ОНТОЛОГІЧНІ МОДЕЛІ ОПИСУ ГОТОВИХ РЕСУРСІВ У РОЗРОБЦІ ПРОГРАМ.

Бабенко Л.П., Поляничко С.Л.

Інститут програмних систем НАН України
03187 м.Київ, пролсп.академіка Глушкова 40,
тел. 266-30-98

Досліджено роль домену програмної інженерії у системі концентрації знань з інфраструктури розробки програмних систем. Визначені базові засади побудови інформаційних моделей готових ресурсів розробки програмних систем як складових інфраструктури розробки, особливості онтологічних залежностей для домену програмної інженерії та запропоновано склад значущих для даного домену структурованих та онтологічних аспектів, запропоновано підхід до вирішення проблеми синонімії при побудові онтологій засобами спеціальної служби

The role of Software Engineering Domain in Knowledges Concentration systems is investigated. Main decisions about reusing program recourses modelling are discussed and special dependencies in ontological descriptions of such models are proposed. Also some decision with synonymy are proposed

1 Вступ

Характерною ознакою сучасного світу стає тенденція до глобалізації. Цей термін, який можна до певної міри трактувати як всесвітню інтеграцію зусиль у певній сфері людської діяльності, постійно присутній у продуктах масс-медіа, а наслідки позначеної ним тенденції відчуваються майже в усіх напрямках розвитку суспільства – у економіці, промисловості, культурі тощо. Не обминула тенденція до глобалізації і більшість областей науки. Інтеграція у науці завжди була її рушійною силою, але наразі мова йде не стільки і не тільки про обмін думками та спільні розробки, але про систематичну керовану діяльність у певній галузі науки, націлену на фіксацію світового рівня досягнень і регулярне відслідковування відповідних поповнень. Продукти такої діяльності знайшли свою матеріалізацію у побудові так званих ядер знань (англ. Body of Knowledges – скорочено BOK). Зокрема, широко відомим серед професіоналів є ядро знань з програмної інженерії – Software Engineering Body of Knowledges (SWEBOK) [1].

З появою SWEBOK створено небачені досі можливості для систематизації усіх процесів програмної інженерії: Нагальна потреба у такій систематизації є загально визнаною, адже накопичені комп'ютеризовані артефакти для різноманітних проблемних галузей (готові компоненти програм, методики, нормативні документи, типові проектні рішення тощо) досягли такого обсягу та розмаїття, що складність проблеми інформування про їх існування та властиві їм ознаки може бути співставлена зі складністю їх створення.

SWEBOK надає структуру упорядкування артефактів програмної інженерії і є базою, узгодженою професійною спільнотою для побудови відповідних онтологій в рамках проблемної області (Про). Слід сказати, що будь-яка BOK відіграє таку саму роль для своєї Про , тому дослідження методів використання SWEBOK при створенні програмних систем можна вважати пілотним дослідженням для інших BOK.

Метою даної доповіді є дослідження концепції, методів та засобів використання SWEBOK у побудові онтологічних моделей для інформаційної підтримки розробки програм, зокрема, використання готових рішень розробки (ГОР).

Дослідження виконане в рамках державної програми фундаментальних досліджень Національної академії наук України.

2 Підхід до побудови інформаційних моделей готових рішень

Інформаційною моделлю готового рішення будемо називати його опис, орієнтований на розуміння розв'язуваної ним проблеми, пошук необхідного рішення серед йому подібних і зіставлення з його конкуруючими варіантами. Таким чином, інформаційна модель є концептуальною. Проблема, яку моделюють, як правило, формулюється в просторі певної предметної області (домену) або на перехресті декількох доменів.

В основі запропонованої авторами інформаційної моделі ГОР лежить декілька суттєвих процесів. Це наступні:

1. Трансформація шляхом узагальнення формалізованих вимог на розробку окремих програм у концептуальну модель відповідної предметної області. На сьогодні існують засоби зрозумілого представлення вимог на розробку програмних систем, які належать до певних доменів, тож ці вимоги можна розглядати як джерела кооперативної, створеної колом професіоналів домену, бази знань домену.

Виходячи з об`єктно орієнтованої парадигми, домен може розглядатися як деякий суперклас, а віднесені до нього прикладні системи і компоненти - як підкласи, що його успадковують. Відповідно, знання про домен в цілому також можуть бути представлені як суперклас, що успадковується під час розгляду знань про кожну конкретну систему чи компоненту, включену у домен. Тоді продуктом аналізу домена є повторно використовувані знання про предметну область, що представляють собою узагальнення знань про конкретні ГОР, які віднесені до домену Побудова інформаційної моделі кожного ГОР - його атестація - може

розглядатися як конкретизація суперкласу "*база знань домену*, а інформаційна модель ГОР - як пошуковий образ останнього, націлений на пошук компонентів, адекватних новим розробкам [2] .

2. Аспектний розгляд моделей. Складність сучасних ГОР вимагає певних прийомів її переборення. Одним з таких прийомів, на наш погляд досить ефективним, є декомпозиція розгляду суцільної проблеми на окремі аспекти, тобто з позицій окремих точок зору або окремих носіїв інтересів. Прикладами для аспектів, широко застосовуваних при розгляді ГОР, є їх функціональні властивості, виконавчі показники, паспортні дані, візуальні чи аудіо засоби представлення даних. Аспектне представлення ГОР дозволяє побудувати для кожного аспекту свою систему навігації при інформаційному пошуку потрібного ГОР. При цьому доцільно використовувати два типа інформаційних моделей – структуровані та онтологічні.

Структуровані інформаційні моделі представлені у вигляді визначеної упорядкованої послідовності ключових ознак, значення яких належіть до визначених типів даних, як наприклад, ціле, текст, піктограма, одне або декілька з наперед визначених значень, тощо. Інформаційний пошук в структурованому середовищі даних може бути реалізований засобами СУБД та потребує менших ресурсів (часу, пам'яті) в порівнянні з іншими. Прикладами аспектів, для яких достатньо структурованої моделі, є паспортні дані (ім'я та адреса розробника, спосіб придбання, ціна, тощо), середовище реалізації (операційна система, СУБД, тощо), потрібні апаратні ресурси.

Онтологічні моделі застосовуються для опису функціональних особливостей ГОР і дозволяють відобразити складні відношення між окремими характеристиками. Цей тип моделей базується на використанні онтологій [3-5].

Розбудова онтологій з різних проблемних областей, створення інтегрованих міжнародних бібліотек онтологій є відображенням загальних процесів глобалізації. Підтвердженням цьому є той факт, що на сьогодні WEB-запит по ключовому слову "онтологія " видає 64 тис. сторінок відповідей [3].

Онтологія відіграє роль дерева знань певної проблемної галузі. У самому загальному випадку, онтологія – це угода про спільне використання понять, що включає засоби представлення предметних знань і домовленості про методи розмірковувань. Звичайно, онтології представляються семантичними мережами, у яких вузлами є поняття , а дугами - зв'язки чи відносини, асоціації між ними. Онтологія, *або* понятійна база, визначає не тільки узгоджену, уніфіковану термінологію, якою повинні користуватися різні носії інтересів, що приймають участь у процесі аналізу вимог, але також істотні відносини між поняттями і їх інтерпретацію. У нашому випадку під інтерпретацією понять або відношень будемо розуміти їх неформальні визначення або коментарі та засоби вирішення проблеми синонімії. Якщо взаємодія інформаційної системи з користувачем відбувається під керуванням онтології, це суттєво підвищує рівень взаєморозуміння взаємодіючих сторін, бо усувається ефект неузгоджених термінів, синонімії та антонімії, а формулювання користувачем запиту на пошук зводиться до вибору запропонованих системою можливостей. Однак, треба сказати, що цей тезис є справедливим тільки за умови, якщо онтологія адекватно відображує проблемну область, у якій працює користувач, і сприймається ним з розумінням. Щоб задовольнити таку умову, треба вкласти певні ресурси у створення та супроводження онтології, залучити високо кваліфікованих експертів проблемної галузі та програмної інженерії. Слід зазначити, що для відносно нових і досить складних областей застосування комп'ютерних систем онтологія може не відбивати повний спектр можливостей і має потребу в постійному відстеженні змін.

3. Упорядкування онтологій окремих аспектів уніфікованим набором класифікаторів вищого рівня. Стратегія їх вибору була обумовлена наступним.

Серед аспектів, представлених онтологічними моделями, будемо виділяти аспекти домену, які відображають специфіку певної проблемної області, та аспекти домену програмної інженерії, які відображають особливості категорії ГОР в термінах базових понять програмної інженерії. Такими аспектами будемо вважати структурний, поведінковий, інтерфейсний та варіаційний аспекти.

Визнаним стандартом представлення таких аспектів є універсальна мова моделювання UML, яка набула широкого поширення. Тож в якості мета класифікаторів зазначених аспектів доцільно обрати мета класифікатори UML. Ця ідея обіцяє бути особливо продуктивної для повторно використовуваних компонентів, більшість яких нині розробляється за технологією UML і їх специфікація може слугувати як джерело знань для відповідної онтології.

Відповідно до сказаного вище, у якості класифікаторів структурного аспекту будемо використовувати мета класифікатори діаграм класів і варіантів використання, для поведінкового аспекту – мета класифікатори методів класів (описаних у діаграмах класів), станів і дій, взаємодії об'єктів (у діаграмах послідовності і кооперації).

Інтерфейсний аспект відображається в онтології на перелік сервісів, які може надавати ГОР, вхідні та вихідні параметри сервісів та їх типи, перед- та пост-умови функціонування ГОР, та на перелік інших ГОР, сервісів яких він потребує для здійснення своїх сервісів. Класифікатори наведених вище аспектів пропонується використовувати як вищезазначені уніфіковані класифікатори вищого рівня.

Варіаційний аспект дозволяє виразити прогнозовану варіантність для більшості використовуваних у діаграмах UML типів елементів моделювання. У [6] проведене дослідження способів специфікації прогнозованої варіантності програмних проектів засобами UML. Атестація адаптивних властивостей ГОР формується відповідно з рекомендаціями, наведеними у [6].

4. Сегментація репозиторію на розділи, відповідні окремим аспектам певних доменів. Систему зберігання, пошуку та співставлення ГОР ми будемо в подальшому називати репозиторієм. Репозиторій як загальне сховище ГОР вважається віртуально поділеним на розділи, кожен з яких може бути віднесений до певної проблемної області, тож поточний перелік проблемних областей, представлених в репозиторії, є його класифікатором першого рівня. Класифікаторами наступних рівнів можуть слугувати аспекти, визначені для проблемних областей. Кожному аспекту відповідає своя інформаційна модель, яка належить до структурованого типу або онтології.

5. Створення фасетної структури пошукового образу ГОР. Аспектний підхід до інформаційного моделювання відомий в інформаційних системах як фасетний, тож кожному виділеному аспекту буде відповідати онтологія у репозиторію, своя фасета пошукового образа окремого ГОР, свій фрагмент пошукового запиту на необхідні ГОР. При цьому вказані у конкретній фасеті конкретного пошукового образу (або запиту, або довіднику репозиторія) дескриптори належать до концептів або відношень онтології , відповідної аспекту даної фасети, тобто в термінах концептів інформаційної моделі представляються пошуковий образ ГОР і пошуковий запит. Передбачається, що формування двох останніх буде відбуватися під керуванням бази знань домена. Таким чином, три складові інформаційної середовища повторного використання, а саме база знань домена, колекції пошукових образів компонент і запити на пошук компонентів представлені в єдиному лексичному та семантичному просторі.

6. Суміщення принципів автономії та узгодження. Передбачається кожну з тих онтологій, що є доступною у репозиторії, створювати і супроводжувати автономно. В той же час між зазначеними фрагментами для різних за типом артефактів можуть бути встановлені певні залежності, як-от: <<реалізує >>, <<є поясненням>>, <<є стандартом в частині визначення термінів>> тощо.

7. Вирішення проблеми синонімії. Пропонується спеціальна служба: для усіх лексичних одиниць, з якими оперують бази знань даного проекту (представлення імен доменів, аспектів , концептів, типів значень концептів, відношень та стереотипів відношень тощо), яка встановлює (силами експертів доменів або адміністраторами бази знань системи) наявні відношення синонімії, які фіксуються у спеціальному словнику, який ми будемо називати словником синонімів. Цей словник фіксує гнізда синонімів, при чому один з термінів, що складають гніздо, обирається канонічною формою терміну. Саме канонічна форма фігурує в базі знань даного проекту і, відповідно, в пошукових образах і запитах. Передбачається, що служба синонімів має сервіси надання канонічної форми для вказаного терміну, поповнення гнізда синонімів за запитом адміністратора бази знань , надання для канонічного терміну дефініції або коментарів, якщо вони передбачені у проекті. Кооперативні домовленості по узгодженню дескрипторів та асоціацій між ними планується здійснювати через службу ведення синонімів, яка є повторно використовуваним компонентом ведення усіх онтологій, а її база знань використовується як кооперативна. Усі зміни у онтологіях провадяться через службу синонімів і тільки вона потребує разових узгоджень між експертами, відповідальними за різні онтології.

3 Проблеми створення онтологій ГОР

Опис онтології певного аспекту є специфікацією його концептів, відношень та правил інтерпретації.

Суттєвою рисою даної розробки є використання спільної онтології для чотирьох рознесених у часі, але концептуально узгоджених між собою процесів, що оперують спільними даними, а саме:

- побудови і модифікації онтології як бази знань ПрО ,
- побудови пошукового образу ,
- побудови пошукового запиту ,
- формулювання вимог на новий програмний компонент

Усі вони мають подібний синтаксис та реалізуються за допомогою подібних інтерфейсів. База знань репозиторію є повторно використовуваний ресурс для трьох інших процесів: створення пошукових образів об`єктів зберігання репозиторію, формулювання користувачем репозиторію своїх інформаційних потреб як пошукового запиту, формулювання розробником вимог до нової системи. Пошуковий образ ресурсу індексується репозиторієм та заноситься у репозиторій для подальшого зберігання. В процесі зміни онтології (наприклад, видалення вершини) проводиться його переіндексування. Створений пошуковий запит дозволяє запустити процес пошуку потрібного ГОР в репозиторії, при цьому можна вказати критерій релевантності, як це було запропоновано в [7].

Останнім часом було проведено цілий ряд інтенсивних досліджень по використанню онтологій засобами постачання Інтернет-ресурсів. Результатами цих досліджень стала поява ряду пропозицій від потужних професійних об`єднань провідних розробників щодо представлення онтологій [8] та специфікації онтологічних залежностей повторно використовуваних ресурсів, зокрема, так званих веб-сервісів [9-11]. Головною метою цих пропозицій була орієнтація на переносність представлень онтологій, тобто їх незалежність від платформи, на якій створена або використовується онтологія. Тому базовими мовними засобами для опису та представлення онтологічних залежностей у згаданих пропозиціях (які, до речі, претендують на роль універсальних стандартів) є ті чи інші модифікації мови XML [12], яка набула в останні роки широкого розповсюдження. Одначе платою за сумісність та універсальність, як відомо, майже завжди є незручність та надлишковість.

Для всіх згаданих вище мовних засобів, на нашу думку, характерною властивістю є те, що вони досить “недружні” і до того, хто описує залежності між концептами онтології, і до тих, хто намагається візуалізувати згадані залежності або провести за ними інформаційний пошук

Життєвий цикл онтології (для обраної предметної області) може бути представлений наступними етапами:

- первинне накопичення загальновідомих понять предметної області як тлумачного словника останньої;
- встановлення відношень між накопиченими поняттями;
- накопичення розробок окремих програмних систем або інших ГОР для обраної предметної області;
- виявлення спільності накопичених розробок, узгодження їх зі тлумачним словником предметної області і створення концептуальної моделі домена як онтології домена;
- представлення вимог до нової розробки під керуванням онтології домена;
- модифікація онтології і концептуальної моделі домена відсутніми для нової розробки можливостями;
- побудова пошукового образа нового ГОР як конкретизації онтології домена.

Ефективність використання онтологічної системи багатоаспектного представлення знань щодо готових програмних рішень залежіть від зручного інтерфейсу, який підтримує керований доступ до всіх функціональних частин онтологічної системи. Основною частиною такого інтерфейсу є графічний редактор аспектів онтології.

Такий інтерфейс дозволяє розподілити у часі роботу з різними частинами онтології. Редактор онтологій дозволяє візуальне проектування аспектів, при цьому можна в зручній формі створювати і редагувати різні аспекти представлення програмних ресурсів, концепти та відношення між ними.

До інтерфейсних об’єктів онтологічної системи можна віднести панель створення та модифікації онтології, панель атестації ГОР, панель побудови запитів, панель текстового браузінгу онтологій, панель звітів, панель представлення вихідних даних про ГОР, панель роботи зі словником онтології.

Кожний ГОР представлено в репозиторії у двох взаємопов’язаних складових –представлення коду або тіла ГОР та його пошукового образу (ПОШ), який є інформаційною моделлю ГОР. Кожна з цих складових зберігається у відповідному сегменті репозиторію під ім’ям ГОР. Будемо називати їх базою ПОБ та базою ГОР.

Створення ПОБ починається зі встановлення імені ГОР, під яким ПОБ буде зберігатися в репозиторії. Після цього відкривається аспект домену, який приймає участь в атестації ГОР. Фрагмент ПОБ ГОР, який відповідає окремому аспекту онтології домену, будемо називати фасетою ПОБ для ГОР. Побудова фасет ПОБ проводиться поаспектно.

Для побудови кожної фасети проводиться:

- вибір аспекту,
- відкриття онтології, відповідної обраному аспекту,
- активація (відмічання в онтології аспекту) концептів та відносин, які передбачається занести до ПОБ ГОР ,
- занесення помічених концептів та відношень до поточної фасети ПОБ,
- кожний етап формування образу зберігається в пам’яті ,
- можливе повернення до попередніх ітерацій формування образу у випадку невдалої атестації,
- занесення побудованої фасети до складу ПОБ поточного аспекту,
- занесення побудованого ПОБ під визначеним ім’ям до бази ПОБ репозиторію,
- занесення тіла ГОР до бази ГОР репозиторію,
- встановлення посилань між ПОБ ГОР та тілом ГОР.

Пошуковий образ може супроводжуватися поясненням у вигляді неформалізованих коментарів, діаграм UML, RDF схем його опису та ін.

Кожна онтологія пов’язана з певними аспектом певного домену. Передбачені спеціальні засоби для заснування нового домену в системі.

Перед створенням нової онтології потрібно вибрати з запропонованого системою списку існуючих доменів той, в рамках якого буде існувати нова онтологія. У випадку, коли в репозиторії не створено домену, який охоплює предметну область нової онтології, необхідно виконати дії по створенню нового домену. Застарілий домен може бути видалений з системи, за умови правильно введеного паролю на видалення домену, разом зі всіма онтологіями, які на даний момент в ньому були напрацьовані.

Кожна онтологія вважається пов’язаною з окремим аспектом певного домену. Посилання на онтологію представлено як <ім’я домену> <ім’я аспекту>.

При модифікації онтології в даному проекті прийняті такі правила:

- зміна назви аспекту веде до перегляду та зміни посилань на відповідні онтології усіх аспектів репозиторію,
- зміна назви концепту веде до перегляду і зміни посилань на адресати (об’єкти) відношень, у яких приймає участь даний концепт,
- вилучення певного концепту веде до перегляду усіх концептів, з якими пов’язаний той, що вилучається. При цьому в залежності від типу відносин зв’язку залежні концепти або вилучаються (як наприклад, у випадку зв’язку узагальнення) або модифікуються під керуванням аналіста предметної області,

- додавання нового концепту або встановлення нових зв'язків не міняє існуючих.
- При визначенні пошукового образу в даному проекті дотримуються наступних правил:
- пошуковий образ є конкатенацією своїх фасет,
- фасета пошукового образу є конкатенацією своїх складових,
- складова фасет пошукового образу є виразом для одного або декількох концептів відповідного аспекту,
- для структурованого аспекту складова фасети є виразом
- *концепт <відношення> значення*,
- де відношення – це одне з відношень *"точно", "рівно", "більше", "менше", "не більше", "не менше"*,
- для онтологічного аспекту складова фасети є назвою концепту або трійкою *концепт-об'єкт <стереотип відношення> концепт–суб'єкт*.

4 Засоби нормалізації лексики в інформаційних моделях. Підхід до вирішення синонімії

Словник синонімів є засобом автоматичної підтримки онтології і може виконувати наступні функції:

- тлумачний словник,
- навігація по аспектах,
- побудова синонімії,
- включення нових термінів у словник, підключення додаткових термінів у домен (з метою навчання).

Словник є зібранням канонічних імен концептів, їх синонімів, дефініцій та ознак приналежності до аспектів в домені. Основними функціями словника в онтологічній системі для нормування лексики можна вважати підтримку взаємних посилань між наступними сторонами:

- між канонічним ім'ям і множиною його синонімів,
- між канонічним ім'ям концепту та ім'ям аспекту, до онтології якого він включений,
- між канонічним ім'ям концепту та його дефініцією.

Якщо відмітка належності концепту до певного аспекту відсутня, будемо називати його вільним.

При введенні нового імені концепту до онтології певного аспекту проводиться перевірка на його входження до словника і, в разі знаходження відповідного концепту, видається повідомлення про те, що він є канонічним ім'ям знаного в онтології концепту або його синонімом. Користувач має можливість змінити введене канонічне ім'я концепту на нове, або на його синонім. Якщо вказане ім'я не відоме системі, експерт повинен переглянути визначені ним фрагменти онтології і прийняти рішення, що нове ім'я є канонічним або синонімом відомого. Для кожного з зазначених випадків є відповідні засоби підтримки в системі.

Передбачаються наступні функції використання словника:

- визначення аспекту, до якого належіть канонічне ім'я,
- візуалізація контексту певного концепту в онтології аспекту,
- введення нового канонічного імені концепту або відношення до онтології аспекту,
- введення синоніму до канонічного імені,
- введення дефініції концепту або відношення, яка передбачає введення текстового коментаря щодо значення змісту терміну,
- сортування канонічних імен за належністю до певних аспектів,
- сортування канонічних імен концептів або відношень за алфавітом,
- зміна канонічного імені концепту на синонім, що має наслідком автоматичну зміну у представленні онтології відповідного аспекту та відповідні зміни в пошукових образах ГОР,
- пошук канонічного імені концепту або синоніма з метою подальшого браузінгу онтології з місця локалізації знайденого концепту,
- корекція лексики вказаних у словнику елементів.

5 Суміжні розробки

Дослідження засобів опису та класифікації ГОР в сучасних розробках займають чинне місце у публікаціях останніх років.

ГОР може бути описаний, як характеристики його частин, до яких найчастіше відносять: інтерфейс (метод, атрибут, виключення), типи даних, паспортні дані. Таким чином представляється класифікаційна абстрактна схема компонента.

Опис компонента може бути зроблено на різних рівнях абстракції через ієрархічну структуру. Такий метод описаний в [14]. В цій структурі ГОР представляє найвищий рівень абстракції, в той час як інтерфейси, методи, атрибути, виключення та типи даних представляють нижчі рівні деталізації. Ієрархічні класифікатори не тільки дозволяють ідентифікувати компонент, але і підтримують відповідні пошукові образи компонент та його частин, які можуть надалі використовуватися у відповідності до вимог нової прикладної програми. При пошуку адекватного цілям користувача компонента, структуровані пошукові запити дають можливість користувачу значно скоротити обсяг огляду потенціально підходящих компонентів до найбільш відповідних вимогам, в той час, як напів-структуровані пошукові запити дають можливість аналізувати меншу кількість компонентів з метою визначення їх відповідності наперед визначеними вимогами.

Зараз з'явилося багато підходів, які спираються на опис сервісів компонентів. Зміст цих підходів полягає в тому, що опис компонента базується на описі сервісів, які пропонує компонент, та ідентифікується своєю адресою в мережі (URI адреса). В цьому підході кожний сервіс компонента представляє деякі функціональні можливості, які підтримуються компонентом та описуються в термінах комерційної задачі, якій прислуговує компонент у своєму домені. В цей опис може бути підключено опис модуля перемов про використання компонента. Крім того, опис сервісів представляє опис правил доступу до компонента, що також є характеристикою компонента, яку може бути підключено до його класифікаційної схеми. Саме на цю модель представлення знань про ГОР спираються відомі зараз мови WSDL [10][11], Semantic WEB .

Деякі класифікаційні схеми спираються на опис правил поведінки та ролей компонента в системі. Prieto-Diaz в своїй роботі [15] вважає, що компонента повинна описуватися в термінах її призначення при використанні, тобто в термінах ролі. Але можна представляти можливу роль компонента в системі. Роль може слугувати як ефективний шлях пошуку відповідного компонента.

Не чіткі характеристики опису області та середовища виконання компонента також можуть слугувати для аспектного опису компонента.

Широко вживана класифікація компонентів за їх ім'ям. Семантично відповідна функціональності назва в багатьох системах вважається універсальним ідентифікатором компонента повторного використання. Наприклад, назва "поповнення кредитної карточки" може використовуватися для атестації такого компонента. Оскільки можуть зустрічатися ГОР з однаковими назвами, кожний компонент потребує також унікального ідентифікатора. Окрім цього можливе використання спеціального протоколу кодування та підключення ГОР в систему.

Зараз вже існують чимало широко вживаних таксономій . Наприклад, International Computer Programs (Міжнародні комп'ютерні програми) [15] розподіляють програмне забезпечення на три загальні категорії: програмне забезпечення систем (наприклад, транслятор), загальні прикладні програми (наприклад, зв'язок), та виробництво (наприклад, резервування авіа лінії). Зрозуміло, що кожна з приведених категорій може бути розподілена далі. Так, загальні прикладні програми можуть бути далі класифіковані в більш конкретні категорії: GUI, генераториповідомлень, корелятори друку тощо.

На даному етапі відомо чимало стандартних класифікаторів. Секрет їх успіху полягає в доборі зручних категорій класифікації для комерційних ГОР. До мов, які спираються на зареєстровані таксономії, відноситься мова UDDI [10]. Її класифікація спирається на три базових таксономії. Одна з них - це класифікація послуг та опис продуктів. Також використовуються таксономія ідентифікаторів географічних назв та Північно-Американські класифікація промисловості.

Щоб керувати компонентами ефективно, потрібна доповнююча паспортна інформація. Наприклад, Буртон та ін. [13] характеризували традиційні компоненти у відповідності з їх автором, назвою та номером версії. Що стосується комерційних ГОР, то ім'я продавця, посередника, та версії компонента можуть бути корисні в характеристиці компонента. До цього списку додаються доповнюючи елементи класифікації.

Суттєвою характеристикою ГОР є середовище виконання, а саме: операційна система та мова програмування, які використовуються в реалізації ГОР. Доповнюючи характеристики можуть стосуватися середовища розгортання (типа .NET, EJB, и CORBA). Вищезгадані ідентифікатори представляють структуровану інформацію.

Дескрипторні онтологічні аспекти дозволяють всебічно представити потрібну інформацію в ієрархічному вигляді для різних типів ГОР для подальшого її використання різними користувачами. В ідентифікуванні дескрипторних аспектів, ми досліджували базові характеристики компонентів, які допомагали би користувачам в їх пошуку під час пошукової сесії в репозиторії онтологічної системи.

В [16] пропонується оригінальний підхід до атестації ГОР. Суть цього методу полягає в тому, що будується дескриптор ний опис ГОР. При цьому дескриптори вибираються як найбільш вживані та семантично значущі слова з коментарів у програмному коді ГОР. Запит на пошук потрібної ГОР реалізується також за допомогою конструювання дескрипторного образу із наперед сформованого словника значущих слів коментарів.

В досліджених онтологічних моделях інформація про ГОР найчастіше кодувалася за допомогою XML [12] (або на мові на базі логіки та правил).

При такому кодуванні кожний запит по онтології може містити одне або більше ключових термінів або стислий описувальний текст, який може приймати значення NULL. В онтологічному представленні мова типу XML ефективно підтримує схему кодування на базі напів структурованої інформації про ГОР та підтримує контекстний пошук. Використовуючи визначення типів на базі XML документів (DATA), щоб визначити структуру класифікаційної схеми, кодовані компоненти можуть бути погоджені та відповідати побудованій схемі класифікації. Крім того, XML інструментальні засоби аналізу забезпечують здатність проводити запити на пошук ГОР.

6 Заключення

Інтеграція у єдиній інформаційній моделі домену проблемної області та домену програмної інженерії дозволить створити єдине понятійне середовище як для специфікації постановок задач на розробку програмних

систем у рамках визначеної предметної області, так і для пошуку придатних для використання у розробках знань, у тому числі методик, інструментів, стандартів, готових проектних рішень і компонентів, що відносяться до різних етапів життєвого циклу розробки.

Література

1. http:/www.swebok.org.html
2. *Бабенко Л.П.* Информационная поддержка повторного использования в программной инженерии на базе UML//Кибернетика и системный анализ.-2003.- N1.-С.74-
3. *Gruninger M., Lee J.* Ontology Applications and Design // Communications of the ACM.-2002.-vol.45.-N 2.-P39-41.
4. *Holsapple C.W., Joshi K.D.* A Collaborative approach to Ontology Design// Communications of the ACM.-2002.-vol.45.-N 2.-P.42-47.
5. http://www_ksl.stanford.edu/knowledgesharing/ontologies/README.html
6. *Бабенко Л.П., Поляничко С.Л.* Подходы к аттестации адаптивных возможностей программных компонент //Проблемы программирования.–2001.–N 1-2.–с. 35-41
7. *Поляничко С.Л.* Предикати співставлення для порівняння та аналізу компонент //Проблеми програмування.–2002.–N 1-2.–с.117-120
8. OWL Web Ontology Language, http://www.w3.org
9. *Pahl C.,Casey M.* Ontology Support for Web Service Processes//ACM SIGSOFT Software Engineering Notes.- v.28.- №5.-2003.-P.208-216
10. *F.Curbera,M.Duftler et all.*Untravelling the Web Services Wed, An Introduction to SOAP, WSDL and UDDI//IEEE Internet Computing.-№2.-2002.-P.86-93
11. http://www.w3/org/TR/2003/WD.wsdl20.20031110
12. *Дейтел и др.* Как программировать на XML/М.-«Бином».-2001.-933с.
13. *A. Burton, R.W. Aragon, S.A. Bailey, K.D. Koehler, and L.A. Maves.* The Reusable Software Library// IEEE Software vol. 4, N.4, pp. 25-33, 1987.
14. *E. Damiani, M.G. Fugini, and C. Bellettini.* Corrigenda: A Hierarchy-Aware Approach to Faceted Classification of Object-Oriented Components// ACM Trans. Software Eng. and Methodology, vol, 8, N.3, pp. 425-472, 1999.
15. *R. Prieto-Diaz.* Implementing Faceted Classification for Software Reuse// Comm. ACM, vol. 34, N.5, pp. 89-97, 1991.
16. *Yunwen Ye , Gerhard Fischer.* Supporting Reuse by Delivering Task-Relevant and Personalized Information// Proceeding of 2002 International Conference on Software Engineering (ICSE'02), Orlando, FL, pp. 513-523, May 19-25, 2002.