

DOI: <https://doi.org/10.15276/ict.03.2026.48>

УДК 004.451:378.147

Предметно-орієнтоване семантичне представлення файлових операцій командної оболонки Bash у навчанні дисципліни «Операційні системи»

Блажко Олександр Анатолійович¹⁾

ORCID: <https://orcid.org/0000-0001-7413-275X>; blazhko@op.edu.ua. Scopus Author ID: 57195410976

Задерейко Олександр Владиславович²⁾

ORCID: <https://orcid.org/0000-0003-0497-9861>; zadereyko@onua.edu.ua. Scopus Author ID: 57196005917

Гура Володимир Ігорович²⁾

ORCID: <https://orcid.org/0000-0001-6415-7865>; hura@onua.edu.ua

¹⁾ Національний університет «Одеська політехніка», пр. Шевченка, 1. Одеса, 65044, Україна

²⁾ Національний університет «Одеська юридична академія», Фонтанська дорога, 23. Одеса, 65009, Україна

АНОТАЦІЯ

У роботі розглянуто методику предметно-орієнтованого семантичного представлення базових файлових операцій командної оболонки Bash для навчання дисципліни «Операційні системи». Актуальність зумовлена потребою у формуванні у студентів зв'язків між предметними діями та файловими операціями. Методика містить сім етапів: визначення базового набору Bash-команд, шаблонну структуру опису предметної області, визначення предметної області за критеріями, визначення предметних аналогів базових об'єктів файлової системи (рольовий клас, клас-контейнер, клас-частина), визначення операційних відповідників, технічну реалізацію через Bash-псевдоніми та узгодження з таксономією Блума. Запропоновано критерії вибору предметної області, серед яких наявність дворівневої структури «контейнер-елемент», можливість узгодженого відображення «задача-дія» та функціональна повнота. Розроблено алгоритми визначення предметних аналогів і операційних відповідників, що дають змогу зіставляти Bash-команди з предметними діями рольового класу. Технічна реалізація через псевдоніми не змінює функціональність базових команд і не створює нових операцій файлової системи. Для етапів визначення предметних аналогів базових об'єктів файлової системи, визначення операційних відповідників та технічної реалізації через псевдоніми встановлено відповідність категоріям виміру знань (концептуальне, процедурне) та виміру когнітивних процесів (розуміти, застосовувати, аналізувати, оцінювати, створювати) таксономії Блума. Це дає змогу використовувати етапи як основу для методичних вказівок студентам під час виконання завдань різного когнітивного рівня. Практичне значення полягає у можливості формування варіативних навчальних завдань на основі семантичних відповідників. Подальші дослідження доцільно спрямувати на експериментальну перевірку методики, розширення базового набору команд і предметних областей та удосконалення технічної реалізації через Bash-функції.

Ключові слова: операційні системи; командна оболонка Bash; предметно-орієнтоване семантичне представлення; таксономія Блума; псевдоніми Bash; командний інтерфейс; методика навчання

Для цитування: Блажко О. А., Задерейко О. В., Гура В. І. «Предметно-орієнтоване семантичне представлення файлових операцій командної оболонки Bash у навчанні дисципліни «Операційні системи»». *Інформатика. Культура. Техніка.* 2026; Том 3 № 2 (4): 637–646. DOI: <https://doi.org/10.15276/ict.03.2026.48>

Актуальність. Попри поширення графічних та інших візуальних інтерфейсів, командний інтерфейс користувача (*Command-Line Interface, CLI*) залишається важливим засобом взаємодії з операційними системами (ОС), хоча дослідження підтверджують наявність специфічних проблем використання *CLI* [1], коли текстова форма *CLI* передбачає введення користувачем команд для задання необхідних операцій. Важливим аспектом взаємодії є встановлення відповідності між задачею користувача та формальним засобом її виконання. Дослідження командних мов розглядають структуру таких мов та способи встановлення відповідності між діями користувача і синтаксичними конструкціями команд [2], [3]. Встановлення відповідності між операцією, яку необхідно виконати, та командою, за допомогою якої вона реалізується, можна вважати важливим і в навчанні командних мов. Одним із засобів полегшення такого співвіднесення є використання інтерфейсних метафор, за допомогою яких елементи інтерфейсу пов'язуються з діями, відомими користувачеві з інших контекстів [4], [5]. Такий підхід дозволяє розглядати команду не лише як синтаксичну конструкцію для запам'ятовування, а і як засіб виконання операції. Інтерфейсні метафори досліджувалися для проектування взаємодії, але можуть розглядатися як основу для полегшення співвіднесення задачі та команди в навчанні. Тому для навчання командної

оболонки *Bash* це створює основу для встановлення семантичних відповідностей між операціями предметної області та файловими операціями операційної системи. У цій роботі під семантичним відповідником розуміється поняття або дія предметної області, функціонально пов'язана з певною операцією командної оболонки. Такий відповідник не є синонімом команди, а використовується для представлення її функціонального змісту через поняття, відоме користувачеві в межах предметної області, що узгоджується з підходами до семантичного зіставлення [9]. Одним із способів реалізації такого представлення є використання прикладів для пояснення абстрактних операцій. Результати дослідження використання прикладів у навчанні абстрактного матеріалу дають підстави розглядати їх як можливий засіб підтримки такого навчання [11]. Водночас можливість перенесення цього підходу на навчання командних операцій *Bash* потребує окремої перевірки. Засвоєння команд *Bash* не зводиться лише до запам'ятовування назв і синтаксису команд. Навчальні завдання можуть вимагати від студентів пояснення призначення команди, її практичного застосування, аналізу послідовності операцій, перевірки правильності результату, що може бути співвіднесено з переглянутою таксономією когнітивних процесів таксономії Блума, яка охоплює запам'ятовування, розуміння, застосування, аналіз, оцінювання та створення [12]. У командних операціях *Bash* є можливість побудови семантичних відповідностей із поняттями предметної області та формування на їх основі навчальних завдань із різними рівнями когнітивної складності. Попри наявності окремих досліджень командних мов, метафор, семантичного зіставлення та таксономії Блума, залишається невирішеним питання цілісного поєднання предметно-орієнтованого семантичного представлення файлових операцій *Bash* у поєднанні з навчальними завданнями за когнітивними рівнями.

Метою дослідження є розроблення цілісної методики предметно-орієнтованого семантичного представлення базових файлових операцій *Bash*-команд, яка включає формалізовані критерії вибору предметної області, алгоритми побудови предметних аналогів об'єктів файлової системи та операційних відповідників *Bash*-команд, правило технічної реалізації через *Bash*-функції, а також процедуру узгодження етапів методики з вимірами знань і когнітивних процесів таксономії Блума для формування навчальних завдань. Саме поєднання вказаних вище складових методики і визначає наукову новизну роботи.

Аналіз останніх досліджень та публікацій. Проблема взаємодії користувача з *CLI* досліджується у командних мовах, інтерфейсних метафорах, семантичному зіставленні та таксономії Блума в комп'ютерній освіті, які формують передумови використання предметно-орієнтованих семантичних відповідників для представлення операцій командної оболонки *Bash*. Робота [2] пропонує граматику *CLI* як формальне представлення інтерфейсу інтерактивних систем, яка розділяє систему на концептуальний, комунікаційний і фізичний компоненти на рівні задачі, семантики, синтаксису та взаємодії. Робота [3] експериментально показує, що структурування відображень «задача-дія» в семантичних доменах полегшує засвоєння командних мов, коли взаємодія з *CLI* встановлює зв'язок між дією, яку необхідно виконати, та її командним представленням. Для вивчення *Bash*-команд це обґрунтовує необхідність формування у студента зв'язку між операцією та відповідною командою.

У роботі [4] розглянуто метафору інтерфейсу як частину мови для функціонування в інтерфейсі. У роботі [5] концептуальні метафори розглядаються як інструмент дизайну, що відкриває нові напрямки для проєктування взаємодії в інтелектуальних середовищах. В роботі [6] розглянуто лінгвістичний аспект метафори, де комп'ютерна метафора є одним з механізмів створення нової комп'ютерної лексики через подібність та аналогію, та представлено назву системної операції як лінгвістичний синонім технічного терміну, як семантичний відповідник, сформований на основі функціональної аналогії між операціями різних предметних областей.

Проблема встановлення відповідностей між діями розглядається також у дослідженнях зіставлення онтологій та подібності моделей бізнес-процесів [7], [10]. У роботі [7] запропоновано алгоритм, який використовує лексичні та структурні характеристики двох онтологій, ітеративно обчислює міру подібності, формує вирівнювання та верифікує його на

семантичні несуперечності. У роботі [8] запропоновано три метрики подібності моделей бізнес-процесів: зіставлення вузлів (за мітками та атрибутами), структурну (мітки + топологія) та поведінкову (мітки + причинно-наслідкові зв'язки). У роботі [9] запропоновано підхід на основі вбудованих слів, який підвищує точність зіставлення відповідних активностей між парою моделей процесів. У роботі [10] подано аналіз понад сотні мір подібності моделей бізнес-процесів, їх класифікацію та визначення прогалів, а результати показують, що встановлення відповідності може ґрунтуватися на збігу назв, на семантичній та функціональній подібності. Це може стати близьким для визначення предметно-орієнтованих відповідників *Bash*-команд. Дослідження навчання абстрактних понять через конкретні приклади [11] експериментально підтверджують, що їх використання сприяє засвоєнню абстрактного матеріалу та його розпізнаванню в нових контекстах. Для навчання *Bash* це створює передумову для представлення абстрактних операцій файлової системи через конкретні процеси, знайомі студенту з певної предметної області.

Для диференціації навчальних завдань за рівнем когнітивної діяльності використовується таксономія Блума [12], яка включає два виміри: знання (фактичне, концептуальне, процедурне) та когнітивність (запам'ятовування, розуміння, застосування, аналіз, оцінювання, створення). У роботі [13] на основі систематичного огляду досліджень показано, що таксономію Блума найчастіше використовують у навчанні програмування та для оцінювання результатів, а однією з основних проблем є визначення когнітивного рівня завдання. Все це дає змогу співвідносити навчальні завдання з командної оболонки за змістом та рівнем когнітивної діяльності.

Отже, попередні дослідження окремо обґрунтовують відповідність задач і команд, метафоричного перенесення понять, семантичне зіставлення дій, конкретизацію абстрактних понять у навчанні та застосування таксономії Блума в ІТ-освіті. Але залишається відкритим питання поєднання цих положень для предметно-орієнтованого семантичного представлення базових *Bash*-команд із диференціацією навчальних завдань за когнітивними рівнями.

Виклад основного матеріалу дослідження

1. Етапи методики предметно-орієнтованого представлення файлових операцій *Bash*

Пропонується методика предметно-орієнтованого семантичного представлення базових файлових операцій командної оболонки *Bash*, яка містить наступні етапи (Рис. 1):

- 1) визначення базового набору *Bash*-команд файлової системи *Unix*-подібної ОС [14];
- 2) визначення критеріїв вибору предметної області (ПрО) та вимог до спрощеного опису ПрО, який дозволяє використовувати ПрО у початкових завданнях для студентів;
- 3) формування ПрО на вимог до опису ПрО;
- 4) визначення предметних аналогів базових об'єктів файлової системи на основі опису концепт-моделі ПрО, результатом якого є *UML*-діаграма концептуальних класів ПрО;
- 5) визначення операційних відповідників на основі базового набору *Bash*-команд та *UML*-діаграми концептуальних класів ПрО;
- 6) визначення особливостей технічної реалізації операційних відповідників через опис псевдонімів для нових *Bash*-команд;
- 7) узгодження процесу предметно-орієнтованого семантичного представлення *Bash*-команд із рівнями таксономії Блума для підготовки методичних вказівок до завдань.

Для узгодження процесу предметно-орієнтованого семантичного представлення *Bash*-команд із рівнями таксономії Блума [12] пропонується розглядати етапи методики, які співвідносяться з виміром знань та виміром когнітивних процесів:

- етап 4 – концептуальне знання; категорії «розуміти» (інтерпретація, наведення прикладів), «аналізувати» (диференціація, організація, атрибуція) когнітивних процесів;
- етап 5 – концептуальне/процедурне знання; категорії «застосовувати» (виконання, впровадження), «аналізувати» (диференціація, організація, атрибуція), «оцінювати» (перевірка), «створювати» (генерування, продукування) вимір когнітивних процесів;

– етап 6 – процедурне знання; категорії «розуміти» (пояснення), «застосовувати» (виконання, впровадження), «оцінювати» (перевірка), «створювати» (генерування, планування, продукування) когнітивних процесів.

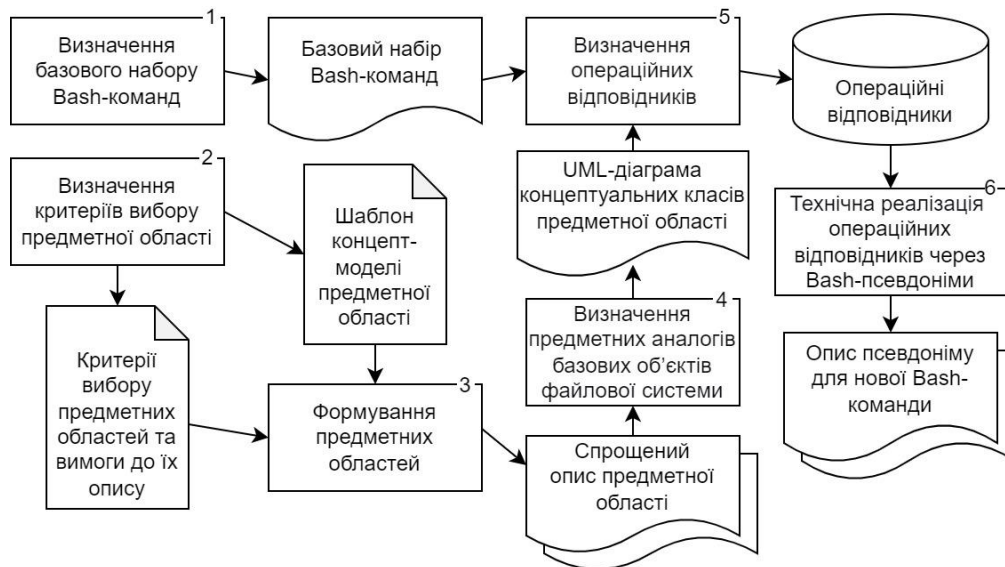


Рис. 1. Методика предметно-орієнтованого семантичного представлення базових файлових операцій командної оболонки Bash

Джерело: розроблено авторами

2. Визначення базового набору Bash-команд керування файловою системою

Відомо, що будь-яке керування будь-яким об'єктом включає в себе чотири основні операції (CRUD-операції): *Create* – створення об'єкта, *Read* – перегляд значень атрибутів об'єкта, *Update* – зміна значень атрибутів об'єкта, *Delete* – видалення об'єкта. На основі аналізу Bash-команд Unix-подібних ОС [14] запропоновано базові Bash-команди керування файловою системою, які можна семантично зіставити з CRUD-операціями, якщо розглядати файл і каталог як сутності, а їхні метадані та вміст – як стани. Команди *ls* і *cat* реалізують читання (*Read*) відповідно метаданих контейнера та вмісту елемента; *cp* – створення (*Create*) нового елемента шляхом дублювання вмісту; *mv* – оновлення (*Update*) прив'язки імені до нового вузла (*inode*) файлової системи; *rm*, *rmdir*, *rm -r* – видалення (*Delete*). Команда *cd* є винятком: вона не виконує CRUD-операції над об'єктами файлової системи, а змінює (*Update*) розташування користувача, тому класифікується як навігаційна дія, що представлено на Рис. 2 у вигляді ланцюжка зв'язків між CRUD-операціями, назвами базових операцій та bash-командами.

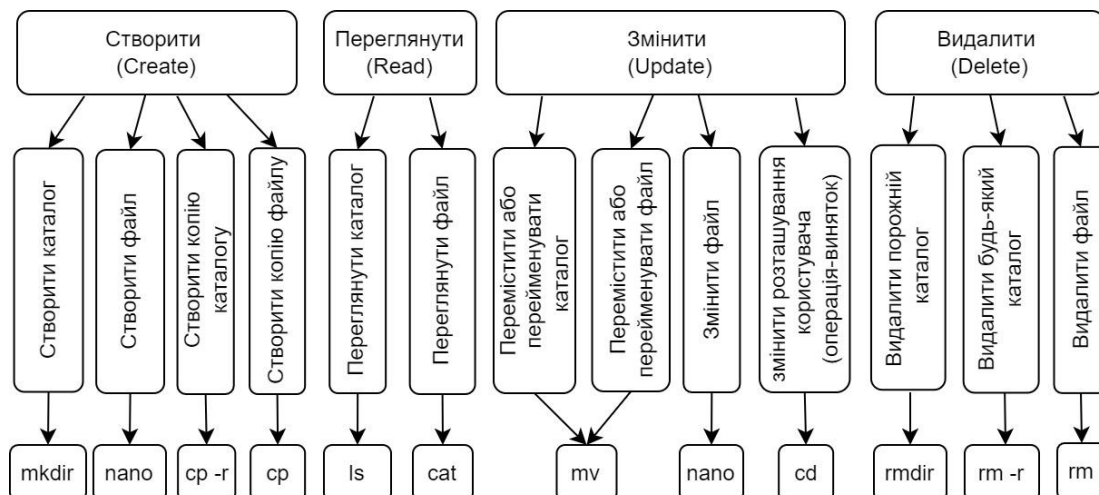


Рис. 2. Ланцюжки зв'язків між CRUD-операціями та відповідними bash-командами

Джерело: розроблено авторами

3. Критерії вибору ПрО та вимоги до опису ПрО для навчальних цілей

Вибір ПрО не є довільним, а базується на критеріях, за якими ПрО вважається придатною для побудови предметно-орієнтованих семантичних відповідників:

1) наявність дворівневої структури «контейнер-елемент»: ПрО має містити об'єкт-контейнер (роль *Aggregate*, аналог каталогу файлової системи) та об'єкт-елемент (роль *Part*, аналог файлу), який належить контейнеру з описів *UML*-діаграми концептуальних класів [15];

2) наявність вмісту контейнера та вмісту елемента: ПрО має визначити, що є «вмістом контейнера» (сукупність елементів) і «вмістом елемента» (його внутрішній стан або дані);

3) наявність зв'язку між контейнером і елементом: композиція – якщо елемент не може існувати поза контейнером, а видалення контейнера призводить до вилучення елемента, агрегація – якщо елемент може існувати окремо від контейнера як самостійний об'єкт;

4) можливість узгодженого відображення «задача-дія»: ПрО має встановлювати узгоджені відображення між діями в межах ПрО та операціями *Bash*-команд, за яких кожна дія ПрО має семантично близьку команду, а різні команди не мають однакових дій ПрО.;

5) можливість метафоричного перенесення за подібністю: ПрО має бути концептуально відмінною від файлової системи, але мати з нею структурну подібність, щоб метафоричне зіставлення «об'єкт ПрО – об'єкт файлової системи» було зрозумілим;

6) наявність покриття дій ПрО: ПрО має давати змогу побудувати семантичні відповідники з *Bash*-командами для більшості дій ПрО, присутніх в описі, а дії, що не мають відповідника, мають бути обґрунтовані;

7) зрозумілість для студентів: ПрО має бути знайомою або інтуїтивно зрозумілою студентам спеціальностей, які з нею не пов'язані, а термінологія ПрО не має вимагати спеціальних знань поза межами загальної освіти.

Після вибору ПрО, яка відповідає цим критеріям, постає завдання створити її початковий текстовий опис, придатний для навчального завдання із предметним контекстом, у якому студент має розпізнати дворівневу структуру «контейнер-елемент», визначити *UML*-ролі класів і встановити відповідності між діями ПрО та *Bash*-командами.

Тому навчальний текстовий опис ПрО пропонується формувати за наступними вимогами:

1) обсяг і структура тексту: текст опису ПрО має з двох речень загальним обсягом 25-30 слів, що визначає критерії 4, 5 через зрозумілість метафори та однозначність зіставлень;

2) обов'язкові слова-маркери: текст опису ПрО повинен містити одинадцять слів-маркерів, які відображають критерії вибору ПрО:

2.1) одне слово, що позначає рольовий клас, бо без актора неможливо визначити, хто виконує дії над контейнером і елементом, актор задає суб'єктну перспективу опису, пов'язуючи опис із критерієм 4, оскільки дії мають виконуватися суб'єктом;

2.2) одне слово, що позначає клас-контейнер, бо контейнер є аналогом каталогу файлової системи і без нього неможливо побудувати дворівневу структуру «контейнер-елемент», пов'язуючи опис із критерієм 1;

2.3) одне слово, що позначає клас-елемент як аналог файлу, який має агрегатний зв'язок із контейнером, бо елемент є аналогом файлу і агрегатний зв'язок дозволяє студентів побачити різницю між композицією та агрегацією, пов'язуючи опис із критеріями 1, 3;

2.4) три різні дієслова для позначення дії над класом-контейнером, які є семантичними синонімами до операцій *Bash*-команд, забезпечуючи мінімальне покриття для зіставлення з командами, які працюють із каталогом (*mkdir*, *cd*, *ls*), примушуючи студента до семантичного аналізу, а не до механічного пошуку слів та пов'язуючи опис із критеріями 4, 6;

2.5) три різні дієслова, що позначають дії над класом-елементом і є семантичними синонімами до операцій *Bash*-команд, але не збігаються з ними буквально, забезпечуючи покриття для команд, які працюють із файлом (*touch*, *cat*, *mv*, *rm*), а синонімічність підтримує розвиток семантичного мислення, пов'язуючи опис із критеріями 4, 6;

2.6) одне слово, що позначає дію над класом-контейнером, яка не має семантичного аналога серед наведених операцій *Bash*-команд, бо це дозволяє перевіряє, чи студент здатен відрізнити операції, які мають відповідник, від тих, що його не мають, запобігаючи механічному зіставленню «кожна дія – якась команда», та пов'язуючи опис із критерієм 4;

2.7) одне слово як дія над класом-елементом, яка не має семантичного аналога серед наведених *Bash*-команд, бо ця перевіряє здатність студента обґрунтовувати відсутність відповідника для елемента, а не лише для контейнера, пов'язуючи опис із критерієм 4.

На основі вказаних вимог створено 25 описів ПрО, які стануть частиною завдання [17].

4. Визначення предметних аналогів базових об'єктів файлової системи

Алгоритм визначення предметних аналогів базується на текстовому описі ПрО:

1) визначення сутностей, які виконують роль: класу-контейнера – аналог каталогу файлової системи; класу-елемента – аналог файлу; рольового класу – аналог актора ПрО.

2) обґрунтоване визначення зв'язку між класом-контейнером та класом-елементом:

– композиція – якщо елемент не може існувати поза контейнером, а видалення контейнера призводить до вилучення елемента, наприклад, *Bash*-команда «*rm -r*» видаляє каталог разом із вмістом, що імітує композицію;

– агрегація – якщо елемент може існувати окремо від контейнера, і, якщо обрано агрегацію, необхідно пояснити, як каскадне видалення узгоджується з можливістю елемента існувати окремо (наприклад, елемент може бути переміщений в інший контейнер);

3) спроектувати *UML*-діаграму концептуальних класів ПрО, враховуючи, що:

– дії над класом-контейнером треба перетворити на зв'язки-асоціації між рольовим класом та класом-контейнером;

– дії над класом-елементом треба перетворити на зв'язки-асоціації між рольовим класом та класом-елементом;

4) відобразити на діаграмі рольовий клас (*Actor*) як окремий клас та показати його зв'язки-асоціації з класом-контейнером і класом-елементом;

5) визначити кратність зв'язків на кінцях асоціацій (наприклад, '1', '0..1', '1..*', '*').

3 позиції таксономії Блума [12] представлений етап співвідноситься з виміром знань – концептуальне знання (знання класифікацій та категорій, знання принципів та узагальнень) і процедурне знання (знання предметно-специфічних навичок та алгоритмів) – та виміром когнітивних процесів: «розуміти» – інтерпретація, наведення прикладів, класифікація, пояснення, «аналізувати» – диференціація, організація, атрибуція, «оцінювати» – перевірка.

5. Визначення операційних відповідників між операціями ПрО та *Bash*-командами

Алгоритм визначення операційних відповідників містить наступні кроки:

1) для кожної операції *Bash*-команди визначити, об'єкт якого типу класу вона обробляє – контейнер чи елемент, і, якщо операція потребує об'єкт іншого типу класу, тоді вона не розглядається, враховуючи, що деякі *Bash*-команди можуть працювати з об'єктами обох типів класів (наприклад, команда *mv* – і з каталогом, і з файлом), її слід розглядати окремо для кожного типу класу або обрати той тип, який відповідає контексту ПрО;

2) визначити семантичну розбіжність між діями ПрО та операціями *Bash*-команд:

– визначити одну дію над класом-контейнером, яка не має семантичного аналога серед базових операцій *Bash*-команд;

– визначити одну дію над класом-елементом, яка не має семантичного аналога серед базових операцій *Bash*-команд;

3) для кожної операції *Bash*-команди, яка обробляє об'єкт типу контейнер, підібрати дію ПрО з переліку назв зв'язків-асоціацій між рольовим класом та класом-контейнером, яка виконує семантично близьку операцію;

4) для кожної операції *Bash*-команди, яка обробляє об'єкт типу елемент, підібрати дію ПрО з переліку назв зв'язків-асоціацій між рольовим класом та класом-елементом, яка виконує семантично близьку операцію;

5) перевірити однозначність: різні *Bash*-команди не повинні мати однакові дії ПрО;

6) сформувати таблицю семантичних відповідників зі стовпчиками: "Операція ПрО", "Операція *Bash*-команди", "*Bash*-команда", "Обґрунтування вибору".

Для збереження семантичних відповідників в *UML*-діаграмі запропоновано використовувати *UML*-коментар (*Note*), враховуючи, що: *Bash*-команда не є ні обмеженням (*Constraint*), ні стереотипом (*Stereotype*), ні іменованим значенням (*Tagged Value*); коментар не спотворює семантику асоціації. Для практичної реалізації *UML*-діаграми обрано *PlantUML* [16] – інструмент із відкритим вихідним кодом, який реалізує доменно-специфічну мову текстового опису *UML*-діаграм, що дозволяє декларативно генерувати діаграми без використання графічного редактора. З позиції таксономії Блума [12] етап співвідноситься з виміром знань (концептуальне та процедурне знання) та виміром когнітивних процесів: «застосовувати» – виконання, впровадження; «аналізувати» – диференціація, організація, атрибуція; «оцінювати» – перевірка; «створювати» – генерування, планування, продукування.

6. Визначення способу технічної реалізації операційних відповідників

Після встановлення семантичних відповідностей виникає технічна задача: яким способом предметно-орієнтована назва повинна бути пов'язана з реальною *Bash*-командою.

Можливі два принципово різні підходи:

1) створення окремого програмного засобу, який аналізує введену користувачем предметно-орієнтовану команду та перетворює її на відповідну *Bash*-команду;

2) використання наявних механізмів командної оболонки для встановлення альтернативної назви вже існуючої команди.

Для зменшення трудомісткості технічної реалізації пропонується 2-й підхід з використанням *Bash*-псевдонімів *alias* через зв'язування імені з командою або її послідовністю. Наприклад: після створення псевдоніму *alias zasiyaty='mkdir'* команда *zasiyaty pole* інтерпретується *Bash*-оболонкою як *mkdir pole*. Водночас *alias* придатний лише для простого додавання аргументів після базової команди, але не забезпечує довільного перетворення порядку аргументів, складної перевірки параметрів та уніфікованої обробки помилок. Тому *alias* доцільно застосовувати лише для тих відповідників, які семантично еквівалентні одній *Bash*-команді з простим додаванням аргументів. Для складніших випадків необхідно використовувати *Bash*-функції.

7. Приклад предметно-орієнтованого представлення операцій командної оболонки Bash

Розглянемо приклад ПрО «рослинництво» з наступним описом. Агроном має розмітити поле, перейти на нього, оглянути та розорати. Потім він має посадити рослину, полити її, пересадити, а за потреби – викорчувати. За методикою створено *UML*-діаграму класів у форматі *PlantUML*, на основі якої через вбудований інструмент згенеровано графічне представлення (Рис. 3). Детальний опис прикладу представлено у *GitHub*-репозиторії [17].

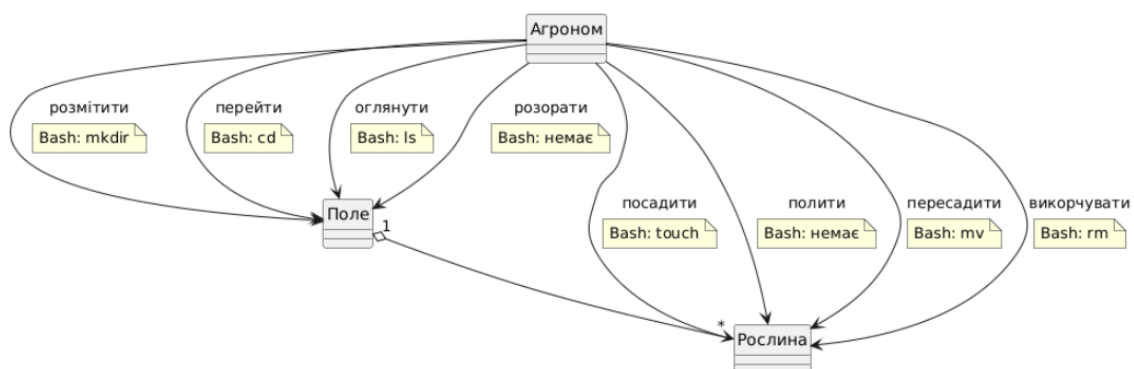


Рис. 3. Приклад *UML*-діаграми класів (*PlantUML*) для ПрО «Рослинництво»

Джерело: розроблено авторами

Висновки. У роботі запропоновано не просте поєднання відомих положень граматики «задача-дія», інтерфейсних метафор, навчання на прикладах і таксономії Блума, а цілісну методику, яка формалізує перехід від ПрО до предметно-орієнтованого семантичного представлення файлових операцій *Bash*-команд.

Її новизну утворюють:

- 1) 7 критеріїв вибору ПрО для побудови таких відповідників;
- 2) вимоги до спрощеного опису ПрО, які можна розглядати як текстовий шаблон на основі з 11 слів-маркерів, який дозволяє використовувати ПрО у початкових завданнях для студентів;
- 3) алгоритм визначення предметних аналогів базових об'єктів файлової системи та побудови *UML*-діаграми концептуальних класів;
- 4) алгоритм визначення операційних відповідників між діями предметної області та *Bash*-командами з перевіркою однозначності й покриття;
- 5) процедура узгодження етапів 4, 5 методики з вимірами знань і когнітивних процесів таксономії Блума.

Сьогодні методика не пройшла експериментальної апробації, тому отримані результати обмежуються підтвердженням можливістю формування навчальних завдань на основі семантичних відповідностей. Питання підвищення результативності навчання залишається відкритим і потребує окремого дослідження. Для перевірки методики на студентських роботах пропонується план експерименту із вимірюванням 4-х: коректності виконання команд, кількості помилок, часу виконання та якості перенесення знань на стандартні *Bash*-команди.

Подальші дослідження будуть спрямовані на експериментальну перевірку ефективності методики за запропонованим планом, розширення переліку предметних областей і базового набору команд, а також удосконалення технічної реалізації через *Bash*-функції для відповідників, які не зводяться до простих псевдонімів.

Конфлікт інтересів: Автори декларують, що не мають конфлікту інтересів, зокрема фінансового, особистого, авторського чи будь-якого іншого характеру, який міг би вплинути на дослідження, а також на результати, опубліковані в цій статті

Фінансування: Дослідження проводилося без фінансової підтримки

Використання засобів штучного інтелекту: Під час підготовки рукопису використано інструменти DeepSeek 4.1 та Gemini 3.5 штучного інтелекту (ШІ) для граматично-стилістичної перевірки тексту. Усі фрагменти, опрацьовані за його допомогою, були перевірені й відредаговані автором. ШІ не використовувався для формування наукових положень, отримання результатів або формулювання висновків. Наукові положення, результати та висновки є власним інтелектуальним внеском авторів

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Sampath, H., Merrick, M. A. & Macvean, A. “Accessibility of Command Line Interfaces”. *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. New York: Association for Computing Machinery. 2021: 489, <https://www.scopus.com/pages/publications/85106719414>. DOI: <https://doi.org/10.1145/3411764.3445544>.
2. Moran, T. P. “The Command Language Grammar: a representation for the user interface of interactive computer systems”. *International Journal of Man-Machine Studies*. 1981; 15 (1): 3–50, <https://www.scopus.com/pages/publications/0019596228>. DOI: [https://doi.org/10.1016/S0020-7373\(81\)80022-3](https://doi.org/10.1016/S0020-7373(81)80022-3).
3. Payne, S. J. & Green, T. R. G. “The structure of command languages: an experiment on task-action grammar”. *International Journal of Man-Machine Studies*. 1989; 30 (2): 213–234, <https://www.scopus.com/pages/publications/0024607787>. DOI: [https://doi.org/10.1016/S0020-7373\(89\)80011-2](https://doi.org/10.1016/S0020-7373(89)80011-2).
4. Erickson, T. “Working with interface metaphors”. In Baecker, R. M., Grudin, J., Buxton, W. A. S. & Greenberg, S., eds. *Readings in Human-Computer Interaction: Toward the Year 2000*. 2nd ed. San Francisco: Morgan Kaufmann. 1995. p. 79–86. DOI: <https://doi.org/10.1016/B978-0-08-051574-8.50018-2>.

5. Kim, J. & Maher, M. L. “Conceptual Metaphors for designing smart environments: Device, robot, and friend”. *Frontiers in Psychology*. 2020; 11: 198. DOI: <https://doi.org/10.3389/fpsyg.2020.00198>.
6. Богачик, М. С. “Комп’ютерна метафора як один з провідних механізмів семантичної деривації в сучасній англійській мові”. *Наукові записки Національного університету «Острозька академія»: серія «Філологія»*. 2018; 1 (69), ч. 1: 64–66. DOI: [https://doi.org/10.25264/2519-2558-2018-1\(69\)/1-64-66](https://doi.org/10.25264/2519-2558-2018-1(69)/1-64-66).
7. Jean-Mary, Y. R., Shironoshita, E. P. & Kabuka, M. R. “Ontology matching with semantic verification”. *Web Semantics: Science, Services and Agents on the World Wide Web*. 2009; 7 (3): 235–251, <https://www.scopus.com/pages/publications/71149117828>. DOI: <https://doi.org/10.1016/j.websem.2009.04.001>.
8. Dijkman, R. M., Dumas, M., Van Dongen, B. F., Käärik, R. & Mendling, J. “Similarity of business process models: Metrics and evaluation”. *Information Systems*. 2011; 36 (2): 498–516, <https://www.scopus.com/pages/publications/78649464425>. DOI: <https://doi.org/10.1016/j.is.2010.09.006>.
9. Shahzad, K., Kanwal, S., Malik, K., Aslam, F. & Ali, M. “A word-embedding-based approach for accurate identification of corresponding activities”. *Computers & Electrical Engineering*. 2019; 78: 218–229, <https://www.scopus.com/pages/publications/85069632031>. DOI: <https://doi.org/10.1016/j.compeleceng.2019.07.011>.
10. Schoknecht, A., Thaler, T., Fettke, P., Oberweis, A. & Laue, R. “Similarity of business process models – A state-of-the-art analysis”. *ACM Computing Surveys*. 2017; 50 (4): 52. DOI: <https://doi.org/10.1145/3092694>.
11. Micallef, A. & Newton, P. M. “The use of concrete examples enhances the learning of abstract concepts: A replication study”. *Teaching of Psychology*. 2024; 51 (1): 22–29, <https://www.scopus.com/pages/publications/85123411072>. DOI: <https://doi.org/10.1177/00986283211058069>.
12. Krathwohl, D. R. “A revision of bloom's taxonomy: An overview”. *Theory Into Practice*. 2002; 41 (4): 212–218, <https://www.scopus.com/pages/publications/0036762146>. DOI: https://doi.org/10.1207/S15430421TIP4104_2.
13. Masapanta-Carrión, S. & Velázquez-Iturbide, J. Á. “A systematic review of the use of Bloom's taxonomy in computer science education”. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE 2018)*. New York: ACM. 2018. p. 441–446, <https://www.scopus.com/pages/publications/77953509711>. DOI: <https://doi.org/10.1145/3159450.3159491>.
14. “The Unix Shell: Summary of Basic Commands”. *The Carpentries*. 2025. – Available from: <https://carpentry.library.ucsb.edu/shell-novice/reference.html>. [Accessed: 20.09.2026].
15. Szlenk, M. “Formal semantics and reasoning about UML class diagram”. In *2006 International Conference on Dependability of Computer Systems*. Szklarska Poreba, Poland. 2006. p. 51–59, <https://www.scopus.com/pages/publications/34848917548>. DOI: <https://doi.org/10.1109/DEPCOS-RELCOMEX.2006.27>.
16. “PlantUML: Open-source tool that uses simple textual descriptions to draw UML diagrams”. *PlantUML*. 2026. – Available from: <https://plantuml.com>. – [Accessed: 20.06.2026].
17. “Bash2Domain”. *GitHub*. 2026. – Available from: <https://github.com/SW-OS3/OS4SWE/tree/main/Bash2Domain>. – [Accessed: 20.06.2026].

Отримано 04.09..2026

Отримано після перегляду 08.10.2026

Прийнято 13.10.2026

DOI: <https://doi.org/10.15276/ict.03.2026.48>

UDC 004.451:378.147

Domain-oriented semantic representation of Bash file operations in Operating Systems education

Oleksandr A. Blazhko¹⁾

ORCID: <https://orcid.org/0000-0001-7413-275X>; blazhko@op.edu.ua. Scopus Author ID: 57195410976

Olexander V. Zadereyko²⁾

ORCID: <https://orcid.org/0000-0003-0497-9861>; zadereyko@onua.edu.ua. Scopus Author ID: 57196005917

Volodymyr I. Hura²⁾

ORCID: <https://orcid.org/0000-0001-6415-7865>; hura@onua.edu.ua

¹⁾ Odesa Polytechnic National University, 1, Shevchenko Av. Odesa, 65044, Ukraine

²⁾ National university "Odesa Law Academy", 23, Fontanska doroha. Odesa, 65009, Ukraine

ABSTRACT

This paper presents a methodology for the domain-oriented semantic representation of basic Bash shell file operations designed for the "Operating Systems" course. Its relevance stems from the need to build explicit connections in students' understanding between domain-specific actions and system file operations. The methodology encompasses seven stages: defining the basic set of Bash commands, template-based structuring of the domain description, selecting a domain according to specified criteria, identifying domain analogues of basic file system objects (role class, container class, part class), establishing operational correspondences, technical implementation via Bash aliases, and aligning the process with Revised Bloom's taxonomy. Domain selection criteria are proposed, including a two-level "container–element" structure, the feasibility of consistent "task–action" mapping, and functional completeness. Algorithms for identifying domain analogues and operational correspondences have been developed to map Bash commands to domain actions of the role class. Technical implementation via aliases preserves the functionality of basic commands without creating new file system operations. For the stages involving object analogue identification, operational correspondence mapping, and alias implementation, alignment with the knowledge dimensions (conceptual, procedural, metacognitive) and cognitive process dimensions (Understand, Apply, Analyze, Evaluate, Create) of Revised Bloom's taxonomy has been established. This enables these stages to serve as a framework for developing methodological guidelines for students completing tasks across various cognitive levels. The practical significance lies in designing adaptable educational tasks based on semantic correspondences. Future research will validate the methodology experimentally, expand the command set and domains, and enhance the technical implementation through Bash functions.

Keywords: Operating systems; Bash shell; domain-oriented semantic representation; Revised Bloom's taxonomy; Bash aliases; command-line interface; teaching methodology

ІНФОРМАЦІЯ ПРО АВТОРІВ



Блажко Олександр Анатолійович - кандидат техн. наук, доцент кафедри Інженерії програмного забезпечення. Національний університет «Одеська політехніка», пр. Шевченка, 1. Одеса, 65044, Україна
Галузь досліджень: програмна інженерія

Oleksandr A. Blazhko - PhD, Associate Professor, Department of Software Engineering, Odesa Polytechnic National University, 1, Shevchenko Av. Odesa, 65044, Ukraine

ORCID: <https://orcid.org/0000-0001-7413-275X>; blazhko@op.edu.ua. Scopus Author ID: 57195410976

Research field: software engineering



Задерейко Олександр Владиславович - кандидат техн. наук, доцент кафедри Штучного інтелекту та математичного моделювання. Національний університет «Одеська юридична академія», Фонтанська дорога, 23. Одеса, 65009, Україна
Галузь досліджень: системний аналіз, аналіз та захист цифрових систем, штучний інтелект

Olexander V. Zadereyko - PhD, Associate Professor, Department of Artificial Intelligence and Mathematical Modeling, National university "Odesa Law Academy", 23, Fontanska doroha. Odesa, 65009, Ukraine

ORCID: <https://orcid.org/0000-0003-0497-9861>; zadereyko@onua.edu.ua. Scopus Author ID: 57196005917

Research field: systems analysis, analysis and security of digital systems, artificial intelligence



Гура Володимир Ігорович - кандидат техн. наук, доцент кафедри Штучного інтелекту та математичного моделювання. Національний університет «Одеська юридична академія», Фонтанська дорога, 23. Одеса, 65009, Україна
Галузь досліджень: програмна інженерія, мережеві технології, штучний інтелект

Volodymyr I. Hura - PhD, Associate Professor, Department of Artificial Intelligence and Mathematical Modeling, National university "Odesa Law Academy", 23, Fontanska doroha. Odesa, 65009, Ukraine

ORCID: <https://orcid.org/0000-0001-6415-7865>; hura@onua.edu.ua

Research field: software engineering, networking technologies, artificial intelligence