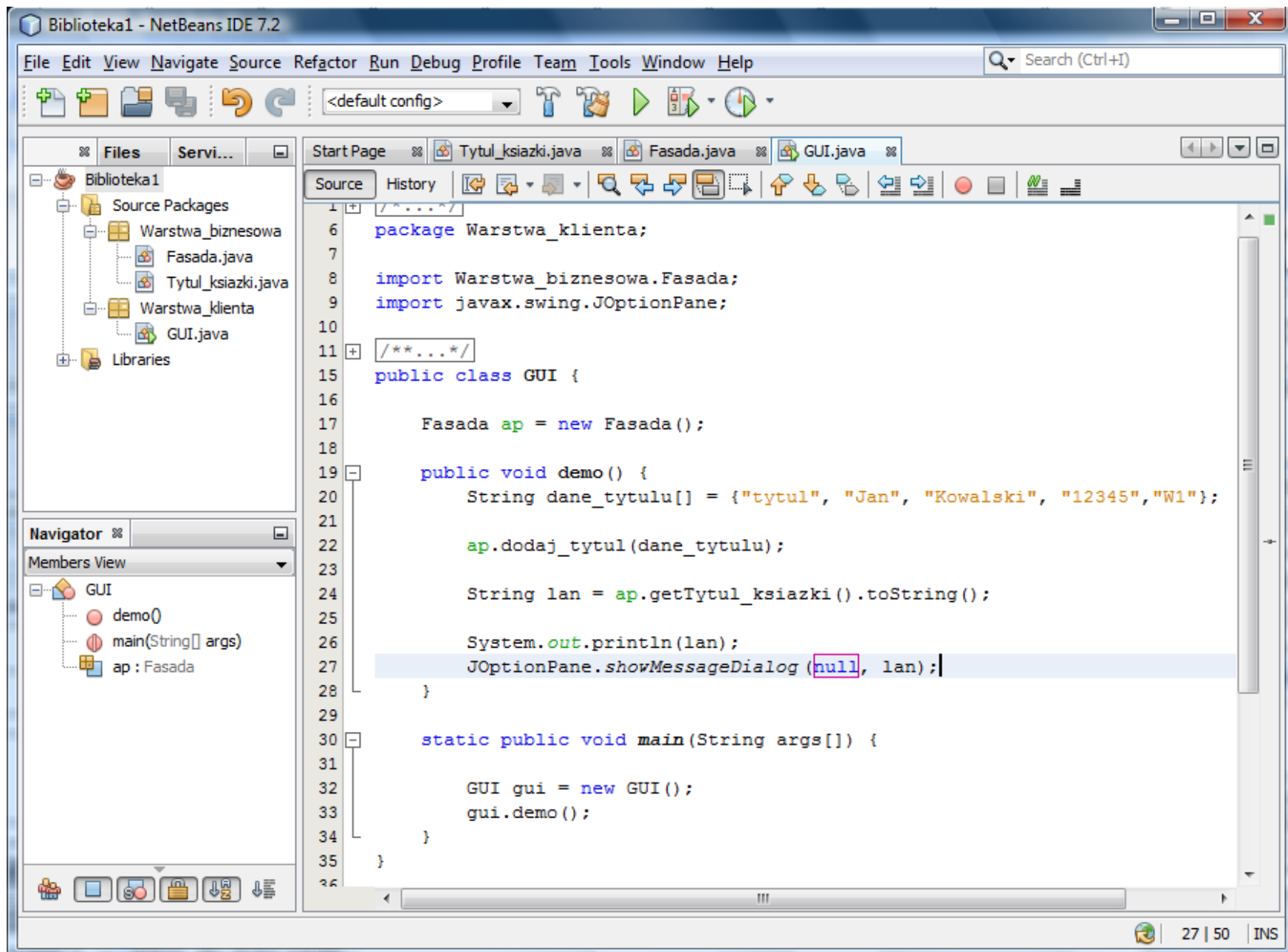
The screenshot displays the NetBeans IDE 7.2 interface. The main editor window shows the source code for `GUI.java` in the `Warstwa_klienta` package. The code defines a `GUI` class that uses the `Fasada` class from the `Warstwa_biznesowa` package. A context menu is open over the `GUI` class definition, listing various actions and their keyboard shortcuts.

```
1  /**...*/
2
3  package Warstwa_klienta;
4
5
6  /**...*/
7
8  public class GUI {
9
10
11
12
13     Fasada ap = new Fasada ();
14
15
16     public void demo () {
17         String dane_tytulu[] = {"tytul",
18
19
20
21
22
23
24
25
26         ap.dodaj_tytul(dane_tytulu);
27
28
29         String lan = ap.getTytul_książki();
30
31
32         System.out.println(lan);
33         JOptionPane.showMessageDialog (n
34     }
35
36     static public void main(String args[]
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

The context menu is open over the `GUI` class definition, showing the following items:

- Navigate
- Show Javadoc (Alt+F1)
- Find Usages (Alt+F7)
- Call Hierarchy
- Insert Code... (Alt+Insert)
- Fix Imports (Ctrl+Shift+I)
- Refactor
- Format (Alt+Shift+F)
- Run File (Shift+F6)
- Debug File (Ctrl+Shift+F5)
- Test File (Ctrl+F6)
- Debug Test File (Ctrl+Shift+F6)
- Run Focused Test Method
- Debug Focused Test Method
- Run Into Method
- New Watch... (Ctrl+Shift+F7)
- Toggle Line Breakpoint (Ctrl+F8)
- Profiling
- Cut (Ctrl+X)
- Copy (Ctrl+C)

13.7. Kod klasy **GUI** po uzupełnieniu importów pakietów **Warstwa_biznesowa** oraz **javax.swing**



The screenshot displays the NetBeans IDE 7.2 interface. The main editor window shows the source code for the `GUI.java` file. The code is as follows:

```
1  /**...*/
2
3  package Warstwa_klienta;
4
5
6  package Warstwa_klienta;
7
8  import Warstwa_biznesowa.Fasada;
9  import javax.swing.JOptionPane;
10
11 /**...*/
12
13 public class GUI {
14
15     Fasada ap = new Fasada();
16
17
18
19     public void demo() {
20         String dane_tytulu[] = {"tytul", "Jan", "Kowalski", "12345", "W1"};
21
22         ap.dodaj_tytul(dane_tytulu);
23
24         String lan = ap.getTytul_ksiazki().toString();
25
26         System.out.println(lan);
27         JOptionPane.showMessageDialog (null, lan);
28     }
29
30     static public void main(String args[]) {
31
32         GUI gui = new GUI();
33         gui.demo();
34     }
35 }
36
```

The `NullPointerException` is indicated by the `null` argument in the `JOptionPane.showMessageDialog` call on line 27, which is highlighted with a pink box in the original image.

The left sidebar shows the project structure for `Biblioteka1`, including source packages `Warstwa_biznesowa` and `Warstwa_klienta`, and the `GUI.java` file. The Navigator window shows the `GUI` class with methods `demo()` and `main(String[] args)`, and a field `ap : Fasada`.

The status bar at the bottom right shows the page number 27 out of 50 and the text "INS".

13.8. Kod klasy **GUI** korzystającej z metod klasy **Fasada**

```
package Warstwa_klienta;
```

```
import Warstwa_biznesowa.Fasada;
```

```
import javax.swing.JOptionPane;
```

```
public class GUI {
```

```
    Fasada ap = new Fasada();
```

```
    public void demo() {
```

```
        String dane_tytulu[] = {"tytul", "Jan", "Kowalski", "12345","W1"};
```

```
        ap.dodaj_tytul(dane_tytulu);
```

```
        String lan = ap.getTytul_ksiazki().toString();
```

```
        System.out.println(lan);
```

```
        JOptionPane.showMessageDialog(null, lan);
```

```
    }
```

```
    static public void main(String args[]) {
```

```
        GUI gui = new GUI();
```

```
        gui.demo();
```

```
    }
```

```
}
```


14. Uruchomienie programu – metoda **demo** z klasy GUI pozwala przetestować metody klasy **Fasada** i **Tytul_książki**

The screenshot shows the NetBeans IDE 7.2 interface. The main editor window displays the source code of `GUI.java` in the `Warstwa_klienta` package. The code includes imports for `Warstwa_biznesowa.Fasada` and `javax.swing.JOptionPane`. A `public class GUI` is defined, which creates a `Fasada` object and calls `JOptionPane.showMessageDialog` with a message string. The message string is `"Tytuł: tytuł Autor:Jan Kowalski ISBN: 12345 Wydawnictwo:W1"`. The `showMessageDialog` method is called with `null` as the first argument and `lan` as the second. The `System.out.println` statement is also present in the code.

A `Message` dialog box is displayed in the foreground, showing the message: `Tytuł: tytuł Autor:Jan Kowalski ISBN: 12345 Wydawnictwo:W1`. The dialog has an information icon and an `OK` button.

The `Output - Biblioteka1 (run)` window at the bottom shows the output of the program: `run: Tytuł: tytuł Autor:Jan Kowalski ISBN: 12345 Wydawnictwo:W1`.