004.942

А. Зелінський, В. Теслюк
Національний університет “Львівська політехніка”,
кафедра систем автоматизованого прогнозування

РОЗРОБЛЕННЯ АРХІТЕКТУРИ СИСТЕМИ СИНТЕЗУ МІКРОЕЛЕКТРОМЕХАНІЧНИХ СИСТЕМ

Стаття присвячена розробленню архітектури системи, що дає змогу розробнику автоматизувати процес структурного синтезу МЕМС, використовуючи ситуативно базовану аргументацію.

Ключові слова: МЕМС, САПР, ситуативно базована аргументація, архітектура, синтез, структурна схема.

This article is devoted to system architecture development, which solves MEMS structural synthesis tasks using case-based reasoning.

Key words: MEMS, CAD, case-based reasoning, architecture, synthesis, structural scheme.

Вступ

Сьогодні існує низка систем, що дають змогу розробнику з достатнім досвідом у галузі мікроелектромеханічних систем (МЕМС) проектувати МЕМС відповідно до поставленого завдання. В процесі проектування використовуються конструктивні елементи (блоки) МЕМС, з яких будується система [1]. Такі потужні САПР мають багато переваг, але є і недоліки. З погляду звичайного (не професійного) користувача недоліками таких систем є: їх значна ціна; необхідність у глибоких знаннях у галузі проектування МЕМС; закритість бази даних елементів та проектних рішень всередині системи та ін. [2].

Отже, метою цієї роботи є розроблення загальнодоступної системи автоматизованого проектування МЕМС, зрозумілої для проектантів з невеликим досвідом роботи у галузі МЕМС технологій. Перед початком розроблення такої системи слід було б знайти відповіді на декілька ключових запитань, таких як: де взяти базу структурних елементів МЕМС? як отримати і зберігати експертні знання щодо поєднання і побудови МЕМС? як зробити таку систему зрозумілою і доступною для користувачів з різним рівнем досвіду в галузі розроблення МЕМС[3]?

Для розв’язання цих задач запропоновано використати ситуативно базовану аргументацію(СБА) [4] у поєднанні з об’єктно-орієнтованою моделлю представлення знань про об’єкти МЕМС та їх поєднання. Стаття присвячена розробленню архітектури системи, яка грунтується на ситуативно-базованій аргументації та дає змогу розв’язувати поставлені перед проектантом задачі на основі наявних схемотехнічних рішень МЕМС.

Ситуативно базована аргументація

Ситуативно базована аргументація (СБА) – метод розв’язання нових задач на основі схожих проблем, розв’язаних у минулому. Процес розв’язання поставленої задачі методами СБА можна звести до таких основних кроків: 1.Пошук і отримання проектних рішень, що задовольняють поставлену задачу, в базі проектних рішень. 2. Розв’язання поставленої задачі за допомогою отриманих рішень. 3. За необхідності оптимізувати існуючі рішення (синтезувати нові на основі наявних) відповідно до вимог поставленої задачі. 4. Зберегти новостворені проектні рішення з метою забезпечення їх подальшого використання під час розв’язання задач[5].

На рис.1 зображено розроблену схему розв’язання задачі за допомогою методу, який ґрунтується на використанні СБА.

Використання СБА дає змогу не використовувати базу знань (як набір правил для формування рішення) і розв'язувати поточні задачі на основі наявного досвіду. Ми вдосконалили підхід до використання СБА, розширивши її використанням правил поєднання структурних елементів МЕМС. Такі правила дадуть змогу краще і швидше формувати можливі структурні схеми та мінімізують кількість можливих помилок у разі використання методів оптимізації на основі генетичних алгоритмів.

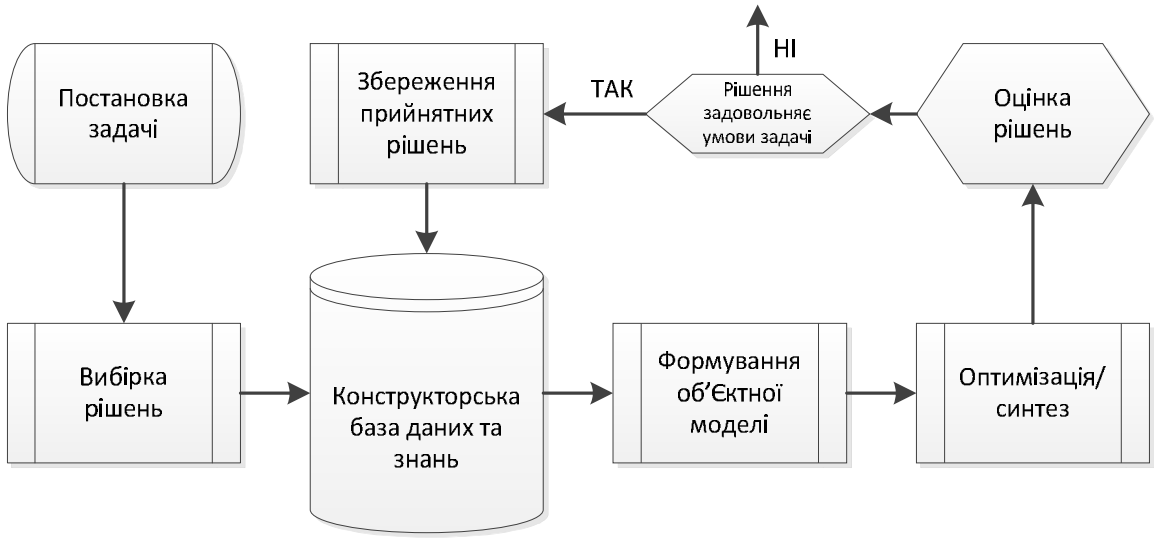

Рис.1. Схема розв'язання задачі методами ситуативно базованої аргументації

Вибір технології LINQ та її можливості застосування при проектуванні підсистеми управління КБД

Концептуальні моделі програм відрізняються від логічних моделей баз даних. Модель Entity Data Model(EDM) є концептуальною моделлю даних конкретного домена і забезпечує можливість програмної взаємодії з даними як з сутностями чи об'єктами [6].

На рис 2. зображено структурну схему LINQ to Entities.

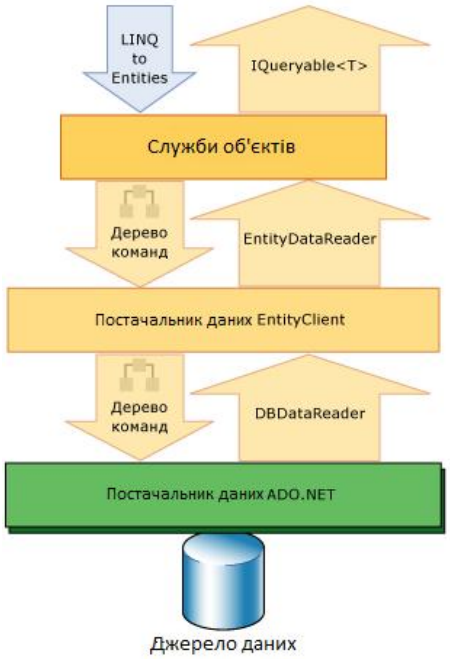

Рис. 2. Схема LINQ to Entities

За допомогою EDM можна представляти сутності у вигляді об'єктів. Тобто ця технологія надає розробнику можливість доступу до даних у базі за допомогою тієї самої мови програмування, яка використовувалась при написанні бізнес-логіки.

LINQ можна застосовувати з різними джерелами даних, зокрема і архітектури даних, які зберігаються в пам'яті, XML-документи, БД, моделі сутностей і набори даних.

Є п'ять реалізацій LINQ, які забезпечують доступ до даних, і одними з них є:

LINQ to SQL підтримує оперативну розробку програм, здатних комунікувати з будь-якими редакціями Microsoft SQL Server через програмні об'єкти, які безпосередньо відповідають об'єктам в базі даних, такими як таблиці, відображення, процедури і функції.

LINQ to Object дає змогу виконувати запити до об'єктів, які зберігаються в пам'яті [7]. Окрім того, засоби LINQ дають можливість працювати з даними в пам'яті як з даними з будь-якого іншого джерела. Отже, використовуючи LINQ для проектування бази знань ми зможемо досягти таких результатів: об'єкти і сутності з бази і пам'яті будуть представлятись одними і тими ж об'єктами, що дасть змогу маніпулювати ними, використовуючи єдину мову програмування. Це, своєю чергою, усуває необхідність розроблення модулів приведення одних об'єктів до типу інших об'єктів чи до універсального типу, з яким би працювала логіка БЗ.

Структурні одиниці системи

З визначення СБА зрозуміло, що основою системи, що ґрунтується на цьому методі, є ***конструкторська база даних(КБД)***, що містить проектні рішення. Така КБД забезпечить зберігання, швидкий доступ і вибірку таких сутностей, як: інформація про проблему, яка вирішується; опис рішення; опис елементів МЕМС; конструкторська документація, що стосується рішень та їх елементів; правила комбінування елементів для формування рішень[8].

Основою такої КБД є КБД елементів МЕМС, що містить повну інформацію про складові елементи МЕМС, їх характеристики та властивості, а також конструкторсько-технологічну документацію.

Для роботи з КБД розроблено спеціальну ***підсистему***, яка містить всю необхідну логіку ***програмної роботи з КБД***. Наявність такої підсистеми дає можливість розвантажити інші структурні одиниці системи і звільнити їх від зайвої логіки, що стосується роботи з даними. Кожна структурна одиниця має виконувати лише свої спеціалізовані функції. Для реалізації програмної логіки роботи з КБД використано технологію LINQ, яка дає можливість представляти об'єкти і сутності в базі і пам'яті одними і тими самими об'єктами, що дає змогу маніпулювати ними, використовуючи єдину мову програмування. Це, своєю чергою, усуває необхідність розроблення модулів приведення одних об'єктів до типу інших об'єктів чи до універсального типу, з яким би працювала логіка БЗ.

У початковому стані КБД є пустою – її потрібно наповнити наявними рішеннями МЕМС та даними про задачі, для розв'язання яких ці МЕМС можна використовувати. Така інформація має вводитись користувачами системи в процесі роботи з нею. Тому задача розробника полягає в реалізації максимально простого та зрозумілого інтерфейсу, який би дав змогу швидко та зручно наповнювати КБД новою інформацією. Своєю чергою, система має зробити введену інформацію доступною для всіх користувачів. У процесі синтезу нових проектних рішень на основі наявних система поповнюватиме КБД рішеннями, згенерованими в процесі розв'язання задач. За наповнення КБД відповідатиме ***підсистема наповнення КБД***.

Одну з найважливіших функцій в роботі системи, а саме вибірці рішень або конструктивних елементів МЕМС та перевірці на відповідність цих рішень поставленій задачі має відповідати ***підсистема вибірки даних***. Ця підсистема має реалізувати логіку оптимізації пошукової задачі, фільтрування та швидкого пошуку даних у КБД. Для реалізації такої підсистеми слід розробити унікальний ієрархічний алгоритм пошуку та вибірки даних з КБД. Такий алгоритм має відповідати таким критеріям: пошук в базі має бути максимально швидким; алгоритм пошуку має володіти достатньою точністю вибірки, для забезпечення вибірки оптимальної множини альтернативних проектних рішень і одночасного зменшення можливих втрат при вибірці рішень чи елементів;

потрібно врахувати темп та обсяг розростання КБД, що, своєю чергою, вимагає використання алгоритмів ієрархічного чи шаблонного пошуку.

Підсистема оптимізації та синтезу має реалізувати логіку оптимізації наявних рішень для поставленої задачі. Також у разі відсутності проектних рішень, придатних для розв'язання задачі, підсистема має "послабити" умови задачі з метою отримання вибірки рішень і їх подальшої оптимізації. "Послаблювати" умови задачі необхідно, враховуючи ваги вхідних параметрів задачі яку розв'язують. У випадку, якщо не вдалось отримати розв'язку задачі, використавши наявні рішення або якщо множина рішень була отримана після "послаблення" умов задачі, ***підсистема оптимізації та синтезу*** за допомогою альгоритмів штучного інтелекту та наявних правил поєднання компонентів МЕМС спробує згенерувати нові розв'язки, які б задовольняли умову поставленої задачі. Для генерації нових проектних рішень будуть використані генетичні алгоритми, які, опираючись на правила поєднання елементів МЕМС, мають згенерувати достатню кількість розв'язків задачі.

Для реалізації комунікації користувача з системою потрібно розробити ***графічний інтерфейс***, що дасть змогу проектанту поетапно відслідковувати процес розв'язання задачі і вносити свої корективи у розв'язок.

Проаналізувавши вищеописані підсистеми, було розроблено структуру системи синтезу МЕМС.

Проаналізувавши наявні елементи МЕМС та їхні специфікації, було прийнято рішення включити в опис елементів конструкторську документацію, креслення та моделі. Проаналізовано наявні формати для визначення універсального формату збереження конструкторської інформації про елементи МЕМС, який міг би спільно використовуватись різними САПР. Наразі для зберігання конструкторсько-технологічної документації використовується формат DWG, який забезпечує можливість інтеграції з іншими САПР.

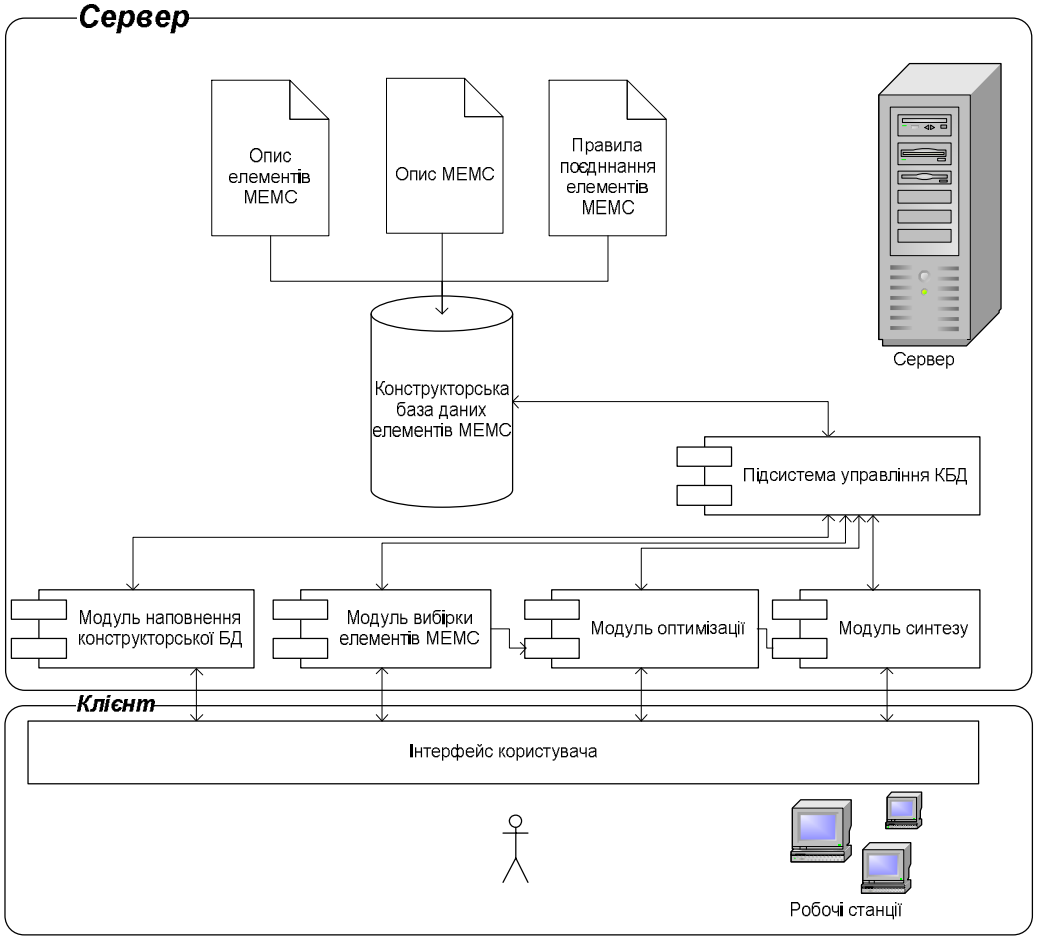

Рис. 3. Архітектура системи синтезу МЕМС

Розроблена архітектура системи, що ґрунтується на вищевказаних підсистемах, дає можливість реалізувати архітектуру клієнт-сервер. Переваги такого архітектурного рішення очевидні, а саме: система може одночасно використовуватись кількома користувачами; система керування КБД забезпечить можливість конкурентного доступу до БД; паралельне проектування, введення даних тощо.

На рис.3 зображено архутектуру системи і розподіл її модулів під час реалізації архітектури клієнт-сервер.

Розроблена архітектура має такі переваги: надає можливість реалізації як у Web, так і в WinApi; забезпечує чітку структурованість даних та функцій; наявність спеціалізованих модулів істотно зменшує обсяги роботи під час доопрацювання чи зміни логіки окремих функціональних блоків системи; відокремлена підсистема роботи з КБД забезпечує додаткову гнучкість всієї системи.

Висновки

1. На основі аналізу існуючих методів розв'язання задач синтезу МЕМС було обрано модель ситуативно базованої аргументації. Обраний метод дає можливість роз'язувати нові задачі синтезу МЕМС на основі схожих проблем, розв'язаних у минулому.

2. Розглянуто переваги та недоліки методу СБА та розроблено архітектуру системи для автоматизації проектування МЕМС.

3. Як ядро системи використано розроблену КБД елементів МЕМС.

4. Розроблено архітектуру системи, яка складається з окремих модулів, поділених за принципом функціонального навантаження. Розроблену структуру системи можна реалізувати за допомогою архітектури клієнт-сервер.

5. Архітектура клієнт-сервер забезпечує можливість для доступу і роботи з системою множини користувачів, що, своєю чергою, пришвидшить процес наповнення КБД елементів МЕМС.

1. Graf S. GA Building Blocks and Data Structures for MEMS/NEMS Design Automation and Synthesis, Master 2. Норенков И.П. Основы теории и проектирования САПР. – М.: Высшая школа, 1990. – 334с. 3.Системы автоматизированного проектирования: Учеб. пособие для вузов: В 9 кн. / И.П. Норенков. Кн.1. принципы построения и структура. – М.: Высшая школа, 1986. – 127 с.; 4.M.L. Maher and A. Gomez de Silva Garza, "Developing case-based reasoning for structural design," IEEE 5.Corie L. Cobb, Alice M. Agogino "Case-Based Reasoning for the Design of Micro-Electromechanical Systems" 6. Джозеф С. Раттц-мл.. LINQ: язык интегрированных запросов в C# 2008 для профессионалов. : Пер.с англ. – М.: ООО "Вильямс", 2008 – 1168с.: ил. 7. Троелсен Эндрю. Язык програмирования C# и платформа .NET 2.0, 3-е издание. : Пер.с англ. – М.: ООО "Вильямс", 2007 – 1168с.: ил. 8.Zelinskyy A., "Розробка конструкторської бази даних елементів МЕМС"