МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування

Кафедра інформаційних технологій

Кваліфікаційна робота бакалавра

на тему: Формування бібліотек даних електротехнічних елементів для САПР sPlan

Виконав: студент групи К-19
спеціальності 122 Комп'ютерні науки
Мироненко Юрій Дмитрович

Керівник: професор каф. АСМНСІ, д-р т. н.,
доцент Великодний Станіслав Сергійович

Консультант _____
_____

Рецензент: канд. техн. наук, доцент
Фразе-Фразенко Олексій Олексійович

Одеса 2023

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ДВВ – діаграма варіантів використання

ДД – діаграма діяльності

ДК – діаграма класів

ДКМ – діаграма компонентів

ДО – діаграма об’єктів

ДП – діаграма послідовності

ДС – діаграма станів

ДСТУ – державний стандарт України

ЕОМ – електронна обчислювальна машина

ЄСКД – єдина система конструкторської документації

ІМС – інтегральна мікросхема

КЗ – коротке замикання

МОП – метал-оксид-напівпровідник

НДІ – науково-дослідний інститут

САПР – система автоматизованого проектування

ТТЛ – транзисторно-транзисторна логіка

3D – 3-Dimensional

CAD – Computer-Aided Design

CAE – Computer-Aided Engineering

CAM Computer-Aided Manufacturing

CASE Computer-Aided Software Engineering

DRC – Design Rule Check

DWG – бінарний формат файлу

DXF – Drawing eXchange Format

EDA – Electronic Design Automation

ERC – Electrical Rule Check

FSF – Free Software Foundation

GNU – UNIX-подібна операційна система

IEEE – Institute of Electrical and Electronics Engineers

UML – Unified Modeling Language

VHDL – Very high speed integrated circuits Hardware Description Language

ВСТУП

Система автоматизованого проектування (САПР) – це система, що об'єднує технічні засоби, математичне і програмне забезпечення, параметри і характеристики яких вибирають з максимальним урахуванням особливостей завдань інженерного проектування і конструювання.

Структурними складовими САПР є підсистеми, що володіють всіма властивостями систем та які створюються як самостійні системи. Це виділені, за деякими ознаками, частини САПР, що забезпечують виконання деяких закінчених проектних завдань з отриманням відповідних проектних рішень і проектних документів.

Розробка САПР являє собою велику науково-технічну проблему, а її впровадження вимагає значних капіталовкладень. Всі створені і створювані системи проектування за допомогою ЕОМ – є автоматизованими, важливу роль у них грає людина – інженер, який розробляє проект технічного засобу.

У даний час і у найближчі роки – створення систем автоматичного проектування не передбачається, і ніщо не загрожує монополії людини на прийняття вузлових рішень у процесі проектування.

Людина у САПР повинна вирішувати, по-перше: всі завдання, які не формалізовані, по-друге: завдання, вирішення яких людина здійснює на основі своїх евристичних здібностей більш ефективно, ніж сучасна ЕОМ на основі своїх обчислювальних можливостей. Тісна взаємодія людини та ЕОМ у процесі проектування – один з принципів побудови та експлуатації САПР.

При побудові САПР – головна увага приділяється сукупності інформаційно-узгоджених підсистем. Цей дуже важливий принцип повинен ставитися не тільки стосовно до зв'язків між великими підсистемами, але і до зв'язків між більш дрібними частинами підсистем.

Інформаційна узгодженість означає, що всі або більшість можливих послідовностей завдань проектування обслуговуються інформаційно-узгодженими програмами.

Дві програми є інформаційно-узгодженими, якщо всі ті дані, які являють собою об'єкт переробки в обох програмах, входять в числові масиви, які не потребують змін при переході від однієї програми до іншої (перекодування форматів). Так інформаційні зв'язки можуть проявлятися у тому, що результати розв'язання однієї задачі – будуть вихідними даними для іншої задачі.

Ручне перекомпонування масиву веде до істотних тимчасових затримок, зростання кількості помилок і тому зменшує попит на послуги САПР. Інформаційна неузгодженість перетворює САПР у сукупність автономних програм, при цьому через не врахування в підсистемах багатьох факторів, що оцінюються в інших підсистемах – знижується якість проектних рішень, що є, на сьогоднішній день, основною проблемою з якою стикаються САПР.

Безумовно, може бути сформульовано і ряд інших принципів, що підкреслює багатобічність та складність проблеми розробки САПР. Однак якщо говорити про уніфікацію в конкретній області САПР і ця область є – електротехнікою, то не можна пропустити таку спеціалізовану САПР як sPlan, яка і є об'єктом даної кваліфікаційної роботи.

САПР sPlan – простий і зручний інструмент для креслення електронних і електричних схем, вона дозволяє легко переносити символи з бібліотеки елементів на схему і прив'язувати їх до координатної сітки. У sPlan є багато інструментів для креслення і редагування, які роблять розробку схем зручною та ефективною, такі як автонумерація елементів, складання списків елементів та ін.

Графічний редактор цієї САПР виконано із вбудованими елементами, що дозволяє легко проектувати електричні схеми. Крім того, завдяки досить простому та інтуїтивно зрозумілому інтерфейсові, sPlan може використовуватись як текстовий редактор для великих форматів, у якому легко складаються таблиці.

Для складання електричних схем присутні декілька бібліотек, елементи яких можуть бути відредаговано та додано, а такі функції як прив'язка ліній

до виносок елементів, групування елементів, прив'язка до сітки, можливість малювання ліній під певним кутом, поворот елементів, вставка малюнка, експорт у різноманітні найпопулярніші формати, зручний вивід на друк – роблять sPlan привабливим програмним продуктом у межах сегменту «легких» САПР.

Метою поданої кваліфікаційної роботи – є виконання розробки та проектування технології сполучення бібліотеки електротехнічних елементів із САПР sPlan 7.x.

Для досягнення мети, у спеціальній частині роботи буде вирішено низку встановлених задач:

– виконати огляд спеціалізованих САПР для двовимірного проектування;

– розглянути архітектуру sPlan 7.x, скласти її проектний каркас;

– визначити місце включення та взаємозв’язок нової бібліотеки із вже існуючими;

– виконати тестування нової бібліотеки у вигляді виконання декількох контрольних електротехнічних креслень.

1 АНАЛІТИЧНА ЧАСТИНА

У поданому розділі піддано аналізу різноманітні САПР, призначені для створення електротехнічних креслень, схем, конструкторської та технологічної документації. До цього списку включені найбільш поширені САх-програми (CAD, CAM, CAE) електротехнічного профілю.

1.1 Засіб для проектування печатних плат DipTrace

Програмний комплекс DipTrace являє собою САПР для повномасштабного проектування і моделювання друкованих плат [1].

В комплекс входять 4 програми:

а) DipTrace – створення плат;

б) Schematic – розробка схем;

в) SchemEdit – редактор компонентів;

г) ComEdit – підбір корпусів.

При проектуванні плат в комплексі є така можливість як підсвічування залежних елементів від редагованого елемента. У програмі ComEdit можна скористатися як вже готовими шаблонами (в комплекті з програмою йде бібліотека корпусів), так і інструментами пошарового малювання. Ручне трасування дає можливість підсвічування всіх виводів мережі і видалення в автоматичному режимі всіх розведених зв'язків (див. рис. А.1 додатку А).

Автотрасування працює за допомогою вбудованого сіткового трасувальника Simple Router. Програма може експортувати плати в Gerber, DXF і N / C Drill. У разі експорту текстових даних застосовується векторизація з кроком, який задає користувач (це дає можливість застосовувати будь-який шрифт, який є в операційній системі). Є функція підтримки векторизації растрових зображень. Комплекс включає в себе безліч сховищ корпусів і компонентів. Стандартні бібліотеки містять більше 10 тис. компонентів найбільш відомих фірм-виробників.

Програма містить мінімум керуючих елементів – відображають основні функції, при цьому перехід в основні режими, таких як вибір, переміщення, створення зв'язків, редагування трас – здійснюється автоматично при спробі провести необхідну операцію. Вся робота супроводжується підсвічуванням редагованих та залежних від них елементів, що дозволяють наочно оцінювати ситуацію. Логічна структура принципової схеми або плати формується відразу при побудові і зміна одного елемента відбивається на залежних від нього.

1.2 Програмний засіб для проектування електротехнічного обладнання ElectriCS

Програма ElectriCS спрямована на розробку електротехнічного обладнання різної складності. Програма застосовується в автомобільній, енергетичній та авіаційній промисловостях, а також верстатобудуванні.

За допомогою ElectriCS можна будувати принципові схеми, схеми з'єднань, підключень, таблиці з'єднань. Під час проектування створюється повна цифрова модель об'єкта, що допомагає контролювати правильність конструкції електрообладнання, а також отримувати звіти по складу електрообладнання та багато іншого [2].

Інструменти програми ElectriCS дозволяють робити опис будь-якого електроустаткування, аж до варіантів екранування кабелів і проводів. Програмний комплекс в автоматичному режимі робить трасування ліній зв'язку на дроти, підбір наконечників кабелів і проводів виходячи з типу клем, підбирає марки проводів і багато іншого.

Графічна частина виконується в середовищі AutoCAD (рис. А.2). Плюс до всього є і спеціалізовані інструменти від ElectriCS. Звіти в табличній формі створюються на базі Microsoft Word (у крайньому випадку, на AutoCAD). Для авіаційної промисловості програма може працювати з такими системами тривимірного проектування як Autodesk Inventor Professional і Unigraphics.

В системі є такі модулі як:

а) емулятор принципової схеми – дозволяє створювати цифрову модель електричного пристрою без відображення його на принциповій схемі;

б) контроль помилок – дозволяє відстежити грубі порушення і недоліки при проектуванні.

У програмі можна робити злиття декількох проектів в один проект.

1.3 САПР-платформа NanoCAD

NanoCAD – базова САПР і поширюється як по схемі «freeware», так і як комерційне програмне забезпечення [3].

Володіє AutoCAD-подібним інтерфейсом (рис. А.3) і безпосередньо підтримує формат DWG (за допомогою бібліотек Teigha™, розробник Open Design Alliance). На базі безкоштовної платформи nanoCAD створюється ряд платних додатків для виконання різних вузькоспеціалізованих проектних завдань.

До переваг nanoCAD можна віднести [4]:

а) нульова ціна: програмне забезпечення розповсюджується безкоштовно і доступне для комерційного використання як приватними особами, так і проектними організаціями;

б) звичний інтерфейс: принципи роботи з nanoCAD аналогічні принципам роботи в AutoCAD, що дозволяє користувачеві змінити платформу без серйозного перенавчання;

в) пряма підтримка DWG: креслення, розроблені в nanoCAD можна відкрити в середовищі AutoCAD без додаткових перетворень і навпаки, креслення розроблені в середовищі AutoCAD відкриваються в середовищі nanoCAD;

г) відкритий API: під nanoCAD можна розробляти власні програми на мовах C++,. NET, Visual Basic Script, Java Script або LISP;

До недоліків nanoCAD можна віднести:

а) відсутність підтримки AutoLISP і VBA: будь-які додатки і засоби адаптації, написані на мовах AutoLISP і VBA, на даний момент не працюють в середовищі nanoCAD без додаткової адаптації;

б) потенційні проблеми з підтримкою DWG: тому nanoCAD підтримує формат DWG за допомогою бібліотек Teigha ™, розроблених некомерційною організацією Open Design Alliance, то існує потенційна можливість втратити сумісність з оригінальним форматом DWG від компанії Autodesk.

У сформованих умовах це малоймовірно: бібліотеками ODA користуються близько 750 організацій (ODA Members, серед яких – Adobe, Oracle, Bentley, Dassault Systèmes, Siemens, Graphisoft, російські компанії – Аскон, Нанософт, СіСофт, Інфрасофт та ін.) На даний момент основна маса креслень у форматі DWG обробляється досить вірогідно, включаючи візуалізацію, редагування та збереження.

1.4 Комплексна САПР для проектування друкованих плат Cadsoft EAGLE

Cadsoft EAGLE – це комплексна САПР для проектування друкованих плат від проектування принципових схем до створення друкованої плати та її трасування [5].

Professional-версія програми дозволяє розробляти друкарські плати розміром 1600x1600 мм, з роздільною здатністю 1/10000 мм. Таку точність дозволяють здійснити вбудовану в програму модулі Schematic Module, Layout Editor, Autorouter. Крім цього, програма має досить значну бібліотеку стандартних компонентів, часто використовуваних в електротехніці, що прискорить створення плат, так як не треба буде вимальовувати все самостійно.

Програмний комплекс включає в себе:

а) графічний редактор схем (Schematic Editor) (рис. А.4);

б) редактор друкованих плат (Layout Editor);

в) гнучкий і зручний редактор бібліотек (Library Editor);

г) автотрасувальник (Autorouter).

У стандартний комплект поставки входять також модулі, перевіряючі правильність підключення електричних ланцюгів (ERC – Electrical Rule Check) і правильність розташування компонентів на платі (DRC – Design Rule Check). Причому останні дві мають більш приємний інтерфейс, ніж у більш просунутих систем.

Програма самостійно перевіряє вірність дизайну та з'єднань плати, причому користувачеві для цього не треба запускати якісь сторонні утиліти, а всі дії відбуваються всередині самого комплексу EAGLE.

Найважливішою особливістю програми Cadsoft EAGLE єя повна синхронність змін в проекті. Якщо ви внесли зміни в якій або компонент або видалили його, то ці зміни відразу ж відобразяться на малюнку плати, тоді як в інших відомих програмах доводилося постійно відстежувати синхронність.

1.5 САПР для інтегральних схем та електропроводки Electric

Electric VLSI Design System – САПР, що використовується для розробки електричних схем і проектування топології друкованих плат [6].

Крім іншого – це зручний інструмент для використання мов опису апаратури, таких як VHDL і Verilog. Electric був open-source проектом протягом багатьох років, і зараз він легко доступний через FSF (Free Software Foundation).

Electric VLSI – система автоматизованого проектування надвеликих інтегральних схем (НВІС). За допомогою Electric можна розробляти інтегральні МОП та біполярної схеми, друковані плати або схеми будь-якого типу.

Electric має безліч стилів редагування, включають планування, схематику, ілюстрації, архітектурне проектування (рис. А.5).

Electric може взаємодіяти з різними специфікаціями і форматами файлів як VHDL, CIF, GDS II. Найбільш цінна вбудована в Electric можливість –

це система прив'язок, яка дає можливість здійснювати проектування зверху вниз з дотриманням цілісності всіх з'єднань.

1.6 Комплекс для проектування електронних схем KiCad

KiCad – поширюваний за ліцензією GNU General Public License програмний комплекс класу EDA з відкритими вихідними текстами, призначений для розробки електричних схем і друкованих плат (рис. А.6) [7].

Кросплатформність компонентів KiCad забезпечується використанням бібліотеки wxWidgets. Підтримуються операційні системи GNU / Linux, Windows NT 5.x, FreeBSD і Solaris.

Програми, що входять в KiCad:

а) KiCad – менеджер проектів;

б) Eeschema – редактор електричних схем;

в) вбудований редактор символів схем (бібліотечних компонентів);

г) Pcbnew – редактор друкованих плат;

д) вбудований редактор образів посадочних місць (бібліотечних компонентів);

е) 3D Viewer – 3D-переглядач друкованих плат на базі OpenGL (частина pcbnew);

ж) Gerbview – переглядач файлів Gerber (фотошаблонів);

и) Cvpcb – програма для вибору посадочних місць, відповідних компонентів на схемі;

к) Wyoeditor – текстовий редактор для перегляду звітів.

Компоненти KiCad забезпечує:

а) створення однолістових та ієрархічних схем;

б)перевірку їх коректності ERC (контроль електричних правил);

в) створення списку електричних ланцюгів netlist для редактора топології плати pcbnew або для Spice-моделювання схеми;

г) доступ до документації на використовувані в схемі електронні компоненти (datasheet);

д) розробку плат, що містять до 16 шарів міді і до 12 технічних шарів (шовкографія, паяльна маска тощо);

е) вихід на зовнішні трасувальники з'єднань у вигляді генерації опису плати на Specctra Design Language (on-line FreeRoute та ін.);

ж) генерацію технологічних файлів для виготовлення друкованих плат (Gerber-файли для фотоплоттерів, файли сверловок і файли розміщення компонентів);

и) пошарова друк схем і креслень друкованих плат на принтері або плоттері (у форматах PostScript, HPGL, SVG і DXF), з рамкою формату або без неї;

к) gerbview дозволяє переглядати Gerber-файли.

У складі KiCad поставляються бібліотеки електронних компонентів (звичайних і тих, що поверхнево монтуються – SMD). Для багатьох бібліотечних компонентів є 3D-моделі, створені в Wings3D.

Компоненти і посадочні місця корпусів можна асоціювати з документацією, ключовими словами і здійснювати швидкий пошук компонента за функціональним призначенням.

2 ПРОЕКТНИЙ РОЗДІЛ

У проектному розділі бакалаврської кваліфікаційної роботи виконується побудова проектного каркасу (архітектури) майбутніх функцій розширення САПР sPlan. При формуванні архітектури використовується розширена нотація UML 2.5 та CASE-інструментарій Enterprise Architect 14.0.

2.1 Діаграма варіантів використання

Діаграми варіантів використання (ДВВ) описують функціональне призначення системи або те, що система повинна робити [8]. Розробка ДВВ переслідує наступні цілі:

а) визначити загальні межі і контекст модельованої предметної області;

б) сформулювати загальні вимоги до функціонального поведінки системи, що проектується;

в) розробити вихідну концептуальну модель системи для її подальшої деталізації у формі логічних і фізичних моделей;

г) підготувати вихідну документацію для взаємодії розробників системи з її замовниками і користувачами.

Суть ДВВ полягає в наступному. Проектована система представляється у вигляді безлічі сутностей або акторів, що взаємодіють з системою за допомогою варіантів використання. При цьому актором (actor) або дійовою особою називається будь-яка сутність, що взаємодіє з системою ззовні [9]. Це може бути людина, технічний пристрій, програма або будь-яка інша система, яка може служити джерелом впливу на систему, що моделюється. Акторів зображують у вигляді людських фігурок.

Варіант використання служить для опису сервісів, які система надає актору. ДВВ може доповнюватися пояснювальним текстом, який розкриває сенс або семантику складових її компонентів [10].

Окремий варіант використання позначається на діаграмі еліпсом, усередині якого міститься його коротку назву або ім'я у формі дієслова з пояснювальними словами. Мета варіанти використання полягає в тому, щоб визначити закінчений аспект або фрагмент поведінки деякої сутності без розкриття її внутрішньої структури. В якості такої сутності може виступати система або будь-який елемент моделі, який володіє власною поведінкою.

Прецедентом (Use Case) називається опис безлічі послідовностей дій (включаючи варіанти), виконуваних системою для того, щоб актор міг отримати певний результат. Графічно прецедент зображується у вигляді еліпса.

Варіанти використання можуть застосовуватися як для специфікації зовнішніх вимог до проектованої системи, так і для специфікації функціонального поведінки вже існуючої системи. Безліч варіантів використання, в цілому, повинно визначати всі можливі сторони очікуваної поведінки системи. Крім цього, варіанти використання неявно встановлюють вимоги, що визначають, як актори повинні взаємодіяти з системою, щоб мати можливість коректно працювати з наданими сервісами. Для зручності безліч варіантів використання може розглядатися як окремий пакет.

Спроектована у кваліфікаційній роботі ДВВ наведена на рис. 2.1.

Пояснимо сформовану ДВВ.

Центральною сутністю діаграми – є актор «sPlan 7.x», виконаний зі стереотипом «business actor», що означає об’єднання у собі усіх функцій системи, що розглядається.

Цей актор виконує реалізацію (зв'язок різновиду «Realized» з пунктирною стрілкою) кооперативного варіанта використання (Collaboration Use) «Побудова схем».

Кооперація «Побудова схем» включає (зв'язок «Extended») наступні прецеденти: електронні, електротехнічні, радіотехнічні, електричні, що означають можливість побудови цих різновидів схем у САПР sPlan 7.x.

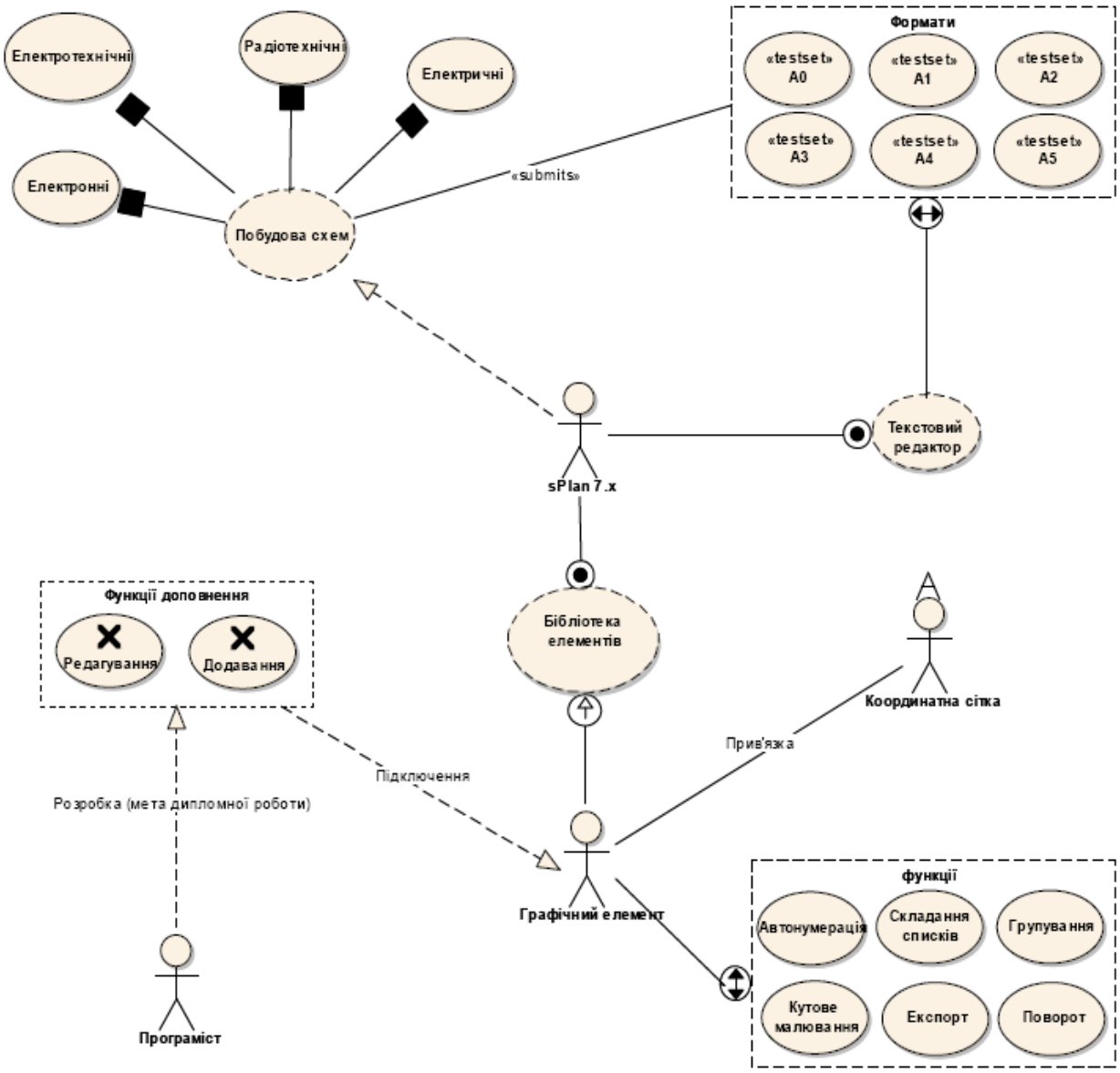

Рисунок 2.1 – ДВВ sPlan 7.x

За допомогою зв’язків зі стереотипами «вмісту» («contained») до центрального бізнес-актора під’єднано кооперації («Collaboration»):

– текстовий редактор;

– бібліотека елементів,

які слід сприймати як ті, що включені до САПР sPlan 7.x. Тобто цю САПР можна використовувати як своєрідний текстовий редактор із сукупність спеціалізованих елементів.

Кооперація «Текстовий редактор», у свою чергу, розширюється (зв'язок «expanded») на сукупність різноманітних форматів умовного креслярського аркушу. Ці формати об’єднані у пакет («Boundary») у вигляді прецедентів тестової послідовності («testset»): A0 – A5.

Цей пакет за допомогою зв’язку «submits» – підлягає кооперації «Побудова схем», що означає: побудову схем можна виконувати тільки на тих форматах, які входять до пакету.

Кооперація «Бібліотека елементів» має зв'язок узагальнення («generalized»), що веде до актору-виконавця («worker») «Графічний елемент». Цей зв'язок означає, що «Графічний елемент» (до речі із множинністю екземплярів) входить до складу «Бібліотеки елементів» Інакше кажучи, кооперація-бібліотека має у собі низку дочірніх графічних елементів.

Крім зв’язку узагальнення, «Графічний елемент» має ще три зв’язки:

– асоціацію (association);

– розширення (expanded);

– реалізацію (realization).

Почнемо з асоціації «Прив’язка» , що веде до вбудованого агенту (agent) «Координатна сітка». У даному випадку – це означає, що будь-який створений графічний елемент прив’язується до координатної сітки, яка, у свою чергу вбудована (у вигляді додаткового модуля) та має свої функціональні особливості використання.

Зв'язок розширення – поєднує «Графічний елемент» із пакетом «Функції» (функціональним пакетом), що містить у собі склад маніпуляційних операцій з графічними елементами у вигляді бізнес-прецедентів, а саме:

– автонумерація;

– складання списків;

– групування;

– кутове малювання;

– експорт;

– поворот.

Нарешті, зв’язком-реалізацією – підключається ціла Use-CASE-гілка, виконання якої й веде до досягнення мети кваліфікаційної роботи.

Почати її коментар слід з актора. Актор «Програміст» реалізує (зв'язок «realization») пакет функцій доповнення, що у кінцевому випадку (через актора «Графічний елемент») підключається до бізнес-актора «sPlan 7.x». Цей пакет містить сукупність тестових даних (testcase):

– редагування;

– додавання;

які (при позитивних випробуваннях) можуть бути додані (через «Графічний елемент») до «Бібліотеки елементів» та, згодом, увійти до розширеного дистрибутиву САПР sPlan 7.x.

2.2 Діаграма послідовності

Діаграми послідовності (ДП) відображають потік подій, що відбуваються у рамках варіанта використання [11].

ДП ілюструє послідовність дій, що реалізують варіант використання. Аналітики бачать послідовність (потік) дій, розробники – об'єкти, які треба створити, та їх операції. Фахівці з контролю якості зрозуміють деталі процесу і зможуть розробити тести для їх перевірки [12]. Таким чином, ДП корисні всім учасникам проекту.

У кваліфікаційній роботі буде розглянуто конкретну ДП, що побудована для останньої гілки ДВВ (див. розд. 2.1).

Спроектовану ДП наведено на рис. 2.2. Першим об’єктом, з якого починається ДП – є актор «Програміст», екземпляр якого імпортовано з ДВВ (див. рис. 2.2). Його перше повідомлення спрямовано до об’єкту, що являє собою пакетний фрагмент (seq fragment) «Головне меню». До цього фрагменту входять:

– керуючий клас (control) «Опції»;

– клас-сутність (entity) «Основні параметри».

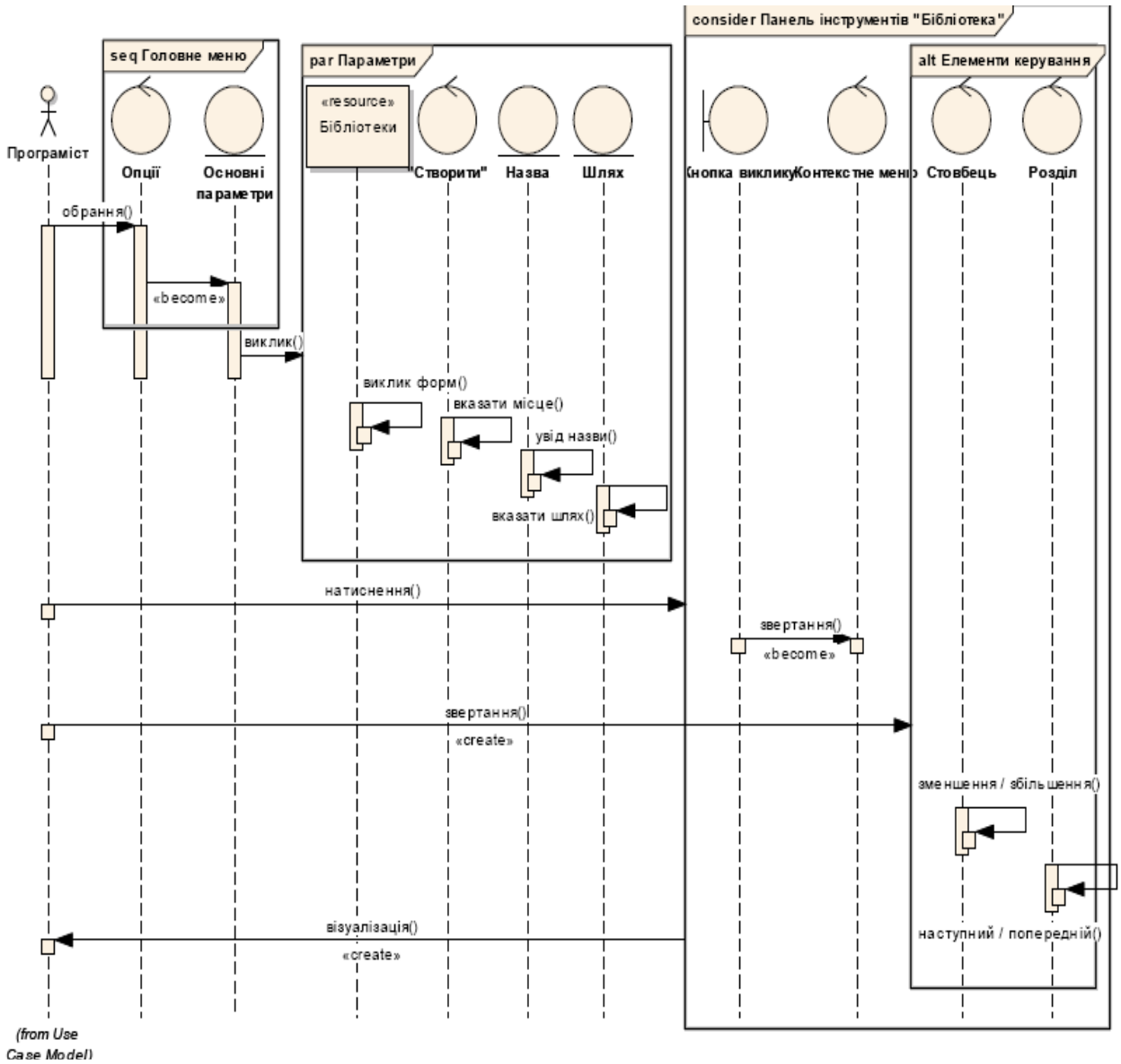

Рисунок 2.2 – ДП створення бібліотеки елементів для САПР sPlan 7.x

Перше повідомлення від об’єкту «Програміст» – «обрання» спрямоване до керуючого класу (control) «Опції» пакетного фрагменту «Головне меню». Це означає, що програміст обирає опції головного меню.

Наступне повідомлення функціонує у межах пакету «Головне меню». Спрямовано воно від керуючого класу «Опції» до класу-сутності «Основні параметри». Це повідомлення має стереотип «звертання» («become») та означає звертання головного меню «Опції» до основних параметрів цих опцій.

Повідомлення «Виклик» спрямовано до пакетного фрагменту (par fragment) «Параметри», який містить у собі об’єкти, що є параметрами створення нової бібліотеки. Такими об’єктами є:

– ресурс загального доступу («resource») «Бібліотеки»;

– керуючий клас (control) «Створити»;

– клас-сутність (entity) «Назва»;

– клас-сутність (entity) «Шлях».

В межах даного пакету на лініях життя (life line) об’єктів – розміщуються різноманітні повідомлення, проте із однаковим рефлексивним (to self) стереотипом, що означають виконання дії, у межах поточного класу. До таких дій відносяться (див. рис. 2.2):

– «виклик форм»;

– «вказати місце»;

– «увід назви»;

– «вказати шлях».

Для пояснення роботи об’єктів в межах пакетного фрагменту «Параметри», слід послідовно розібрати рефлексивні дії. При обранні ресурсу загального доступу (resource) «Бібліотеки» – відбувається виклик форм створення / настроювання / редагування бібліотек графічних елементів. При звертанні до керуючого класу (control) «Створити» – активізується процес створення нової бібліотеки із формуванням запиту «Вказати місце», що вказує майбутнє місце збереження бібліотеки.

При звертанні до класу-сутності (entity) «Назва» – активується запит «Увід назви», що керує процесом нової ініціалізації бібліотеки, яка створюється. При звертанні до класу-сутності (entity) «Шлях» – активується запит «Вказати шлях», що прописує новий адрес збереження параметрі бібліотеки.

Наступна група повідомлень («натиснення») спрямована від «Програміста» до складеного пакету (consider) «Панель інструментів «Бібліотека»», що дозволяє керувати розділами бібліотеки та містить у своєму складі:

– пакетний фрагмент (alt fragment) «Елементи керування»;

– граничний клас (boundary) «Кнопка виклику»;

– керуючий клас (control) «Контекстне меню».

До складу пакетного фрагменту (alt fragment) «Елементи керування» входять об’єкти:

– керуючий клас (control) «Стовбець»;

– керуючий клас (control) «Розділ»;

із рефлексивними (to self) стереотипами (рис. 2.2):

– «зменшення / збільшення»;

– «наступний / попередній».

Пояснимо роботу складеного пакету «Панель інструментів «Бібліотека»».

Після реалізації «Програмістом» повідомлення «Натиснення» – активується складений пакет «Панель інструментів «Бібліотека»». Далі – при натисканні граничного класу «Кнопка виклику» за допомогою повідомлення «Звертання» зі стереотипом «become», викликається керуючий клас «Контекстне меню» з різноманітними параметрами виклику, редагування, маніпулювання та змінення елементів бібліотек.

Паралельним потоком подій можна вважати ініціалізацію повідомлення «Звертання» зі стереотипом породження виклику (create), що спрямовано від «Програміста» до пакетного фрагменту (alt fragment) «Елементи керування». Даний фрагмент реалізує керування об’єктами «Пакет» та «Розділ» за допомогою маніпуляційних опцій «зменшення / збільшення» та «наступний / попередній» (див. рис. 2.2).

Завершується ДП – повідомленням «Візуалізація» від складеного пакетного фрагменту «Панель інструментів «Бібліотека» до імпортованого класу «Програміст» зі стереотипом породження (create), що означає візуалізацію користувачеві усіх форм та параметрів процесу створення.

Таким чином, у процесі складання ДП – були спроектовані основні етапи створення майбутньої бібліотеки електротехнічних елементів для

САПР sPlan 7.x, що будуть сформовані та запрограмовані у наступних підрозділах кваліфікаційної роботи.

2.3 Діаграма об’єктів

Діаграми об’єктів – ДО (object diagrams) застосовують для моделювання фрагментів майбутньої системи [13]. ДО відображують реально існуючі екземпляри класів та їх значення. ДО являють собою структурні діаграми, що є прообразом майбутньої діаграми класів [14].

Спроектовану ДО для структури об’єктів графічних бібліотек, їх значень та засобів керування у САПР sPlan 7.x наведено на рис. 2.3.

На спроектованій ДО можна виділити наступні об’єкти структури:

а) 1 об’єкт-система (object system);

б) 6 структурних модулів (organization unit);

в) 1 об’єкт збереження (object written);

г) 2 сутності спільного використання (collaboration use);

д) 8 звичайних об’єктів (object).

Також на ДО присутні наступні зв’язки:

а) 1 залежність (dependency);

б) 5 реалізацій (realization);

в) 1 агрегація (aggregated);

г) 1 зв'язок до примітки (note link);

д) 6 збірок (assembly).

Центральним об’єктом ДО – є об’єкт-система (object system) «Графічний елемент», що, фактично, представляє набір примітивів для відображення своєї сутності.

Цей об’єкт поміщується на об’єкт збереження (object written) «Аркуш» (тобто накладається на його шар з подальшим збереженням структури як документа). Ця операція описана зв’язком-агрегацією (aggregated), що на ДО має дещо інший вигляд (лінія з кружечком навколо знаку «+» біля об’єкту-цілого).

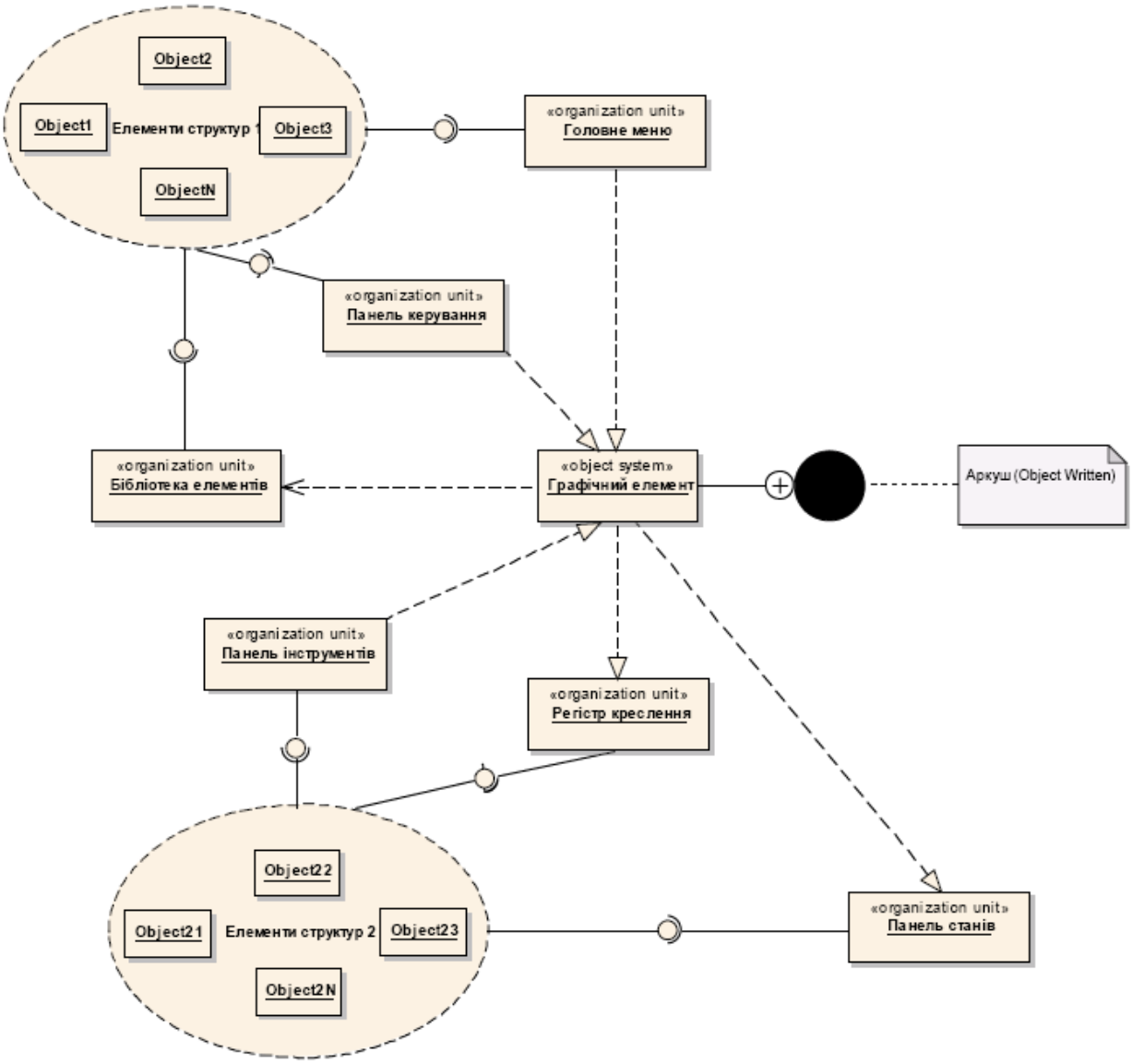

Рисунок 2.3 – ДО для керування створенням графічних елементів у САПР sPlan 7.x

Структурні модулі (organization unit) розміщуються таким чином, щоб вони відповідали геометрії їх розміщення у інтерфейсі sPlan 7.x.

Так у самій верхній частині розміщується структурний модуль «Головне меню», яке збирається за допомогою зв'язку «assembly» зі звичайних об'єктів керування (object1 – objectN), що об'єднані сутністю спільного використання (collaboration use) «Елементи структур 1».

До речі, з подібних об'єктів керування (object1 – objectN) збираються й структурні модулі «Панель керування» та «Бібліотека елементів» (рис. 2.3),

що розміщуються, відповідно, нижче структурного модуля «Головне меню» та нижче «Панелі керування».

За тим же принципом організовано збірки (assembly) структурних модулів (organization unit) «Панель інструментів», «Регістр креслення» та «Панель станів». Ці модулі (стосовно до інтерфейсу sPlan 7.x) будуть розміщуватися геометрично нижче об'єкту-системи (object system) «Графічний елемент», один під одним, починаючи з «Панелі інструментів» і закінчуючи «Панеллю станів». Збірки цих модулів виконуються з другої сутності спільного використання (collaboration use) «Елементи структур 2», яка також об'єднує у собі звичайні об'єкти керування (object21 – object2N)).

Зв'язки структурних модулів (organization unit) з об'єктом-системою (object system) «Графічний елемент» виконані у вигляді реалізацій (realization), причому трьох – вхідних та двох – вихідних. Це означає, що структурні модулі «Головне меню», «Панель керування» та «Панель інструментів» реалізують керування «Графічним елементом», зміни параметрів та властивостей якого реалізують зміни у «Регістрі креслення» та «Панелі станів» з відповідним відображенням змін у інтерфейсі sPlan 7.x.

Об'єкт-система (object system) «Графічний елемент» залежить від структурного модуля (organization unit) «Бібліотека елементів», що показує залежність (dependency) на ДО (рис. 2.3). Це означає, що зовнішній вигляд графічного елементу буде вкладатися у рамки структури тієї бібліотеки, до якої він належить.

Таким чином, у даному розділі було спроектовано ДО, що являє собою подальшу основу для проектування діаграми класів.

2.4 Діаграма класів

Діаграма класів (ДК) – статична структурна діаграма, що описує структуру системи, вона демонструє класи системи, їхні атрибути, методи і залежності між класами [15].

Класи – це найважливіші будівельні блоки будь об'єктно-орієнтованої системи. Вони являють собою опис сукупності об'єктів із загальними атрибутами, операціями, відносинами та семантикою. Клас реалізує один або декілька інтерфейсів [16].

Добре структуровані класи характеризуються чіткими кордонами і допомагають формувати збалансований розподіл обов'язків у системі. Клас (class) – абстрактне опис безлічі однорідних об'єктів, що мають однакові атрибути, операції та відносини з об'єктами інших класів [17].

ДК при моделюванні об'єктно-орієнтованих систем зустрічаються частіше інших. На таких діаграмах показується безліч класів, інтерфейсів, кооперацій і відносин між ними. ДК використовуються для моделювання статичного виду системи з точки зору проектування. Сюди здебільшого відноситься моделювання словника системи, кооперацій і схем. Крім того, ДК складають основу ще двох діаграм – компонентів і розгортання. ДК важливі не тільки для візуалізації та специфікації документування структурних моделей, але також для прямого і зворотного проектування виконуваних систем.

Побудовану ДК для бібліотек графічних елементів САПР sPlan 7.x наведено на рис. 2.4. Пояснимо принцип її проектування.

Перш за все побудова ДК – заснована на базі спроектованої діаграми об’єктів (рис. 2.3).

До статичної картини ДК входять наступні класи та сутності:

а) 1 пакет (public package);

б) 3 сутності сигналу (signal);

в) 3 сутності примітиву (primitive type);

г) 1 табличний клас (table class);

д) 5 типових класів (public class);

е) 1 клас-інтерфейс (interface).

Також до ДК входять наступні зв’язки:

а) 1 залежність (dependency);

б) 5 інформаційних потоків (information flow);

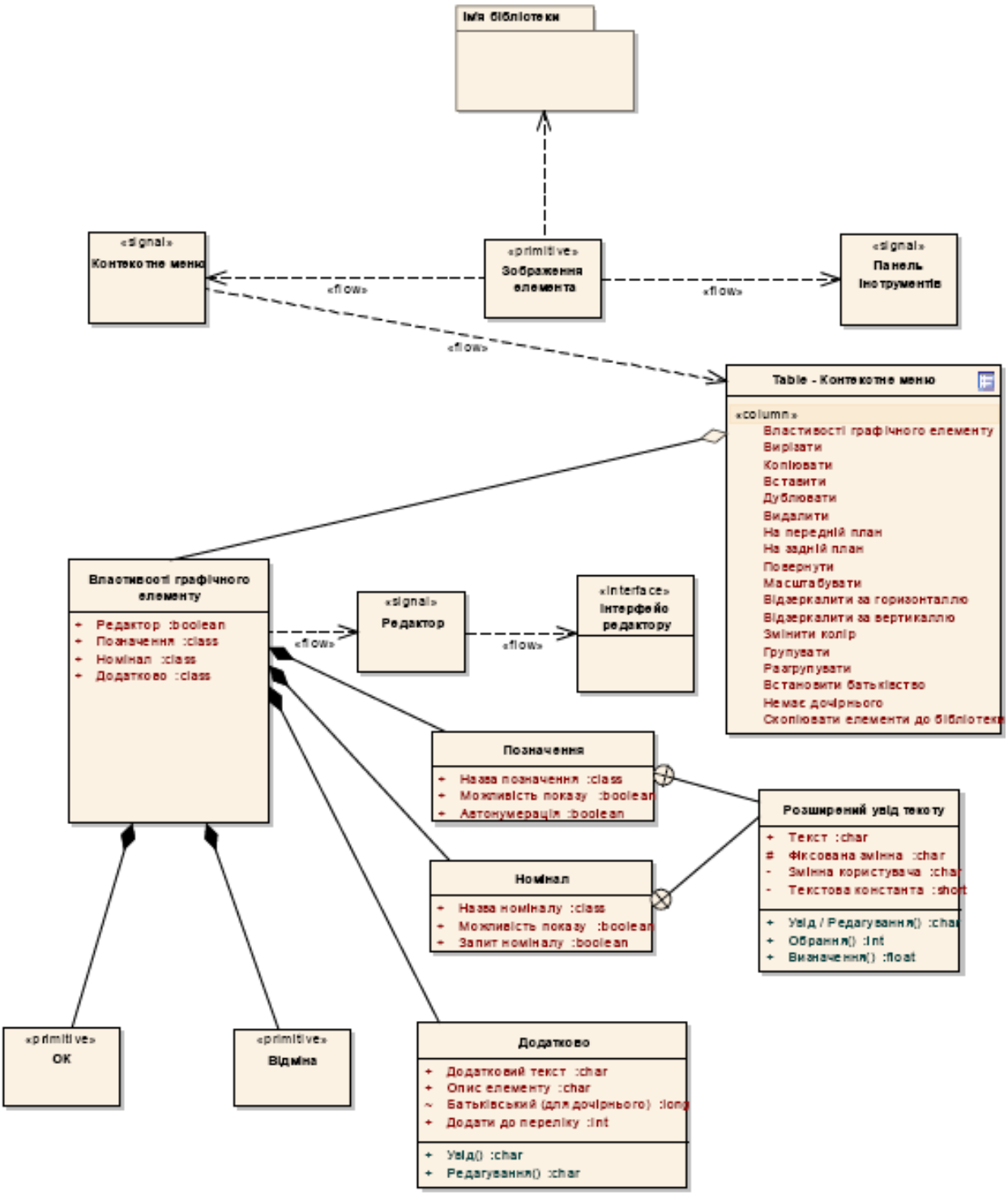

Рисунок 2.4 – ДК для бібліотек графічних елементів САПР sPlan 7.x

в) 1 агрегація (aggregation);

г) 5 композицій (compose);

д) 2 вкладеності (nesting).

ДК починається пакетом (public package) «Ім’я бібліотеки» (див. рис. 2.4), який означає, що усі майбутні графічні елементи будуть об’єднані у конкретні свої пакети.

Далі пакет «Ім’я бібліотеки» пов’язане із примітивом (primitive type) «Зображення елемента» зв’язком-залежністю (dependency), яка означає, що зображення будь-якого елементу залежить від типу його бібліотеки. Таким чином, усі майбутні графічні елементи будуть сортовані у бібліотеки за їх типами.

Від примітиву (primitive type) «Зображення елемента» у два боки розходяться зв’язки інформаційних потоків (information flow). З одного боку – до сутності сигналу (signal) «Контекстне меню», а з другого – до сутності сигналу (signal) «Панель інструментів». Розглянуті інформаційні потоки означають, що деякий сигнал, що приведе до виклику наступних за ним форм або діалогів можна отримати різними способами – у нашому випадку – через «Контекстне меню» або «Панель інструментів».

Оскільки панель інструментів вже спроектована у САПР sPlan 7.x, то зупинимось на способу виклику через «Контекстне меню».

Від сутності сигналу (signal) «Контекстне меню» до табличного класу (table class) «Контекстне меню» веде інформаційний потік (information flow), що означає виклик табличної форми за допомогою контекстного меню.

Ця таблична форма (або табличний клас) у спроектованому вигляді повинна містити наступні рядки однієї колонки:

– властивості графічного елементу;

– вирізати;

– копіювати;

– вставити;

– дублювати;

– видалити;

– на передій план;

– на задній план;

– повернути;

– масштабувати;

– віддзеркалити за горизонталлю;

– віддзеркалити за вертикаллю;

– змінити колір;

– групувати;

– розгрупувати;

– встановити батьківство;

– немає дочірнього;

– скопіювати елементи до бібліотеки.

Рядки табличного класу – будуть командами маніпулювання з графічним елементом. Розглянемо перший рядок «Властивості графічного елементу» до якого під’єднаний зв'язок-агрегація (aggregation). Цей зв'язок переводить фактичну команду рядка до виклику багаторівневого класу «Властивості графічного елементу», що саме й містить виключно усі його властивості.

Багаторівневий клас «Властивості графічного елементу» містить у собі:

а) команду виклику вбудованого редактору (із Булевим типом даних (Boolean));

б) клас «Позначення»;

в) клас «Номінал»;

г) додатковий клас;

д) сутність-примітив (primitive type) «ОК»;

е) сутність-примітив (primitive type) «Відміна».

Розглянемо усі вкладення багаторівневого класу із характеристиками зв’язків та поясненнями структур.

Команда виклику вбудованого редактору «Редактор» – являє собою Булеву функцію (Boolean), яка відповідно викликає або не викликає відповідний редактор. До цієї команди приєднано інформаційний потік (information flow), що веде за собою виклик (або не виклик – у випадку логічного нуля) сигналу до редактора (рис. 2.4). Далі за допомогою аналогічного інфо-

рмаційного потоку викликається клас- інтерфейс (interface) «Інтерфейс редактору», що являє собою редактор графіки елементів.

Клас «Позначення» – служить для уведення позначення для специфікації елемента, містить:

– клас із вкладенням (nesting) «Назва позначення»;

– булевий атрибут (Boolean) «Можливість показу»;

– булевий атрибут (Boolean) «Автонумерація».

Характеристику вкладеному класу (nesting) «Розширений увід тексту» – надамо пізніше тому, що він також вкладається й у інший клас «Номінал».

Булевий атрибут «Можливість показу» – надає або не надає можливість показу графічному елементу. Булевий атрибут «Автонумерація» – надає можливість проводити автоматичну нумерацію усіх елементів на аркуші. Цей та попередній атрибути – будуть мати керування у вигляді прапорця (check box), що найоптимальніше підходить для Булевих функцій.

Клас «Номінал» – необхідний для вказівки параметрів номіналу для кожного з елементів, клас містить:

– клас із вкладенням (nesting) «Назва номіналу»;

– булевий атрибут (Boolean) «Можливість показу»;

– булевий атрибут (Boolean) «Автонумерація».

Вкладений клас (nesting) «Розширений увід тексту» – прикріплено до двох класів – «Позначення» та «Номінал». Вкладений клас містить:

а) відкритий (public) атрибут «Текст» із символьним типом даних (char) – для організації уведення тексту;

б) захищений (protected) атрибут «Фіксована змінна» із символьним типом даних (char) – для уведення змінної;

в) закритий (private) атрибут «Змінна користувача» із символьним типом даних (char);

г) закритий (private) атрибут «Текстова константа» із числовим типом даних у скороченому поданні (short);

д) відкриту (public) операцію «Уведення / редагування» тексту із символьним типом даних (char);

е) відкриту операцію «Обрання» із цілочисельним типом даних (integer);

ж) відкриту операцію «Визначення» із чисельним типом даних з точкою, що змінюється (float).

Додатковий клас необхідний для визначення додаткових та (за необхідністю) розширений властивостей графічних елементів, клас «Додатково» (рис. 2.4) – містить у собі наступні сутності:

а) відкритий (public) атрибут «Додатковий текст» із символьним типом даних (char) – для організації уведення додаткового тексту;

б) відкритий (public) атрибут «Опис елементу» із символьним типом даних (char)

в) пакетний (package) атрибут «Батьківський (для дочірнього)» із чисельним типом даних довгого формату (long) – для вказівки батьківського пакету для елементу;

г) відкритий (public) атрибут «Додати до переліку» із цілочисельним типом даних (integer);

д) відкриту (public) операцію «Увід» із символьним типом даних (char);

е) відкриту (public) операцію «Редагування» із символьним типом даних (char).

Останніми до багаторівневого класу «Властивості графічного елементу» під’єднано дві сутності-примітиву (primitive type) «ОК» та «Відміна», що являють собою звичайні кнопки (button) з реалізацією функцій прийняття або відхилення змін у редагуванні.

Таким чином, на спроектованій ДК приведено всі необхідні статичні класи та сутності, які знадобляться при розробці та подальшому керуванні графічними бібліотеками елементів.

2.5 Діаграма станів

Діаграма станів – ДС (State Machine diagram) – діаграма, на якій представлені кінцевий автомат із простими станами, переходами та композитними станами системи, що проектується [18].

Кінцевий автомат (State machine) – специфікація послідовності станів, через які проходить об'єкт або взаємодія у відповідь на події свого життя, а також відповідні дії об'єкта на ці події [19]. Кінцевий автомат прикріплений до вихідного елементу (стану або методу) та служить для визначення сценарію поведінки його екземплярів.

ДС спроектована для бібліотеки електротехнічних елементів САПР sPlan 7.x наведена на рис. 2.5.

До цієї діаграми входять:

а) 1 початковий стан (initial);

б) 1 автомат (state machine);

в) 4 стани (state);

г) 1 з’єднувач (junction);

д) 1 знищувач (terminate);

е) 1 кінцевий стан (final);

ж) 12 переходів (tratsition).

ДС має циклічну структуру, проте кожний перехід реалізується тільки у випадку необхідності, ініційованої з боку користувача та виконання умов, що обмежують.

Починається діаграма початковим станом (initial). Далі, у випадку виконання умови (guard condition), «Завантаження sPlan 7.x» – виконується перехід до стану (state) «Створення», що саме починає процес створення нового графічного елементу бібліотеки.

Якщо виконується умова (guard condition) «Активація редактору», тобто запущено редактор sPlan – відбувається перехід до центрального елементу ДС – кінцевого автомату (state machine) «Електротехнічний елемент».

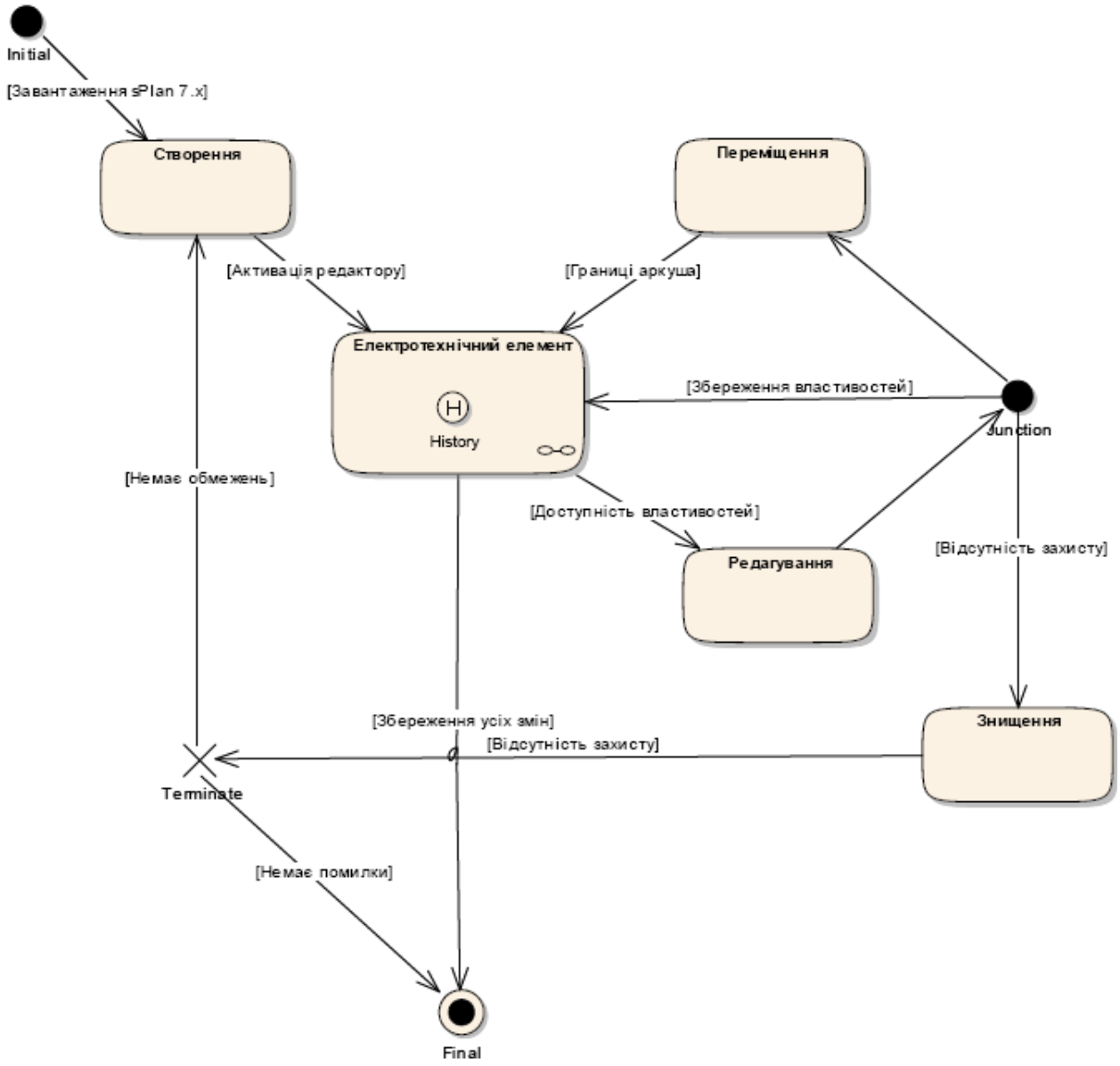

Рисунок 2.5 – ДС бібліотеки електротехнічних елементів для САПР sPlan 7.x

Кінцевий автомат «Електротехнічний елемент» містить у собі сценарій поведінки у вигляді вкладених умов та програмних сторінок, таким чином, він являє собою пакетну структуру. Крім того, кінцевий автомат містить сутність «Історія стану» (state history), що документує не тільки сценарну поведінку, але й фіксує усі точки входу та виходу.

Далі (за головною діагоналлю каскадної моделі) йде перехід до стану «Редагування» елементу, але відбувається він тільки у випадку виконання умови обмеження (guard condition) «Доступність властивостей».

Після проходження стану «Редагування» – перехід веде до з'єднувача переходів (junction), який ми ще пройдемо декілька разів. Зі з'єднувача – веде 3 переходи:

– до кінцевого автомату «Електротехнічний елемент»;

– до стану «Переміщення»;

– до стану «Знищення».

Розглянемо послідовно всі переходи, що ведуть зі з'єднувача (junction).

Перший перехід до кінцевого автомату (state machine) «Електротехнічний елемент» відбувається при виконанні обмежуючої умови «Збереження властивостей». Цей перехід слід розуміти, як остаточне формування змін у властивостях елементу та їх збереження перед відображенням елемента із новими властивостями. Після цього можливе виконання переходу з «Електротехнічного елементу» до кінцевого стану (final) при виконанні умови «Збереження усіх змін» у сценарії автомату.

Другий перехід веде до стану «Переміщення». Цей стан означає вільне пересування елементу бібліотеки за площиною шару аркуша. Вихід зі стану можливий, якщо внаслідок цього переміщення, елемент знаходиться у межах допустимого розміщення на аркуші. За цим слідкує обмежуюча умова (guard condition) «Границі аркуша». Далі – усі виходи з кінцевого автомату «Електротехнічний елемент» – вже було розглянуто.

Третій перехід до стану «Знищення» – відбувається при виконанні умові «Відсутність захисту». Зі стану «Знищення» – виконується перехід до сутності-знищувача (terminate), що виконує остаточне графічне видалення відображення елемента.

Зі знищувача (terminate) – виходить два переходи. Один – без обмежуючої умови веде до стану «Створення» та необхідний для початку нового циклу створення графічних елементів. Другий – відбувається при виконанні обмежуючої умови «Немає помилки» (необхідної для здійснення перевірки помилковості знищення елементу та можливості повернення дій) – веде до кінцевого стану (final) та до виходу з САПР sPlan 7.x.

У процесі побудови ДС – було сформовано сценарну поведінку при формуванні графічних бібліотек електротехнічних елементів, що будуть розроблюватись у САПР sPlan 7.x.

2.6 Діаграма діяльності

Діаграма діяльності (ДД) – англ. activity diagram – діаграма, на якій вказується розкладання деякої діяльності на її складові частини [20]. Під діяльністю (activity) розуміється специфікація виконуваного поведінки у вигляді координованого послідовного чи паралельного виконання підлеглих елементів (вкладених видів діяльності, окремих дій (action)), з'єднаних між собою потоками, які йдуть від виходів одного вузла до входів іншого.

ДД використовуються при моделюванні бізнес-процесів, технологічних процесів, усіх послідовних та паралельних обчислень [21]. Найбільш близьким і точним аналогом діаграм діяльності є математично строгі дракон-схеми візуальної алгоритмічної мови ДРАКОН; більш віддаленим аналогом діаграм діяльності – є схеми алгоритмів з ГОСТ 19.701-90.

Спроектовану ДД для процесу створення бібліотеки елементів (у ескізному варіанті) – наведено на рис. 2.6. На поданому ескізі (рис. 2.6) відображено загальний вигляд ДД, оскільки формат аркуша кваліфікаційної роботи – не дає змогу привести цілу деталізовану ДД.

На поданій діаграмі присутні наступні сутності:

а) 4 сектора активності (Activity Partition);

б) 1 початкова активність (Activity Initial);

в) 2 типові активності (Public Activity);

г) 7 блоків-вирішень (Decision);

д) 5 різноманітних дій (Action);

е) 1 спрямування (Send);

ж) 1 приймання (Receive);

и) 4 накопичувача даних (Data Store);

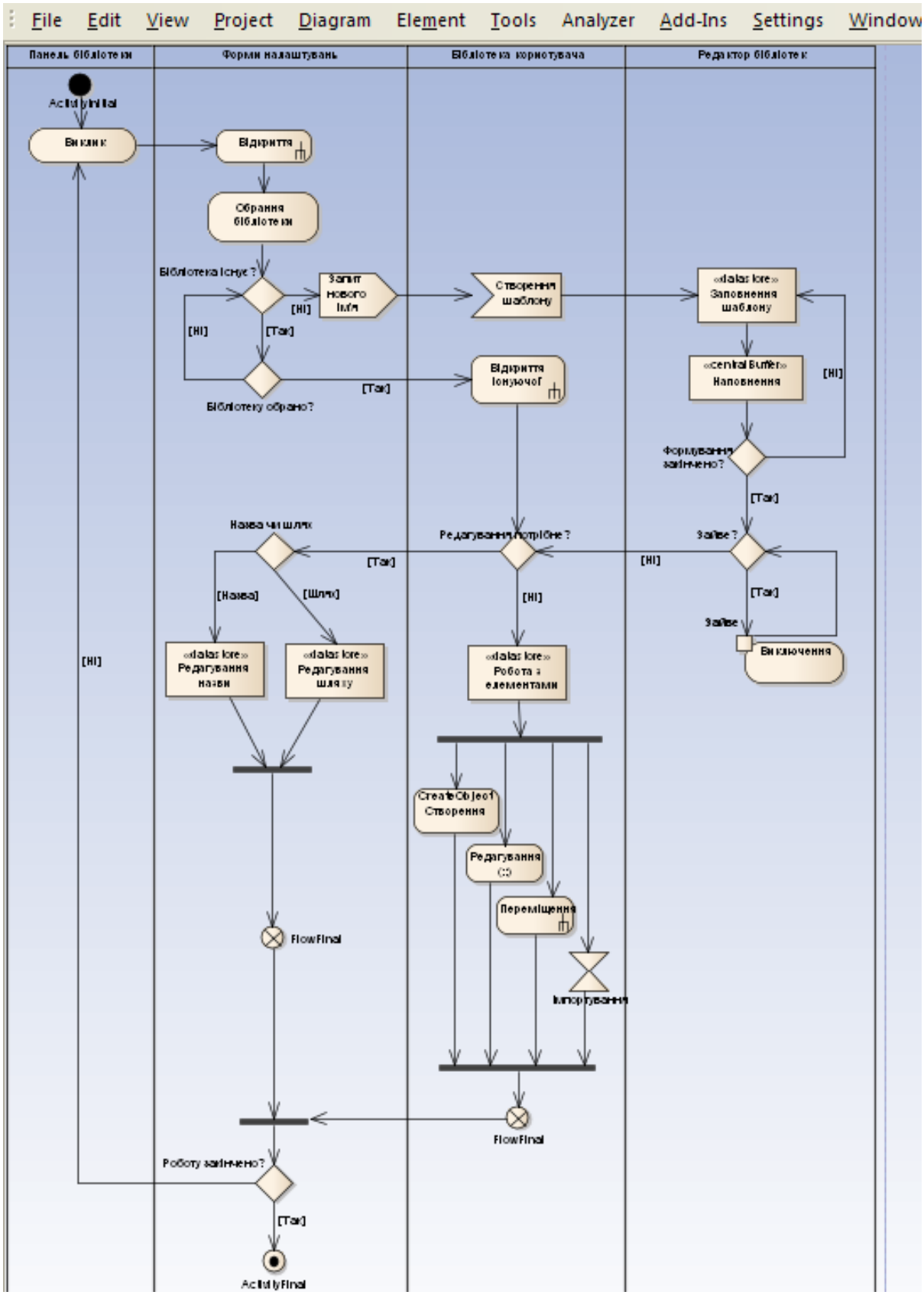

Рисунок 2.6 – Ескіз ДД для процесу створення бібліотеки електротехнічних елементів для САПР sPlan 7.x

к) 1 головний буферний вузол (Central Buffer Node);

л) 1 програма обробки виключень (Exception Handler);

м) 4 синхронізатора (Synchronization);

н) 2 потокових кінця (Flow Final);

п) 1 кінцева активність (Activity Final).

Для детального опису ДД та для повноцінного сприйняття її сутностей, розіб’ємо вихідну діаграму на декілька фрагментів та опишемо кожний з них (рис. 2.7 – 2.8).

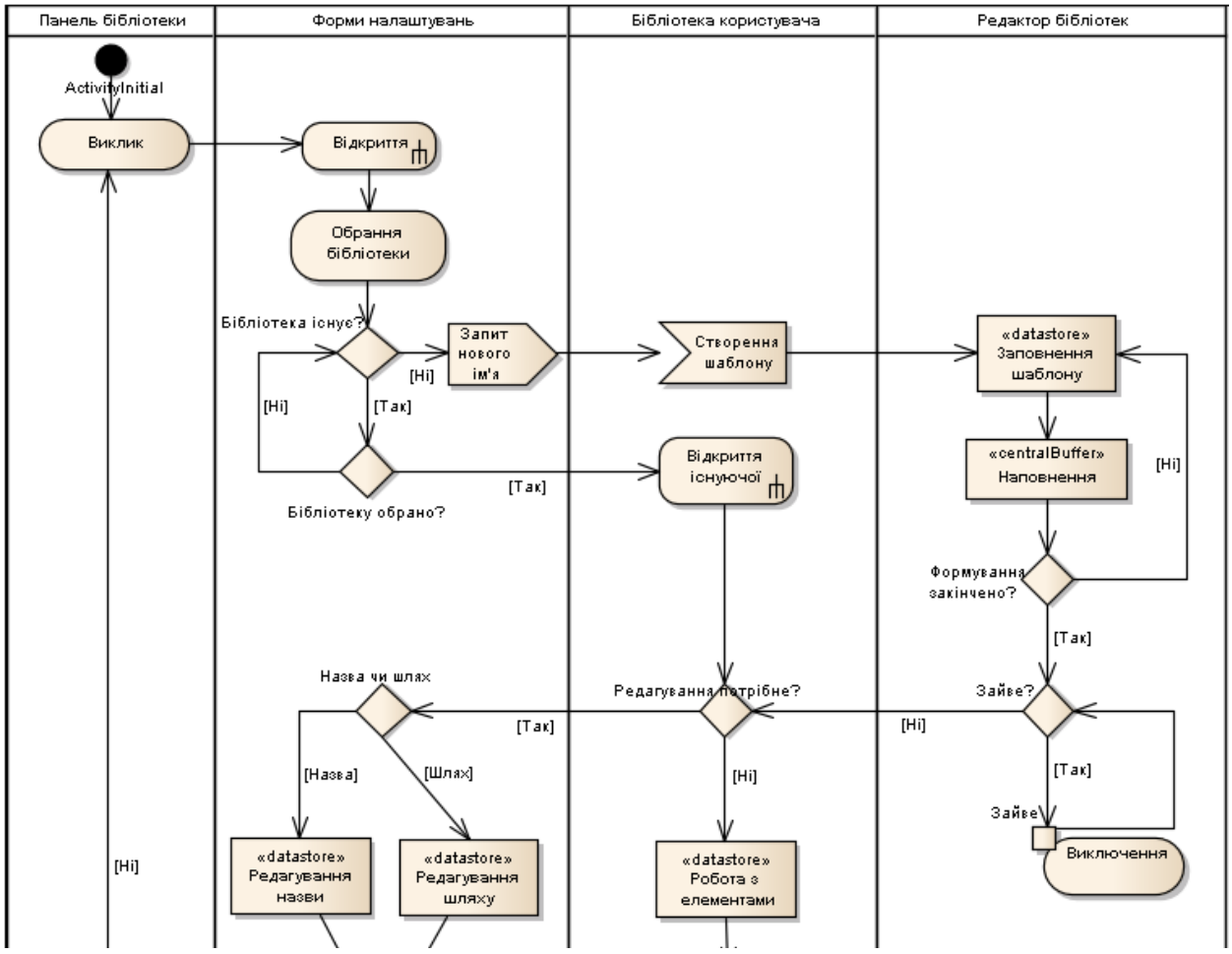

Рисунок 2.7 – Перший деталізований фрагмент ДД процесу створення бібліотек електротехнічних елементів для САПР sPlan 7.x

Перший фрагмент ДД (рис. 2.7) починається з розбиття області діаграми на сектори активності (Activity Partition). Таких на нашій ДД – чотири:

– «Панель бібліотеки»;

– «Форми налаштувань»;

– «Бібліотеки користувача»;

– «Редактор бібліотек».

Починається ДД стандартно на першому секторі активності (Activity Partition) «Панель бібліотеки» сутністю початкової активності (Activity Initial). Далі йде потік керування (Control Flow) до типової активності (Public Activity) «Виклик», що знаходиться на тому ж секторі. Це означає, що відбувається запит (виклик) панелі бібліотек.

Далі потік керування (Control Flow) переходить до другого сектору активності (Activity Partition) «Форми налаштувань» (рис. 2.7), а саме – до дії (Action) класу (Kind) поведінкового виклику (Call Behavior) «Відкриття», що означає відкриття спеціальної форми налаштувань.

Наступною йде типова активність (Public Activity) «Обрання бібліотеки», яка дозволяє обирати бібліотеки. Після неї потік керування (Control Flow) заходить до блоку-вирішення (Decision), у якому міститься питання «Бібліотека існує?». У разі позитивної відповіді – інформаційний потік слідує за обмежуючою умовою (Guard Condition) «Так», у разі негативної відповіді – за обмежуючою умовою «Ні».

Розглянемо, спочатку, позитивну гілку: інформаційний потік потрапляє до наступного блоку-вирішення (Decision), у якому міститься питання «Бібліотеку обрано?». У випадку «Ні» – інформаційний потік знову відправляється до попереднього блоку-вирішення, де відбувається остаточне з’ясування: чи існує бібліотека. У випадку «Так» – потік спрямовується до третього сектору активності (Activity Partition) «Бібліотека користувача», до дії (Action) класу (Kind) поведінкового виклику (Call Behavior) «Відкриття існуючої», що означає відкриття існуючої бібліотеки графічних елементів.

Далі інформаційний потік переходить до блоку-вирішення (Decision) (центральний блок ескізу 2.6), у якому міститься питання «Редагування пот-

рібне?» (рис. 2.7), до якого також дійде негативна гілка, яку ми оголосили вище, проте розглянемо її саме зараз.

Із блоку-вирішення (Decision) «Бібліотека існує?» – негативна гілка веде до спрямування (Send) «Запит нового ім’я», яке запитує у програміста нове ім’я для неіснуючої бібліотеки для початку процесу її створення. Далі – потік спрямовується до третього сектору активності (Activity Partition) «Бібліотека користувача», де потрапляє до приймача (Receive) «Створення шаблону» – саме цей приймач класифікує запити створення й видає найближчий рекомендований шаблон.

Після цього – потік спрямовується до четвертого сектору активності (Activity Partition) «Редактор бібліотек» (рис. 2.7), де потрапляє до накопичувача даних (Data Store) «Заповнення шаблону», що керує даними, що уводяться. Далі – до головного буферного вузла (Central Buffer Node) «Наповнення», що як раз й запам’ятовує усі вже відредаговані (остаточні) дані уводу до форми-шаблону.

Наступним, інформаційний потік заходить до блоку-вирішення (Decision), з питанням «Формування закінчено?», що дозволяє впевнитися у остаточному уводі даних, і якщо відповідь «Ні», то потік повертається на етап заповнення до блоку накопичувача даних (Data Store) «Заповнення шаблону». У позитивному випадку – до наступного блоку-вирішення.

Блок-вирішення (Decision), у якому міститься питання «Зайве?» – дозволяє керувати процесом редагування форми-шаблону. У випадку, якщо до форми потрапила зайва або проміжна інформація (позитивний потік виходу з блока-вирішення) – потік спрямовується до програми обробки виключень (Exception Handler) «Виключення зайвого», де відбувається процес видалення, а вже потім спрямування до повторного тестування до блоку «Зайве?». І так до того часу, поки форма не буде містити «зайвої» інформації.

У випадку відсутності зайвої інформації (негативний потік виходу з блока-вирішення) – інформаційний потік переходе до вже відомого блоку-вирішення (Decision), з питанням «Редагування потрібне?». У цьому блоці

сходяться ті дві гілки, які ми розглядали окремо. Крім того, цей блок керує процесами редагування тих бібліотек, що вже існують.

Розглянемо позитивну гілку.

Починається вона блоком-вирішенням (Decision), у якому міститься уточнення «Назва чи шлях», тобто до чого саме слід переходити до редагування назви чи шляху. В обох випадках – потоки потрапляють до накопичувачів даних (Data Store), де відбувається протоколювання процесів редагування; але у випадку «Назва» – потік спрямовується до накопичувача даних «Редагування назви», у випадку «Шлях» – до «Редагування шляху».

Далі – потоки об'єднаються (рис. 2.8) за допомогою горизонтального синхронізатора-з'єднувача (Synchronization) та потрапляють до потокового кінця (Flow Final) цього сектору активності (Activity Partition).

Розглянемо негативну гілку.

Інформаційний потік потрапляє до накопичувача даних (Data Store) «Робота з елементами» (рис. 2.7), де відбувається процес роботи саме вже зі створеними графічними елементами. Потім (за допомогою горизонтального синхронізатора- роз'єднувача (Synchronization), див. рис. 2.8), потік роз'єднується на дії (Action) з різноманітними класами (Kind) редагування, що говорять самі за себе та зрозумілі без додаткових пояснень:

– «Створення» (клас «Create Object»);

– «Редагування» (клас «Call Operation»);

– «Переміщення» (клас «Call Behavior»);

– «Імпортування» (клас «Accept Event»).

Після виконання дій редагування – потоки об'єднаються (рис. 2.8) за допомогою горизонтального синхронізатора-з'єднувача та потрапляють до потокового кінця (Flow Final) третього сектору активності (Activity Partition).

Слідом за цим, два потокових кінця з'єднуються у горизонтальному синхронізаторі-з'єднувачі, що розміщено на другому секторі активності (Activity Partition) «Форми налаштувань» та остаточний інформаційний потік потрапляє до блоку-вирішення (Decision), у якому міститься питання «Роботу закінчено?».

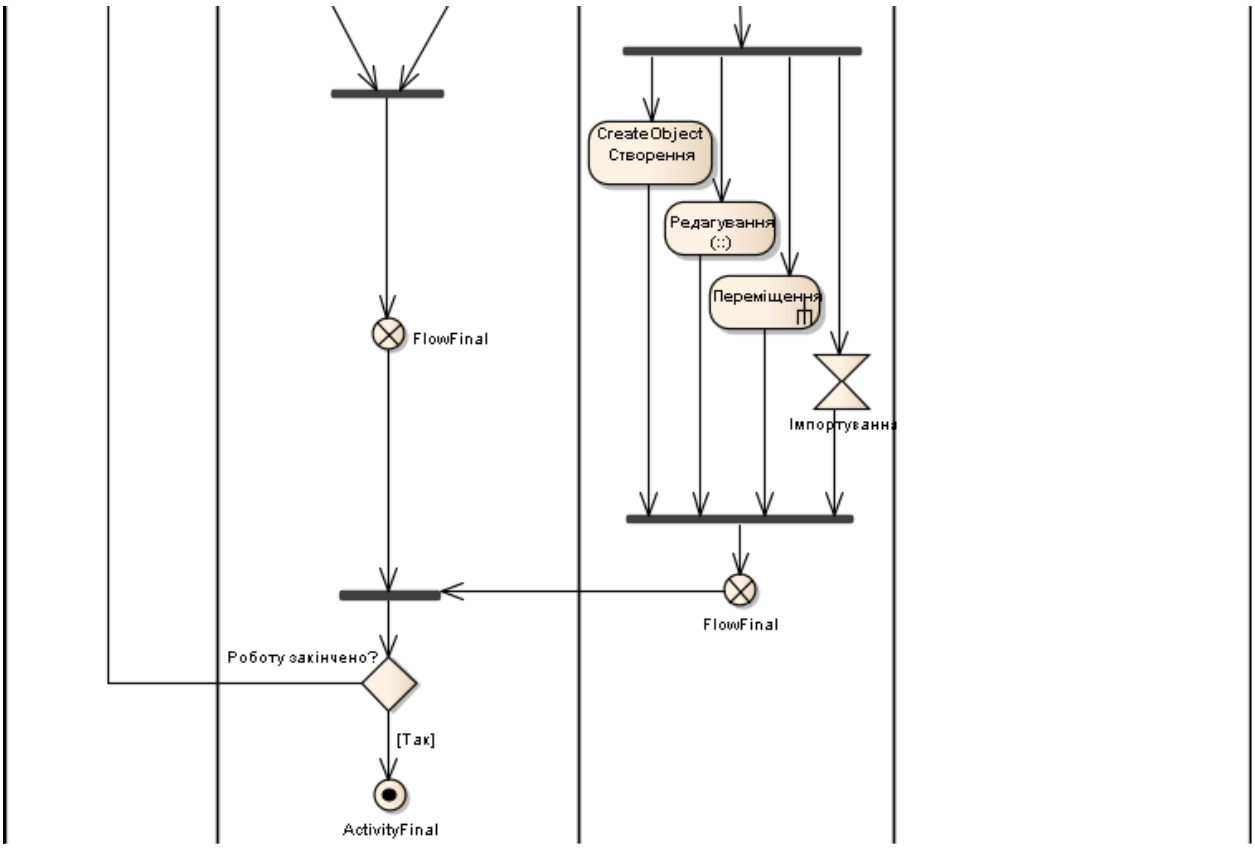

Рисунок 2.8 – Другий деталізований фрагмент ДД процесу створення бібліотек електротехнічних елементів для САПР sPlan 7.x

Як завжди – виходами з цього блоку – є дві гілки. Негативна, тобто «якщо роботу не закінчено», веде до першого сектору активності «Панель бібліотеки», а саме до типової активності (Public Activity) «Виклик», що означає формування нового запиту (виклику) панелі бібліотек (рис. 2.6). Позитивна («роботу – закінчено») веде до кінцевої активності (Activity Final) на ДД.

Таким чином, як результат проектування у даному підрозділі, було сформовано ДД для процесу створення бібліотек електротехнічних елементів для САПР sPlan 7.x.

2.7 Діаграма компонентів

Діаграми компонентів (Component diagram) – ДКМ – це статична структурна діаграма, яка показує розбивку програмної системи на структурні

компоненти та зв'язки (залежності) між компонентами [9]. В якості фізичних компонентів можуть виступати файли, бібліотеки, модулі, файли, що виконуються, пакети та ін.

ДКМ використовуються для моделювання компонентної структури розподілених додатків. Всередині – кожна компонента може бути реалізована за допомогою безлічі різноманітних класів та об’єктів [11].

Оскільки ДКМ, що буде повною мірою характеризувати компоненти системи, для поданої кваліфікаційної роботи – буде займати формат більший за А4 – наведемо, спочатку її ескіз, а потім деталізуємо фрагменти ДКМ.

Ескіз ДКМ бібліотек графічних елементів для САПР sPlan 7.x. – наведено на рис. 2.9.

Розширенні фрагменти деталізованого подання ДКМ бібліотек графічних елементів – наведено на рис. 2.10 – 2.11.

Спроектована ДКМ (рис. 2.9) складається із наступних сутностей:

а) 1 каркасна основа (Framework Package);

б) 1 конструктивне компонентне оформлення (Packaging Component);

в) 1 підсистема (subsystem);

г) 3 папки-пакети (Package);

д) 1 умовне обмеження (Boundary);

е) множинність типових компонентів (Component);

ж) множинність сторінок, що створюються (Web Page Artifact);

и) множинність елементів, що створюються (Script Library Artifact);

к) множинність зв’язків-залежностей (Dependency);

л) множинність зв’язків-збірок (Assembly).

Розглянемо детальніше її перший фрагмент (див. рис. 2.10) та надамо йому стислу характеристику.

Верхньою ієрархічною структурою – є каркасна основа (Framework Package) «sPlan 7.x», що являє собою інстальований пакет САПР. Саме до нього, в кінцевому випадку вкладено усі наступні спроектовані компоненти.

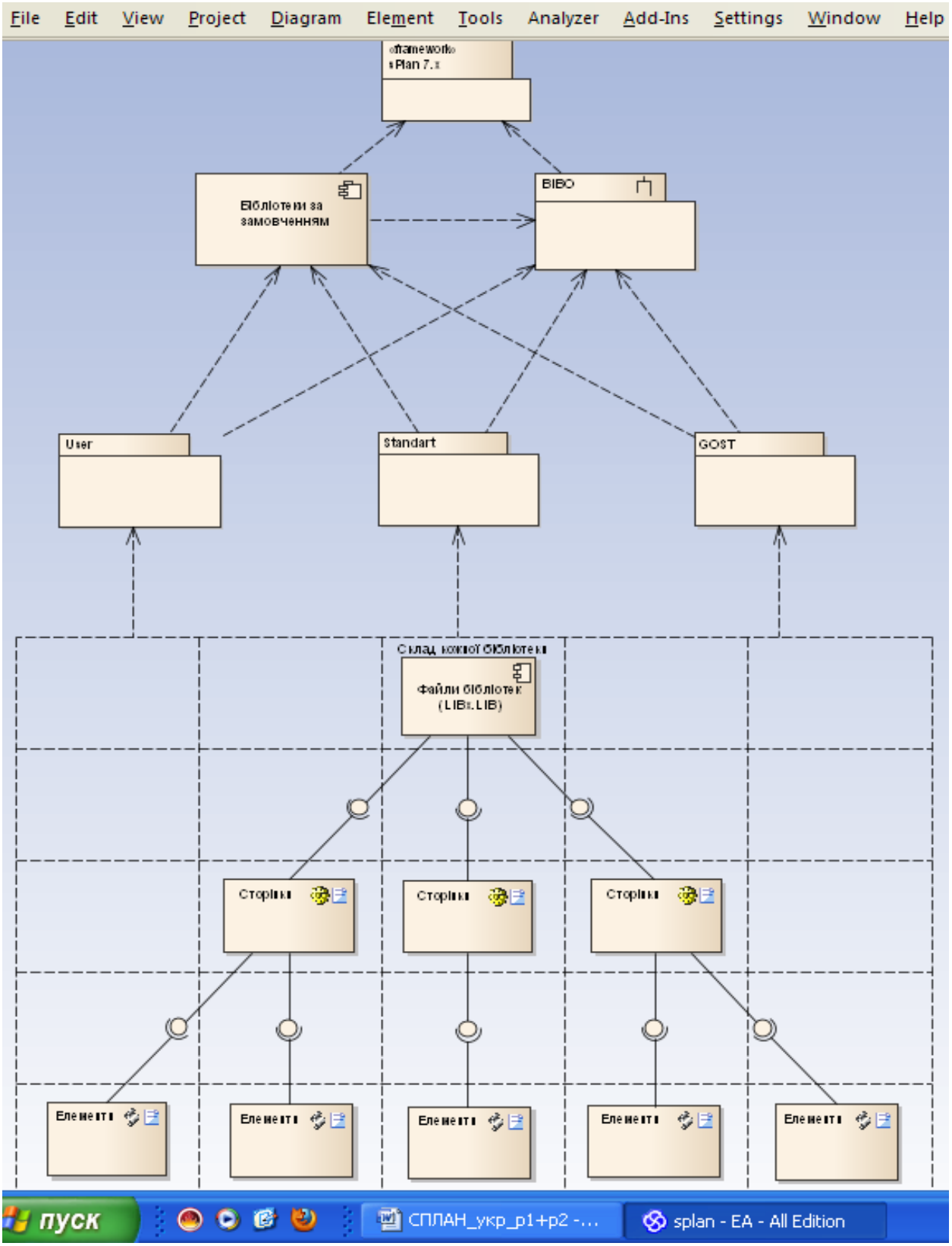

Рисунок 2.9 – Ескіз ДКМ бібліотек графічних елементів для САПР sPlan 7.x

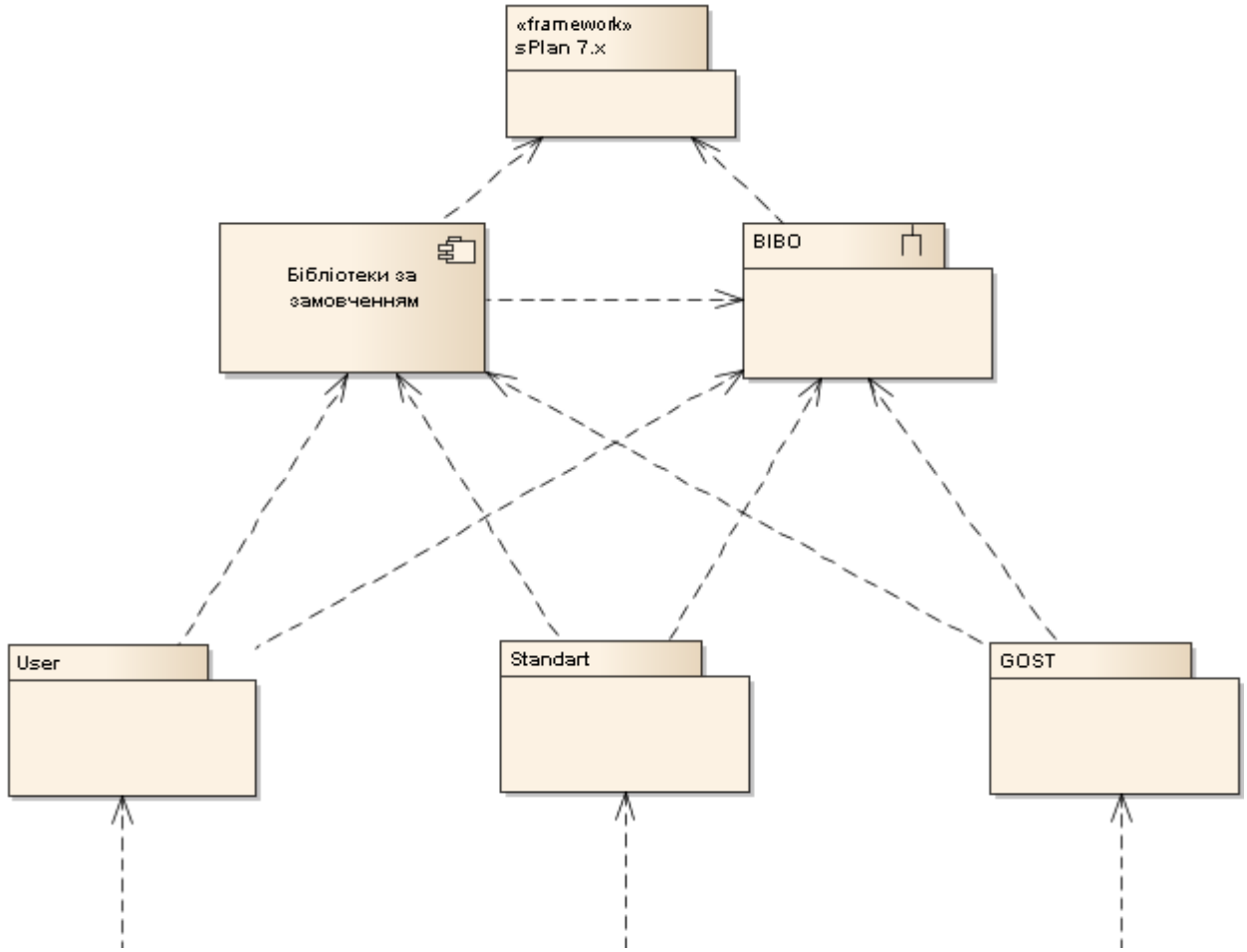

Рисунок 2.10 – Перший розширений фрагмент деталізованого подання ДКМ бібліотек графічних елементів для САПР sPlan 7.x

Першим таким компонентом – є конструктивне компонентне оформлення (Packaging Component) «Бібліотеки за замовченням», що по суті являє собою абстракцію, призначення якої – виділити наступну бібліотечну підсистему, до якої входить це конструктивне оформлення.

Підсистема (subsystem) «BIBO» – являє собою сукупність папок бібліотек із можливістю керування цими папками. При інсталяції, за замовченням, створюються папки-пакети (Package), що входять також до абстрактного компонентного оформлення (Packaging Component) «Бібліотеки за замовченням». Ці залежності (Dependency) – показані на розширеному фрагменті (рис. 2.10).

Такими залежними папками-пакетами (Package) є:

– «User»;

– «Standart»;

– «GOST».

Всі ці папки мають розгалужену структуру, із схожим компонентним оформленням, проте кожна з них призначена для відповідного сортування та зберігання окремих графічних бібліотек.

Так папка «User» зберігає тільки графічні бібліотеки користувача, тобто ті, які були створені, або імпортовані вже після інсталяції. Таким чином, при інсталяції САПР sPlan 7.x – ця папка створюється порожньою.

Папка «Standart» – містить стандартні графічні елементи, та створюється при інсталяції. За допомогою її – можна вибірково формувати бібліотеки користувача (папка «User»).

Папка «GOST» – являє собою бібліотеку стандартів щодо зображень графічних елементів. За допомогою її вмісту можна формувати бібліотеки користувача (папка «User»), проте додавати елементи до самої папки «GOST» без спеціального синхронізатору стандартів – заборонено. Ця опція потрібна для усунення помилкових змін та блокування викривлення еталонних стандартів.

Розглянемо далі умовне обмеження (Boundary) «Склад кожної бібліотеки» (рис. 2.11), що являє собою коміркову структуру, яка означає множинність кожного елемента, що до неї входить. Це умовне обмеження з’єднане із папками-пакетами (Package): «User», «Standart» та «GOST» зв’язками-залежностями (Dependency), що означають вкладеність усіх елементів умовного обмеження (Boundary) до відповідної папки-пакету.

Умовне обмеження (Boundary) «Склад кожної бібліотеки» містить у собі наступні (аналогічні для кожної з папок-пакетів) сутності:

а) множинність типових компонентів (Component);

б) множинність сторінок, що створюються (Web Page Artifact);

в) множинність елементів, що створюються (Script Library Artifact).

Усі ці сутності – об’єднані зв’язками-збірками (Assembly), що означають послідовний збір структури із цих сутностей з низу до гори.

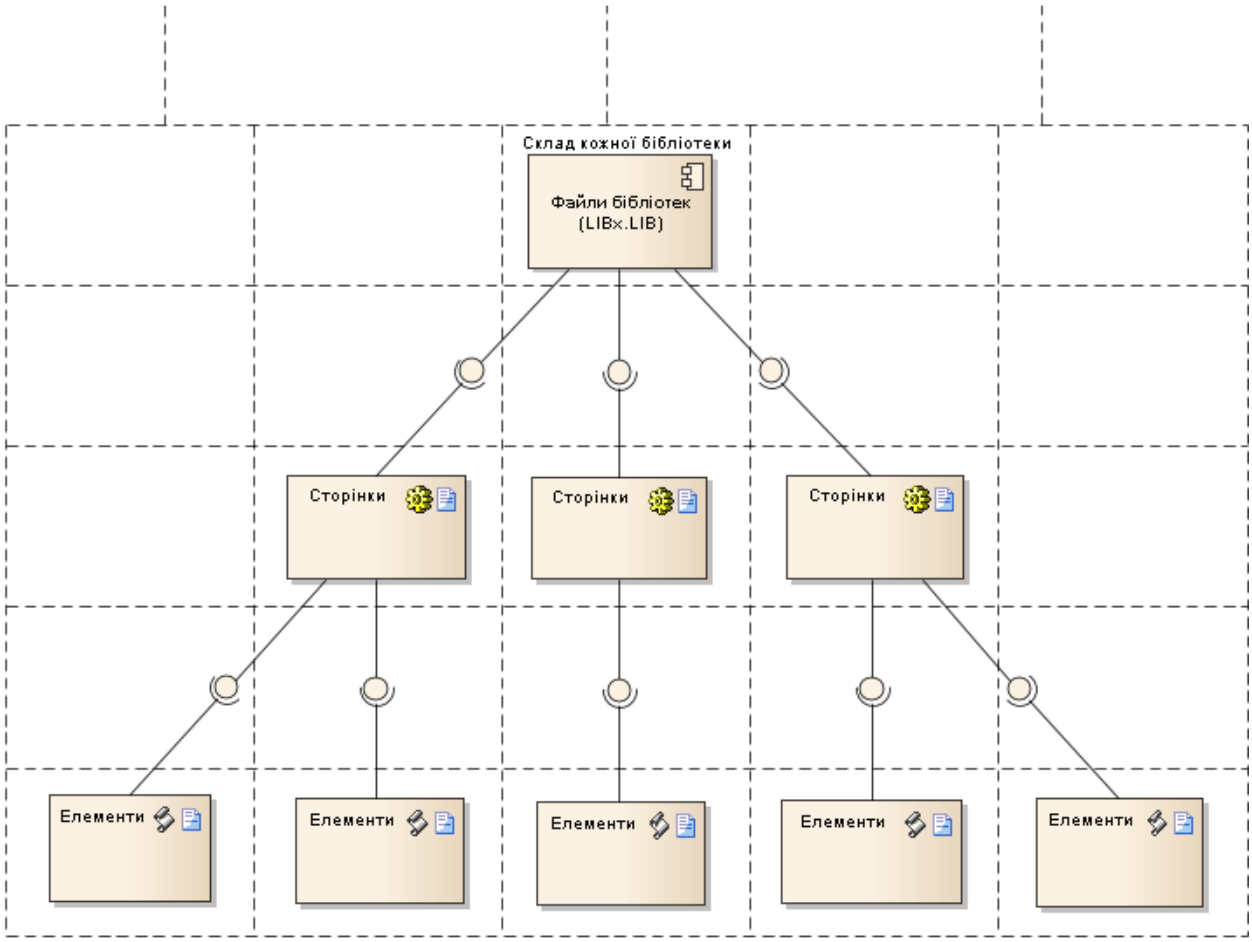

Рисунок 2.11 – Другий розширений фрагмент деталізованого подання ДКМ бібліотек графічних елементів для САПР sPlan 7.x

Деталізуємо пояснення наведеної структури. Множинність типових компонентів (Component) «Файли бібліотек (LIBx.LIB)» – являють собою сукупність графічних бібліотек, що формують папки-пакети (Package): «User», «Standart» та «GOST». Ці бібліотеки при їх формуванні позначуються порядковими номерами та відрізняються при ідентифікації за допомогою змінення розряду «X» у назві LIBx.LIB.

Компоненти «Файли бібліотек» збираються (зв’язками-збірками (Assembly)) з множинності сторінок, що створюються (Web Page Artifact) «Сторінки», які по суті являють собою сторінки бібліотек.

Ці сторінки бібліотек збираються за допомогою зв’язків-збірок (Assembly) з множинності елементів, що створюються (Script Library Artifact) «Еле-

менти», кожний з яких – є представлення окремого графічного елемента із сукупністю його властивостей.

Таким чином, при виконанні поданого підрозділу – було спроектовано ДКМ бібліотек графічних елементів для САПР sPlan 7.x із наведенням її структури та деталізацією фрагментів.

2.8 Висновки за розділом

Під час виконання проектного розділу кваліфікаційної роботи спеціаліста було побудовано проектний каркас (архітектуру) майбутніх функцій розширення бібліотек графічних елементів САПР sPlan 7.x бібліотеками електротехнічних елементів.

Результатом виконання розділу стало створення наступних діаграм:

а) варіантів використання;

б) послідовності;

в) об’єктів;

г) класів;

д) станів;

е) діяльності;

ж) компонентів.

При формуванні проектної архітектури було використано розширену нотацію UML 2.5 та CASE-інструментарій Enterprise Architect 14.0.

3 РОЗДІЛ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

У третьому розділі наводиться детальний опис етапів програмної реалізації створення бібліотек даних електротехнічних елементів для відкритої САПР sPlan 7.x. Цей розділ виконано на основі розробленої (у другому розділі кваліфікаційної роботи) системної архітектури.

3.1 Стисла характеристика середи проектування

У якості об’єкту кваліфікаційної роботи – було обрано середу проектування sPlan – отже наведемо її стислу характеристику.

Програмне середовище sPlan являє собою безкоштовну автоматизовану систему проектування для створення електричних креслень. Цей графічний редактор добре працює з векторною графікою і має всі необхідні інженеру-електрику функції.

sPlan дозволяє переводити векторну графіку у растрову і виконувати її друк в різному масштабі. Головною функцією програми sPlan – є проектування та розробка електронних принципових схем. Спеціально для цього, розробниками зроблена вельми велика база різноманітних геометричних заготовок – позначення електронних елементів. Також – є всі основні елементи, що стануть у нагоді при створенні принципової електричної та електронної схеми.

Якщо, раптом необхідного елемента немає, то його можна зробити з простих фігур, а після зберегти в основний чи спеціально створеній бібліотеці. Дослідження саме такої ситуації і є метою кваліфікаційної роботи.

Уся графіка у САПР sPlan спирається на сітку, що в безпосередньо прив'язана до горизонтальної та вертикальної лінійкиі. Це дозволяє, переміщати об'єкти тільки на певні відстані (за замовчуванням – це 1 мм), а за необхідністю – переміщати елементи на вільну відстань.

3.2 Вбудована бібліотека елементів САПР sPlan 7.x.

Символи та елементи – це найнеобхідніші речі для виконання креслення. У лівій частині інтерфейсу вікна редактора – можна знайти велику бібліотеку з символами і елементами, що розташовані на декількох сторінках (рис. 3.1).

У САПР sPlan можливо виконувати наступні операції із бібліотекою:

а) додавання нових сторінок:

б) перенумерація сторінок;

в) видалення сторінок;

г) видалення символів;

д) переміщення символів на сторінці;

е) переміщення символів на іншу сторінку;

ж) додавання нових бібліотек.

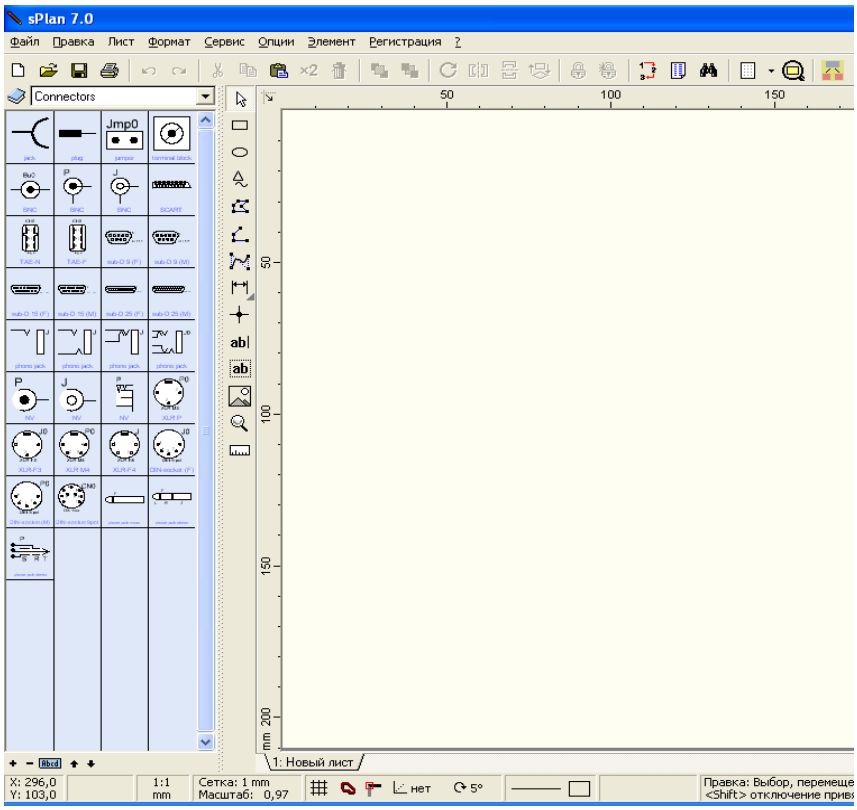

Рисунок 3.1 – Вигляд бібліотеки елементів у інтерфейсі САПР sPlan 7.x, що містить символи та елементами, які розташовано на декількох сторінках

Сторінки бібліотеки розташовані в алфавітному порядку, їх список знаходиться у верхній частині вікна бібліотеки (рис. 3.1).

3.3 Обрання бібліотек

У САПР sPlan – є можливість працювати із декількома бібліотеками, виконуючи перемикання між ними щигликом мишки на іконці з зображенням книги (рис. 3.2).

Якщо необхідно виконати налаштування бібліотеки, або створення нової – слід зайти до пункту меню «Бібліотеки». При цьому – відкриється діалогове вікно.

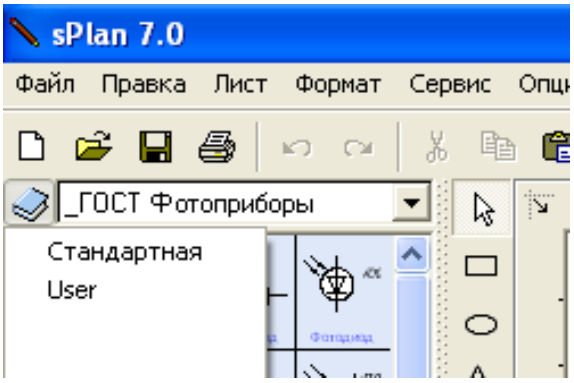

Рисунок 3.2 – Перемикання між бібліотеками користувача у САПР sPlan 7.x

Подане діалогове вікно – містить всі наявні бібліотеки, слід обрати потрібну і клацнути мишкою на кнопку «Вибрати» або просто зробити подвійне клацання мишкою на імені бібліотеки – тоді вона відкриється.

Якщо необхідної бібліотеки не існує, то необхідно створити її, з початку, виконавши запит на створення, шляхом натискання кнопки керування «Створити».

Слід обрати необхідну бібліотеку, якщо потрібно, то можна змінити шлях до неї, клацнувши мишкою на кнопці з трикрапкою «...» та / або змінити ім'я. бібліотеки.

Для видалення бібліотеки – служить кнопка «Видалити», що відкриває діалог підтвердження цієї операції, але навіть після видалення сама бібліотека залишиться на диску, де міститься САПР. У такому разі, буде видалено тільки ім'я бібліотеки у списку. Тобто зберігається можливість нового уведення цієї бібліотеки до списку, якщо це знадобиться.

3.4 Керування бібліотеками

Керування бібліотеками елементів доступно із меню, що розташоване у самому низу вікна бібліотеки (рис. 3.3).

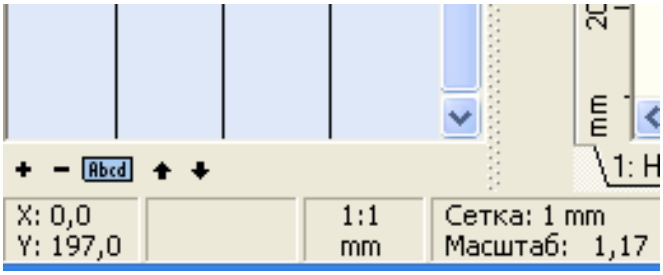

Рисунок 3.3 – Керування бібліотеками елементів за допомогою нижнього меню

За допомогою цього меню, можна обирати кількість колонок, масштаб, показ імен елементів, визначити шлях до бібліотеки або вибрати іншу.

Так, якщо розглянути рис. 3.3: кнопка імен елементів «Abcd», скриває або проявляє підписи імен під елементами (так на рис. 3.3 – імена – проявлено, проте, якщо подана кнопка зображена перекресленою, тоді імена – скрито; але навіть якщо імен не видно, то при наведенні курсору на елемент, його ім'я з'являється у спливаючому вікні – це зручно при великій кількості елементів у бібліотеці);

Ширина вікна бібліотеки може бути змінена приміщенням курсору на кордон бібліотеки до появи подвійної стрілки, після чого можна обрати бажану ширину вікна, утримуючи мишку та пересуваючи межу вікна.

3.5 Редагування бібліотек

САПР sPlan має можливість редагування бібліотек, за допомогою простих операцій «Копіювання», «Переміщення» або «Видалення» сторінок. Для початку процесу редагування, необхідно:

а) клацнути на іконку із зображенням книги (рис. 3.2);

б) зайти до пункту меню «Бібліотеки» (рис. 3.2);

в) у діалоговому вікні – клацнути мишкою на кнопці «Налаштувати»;

г) після цього – з'явиться вікно «Керування бібліотеками» (рис. 3.3).

Тепер можна змінювати вміст двох бібліотек. Треба обрати дві бібліотеки зі списку «Бібліотека 1» та «Бібліотека 2». Наприклад: «Стандартна» та «User» (рис. 3.4).

У лівому і правому вікні діалогу будуть видні всі сторінки бібліотеки (рис. 3.4). Можна скопіювати або перемістити потрібні сторінки між двома бібліотеками. Натискання кнопки «Вилучити» негайно видалить сторінку підсвіченого елемента із бібліотеки.

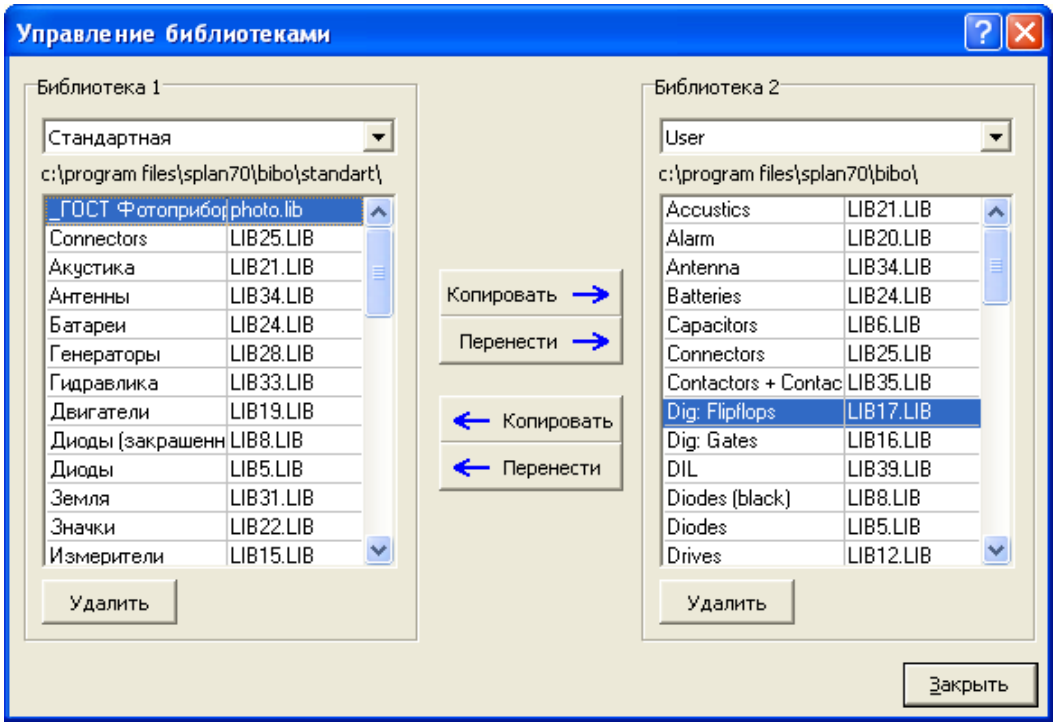

Рисунок 3.4 – Вікно «Керування бібліотеками»

3.6 Редагування сторінки бібліотеки

3.6.1 Додавання нового елементу

Для того, щоб додати новий елемент в бібліотеку, необхідно:

а) викликати контекстне меню у головному представленні бібліотек;

б) обрати «Створити новий компонент» у контекстному меню;

в) внести зміни до діалогового вікна «Властивості елементу».

У відповідних полях «Властивостей елементу» слід уводити дані користувача або обирати із тих, що пропонуються. Так, у полі «Позначення» – слід увести ідентифікатор елемента, який планується або (за допомогою реалізованої функції розширення «…») увести розширений формат (рис. 3.5).

У особливих випадках, за допомогою убудованих функцій, можна формувати розширені формати, що складаються із складеного переліку:

а) фіксована змінна;

б) змінна користувача;

в) текстова константа.

Приклад формування тексту із сформованого набору фіксованих змінних приведено на рис. 3.6.

Параметри «Змінна користувача» та «Текстова константа» визначаються за допомогою спеціальних функцій, які закріплено за відповідними кнопками «Визначити».

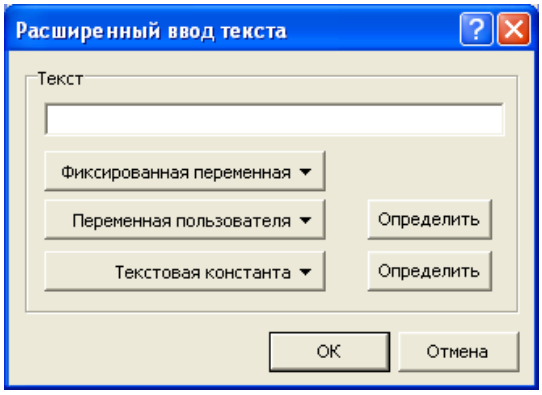

Рисунок 3.5 – Реалізована підсистема уводу розширеного формату тексту

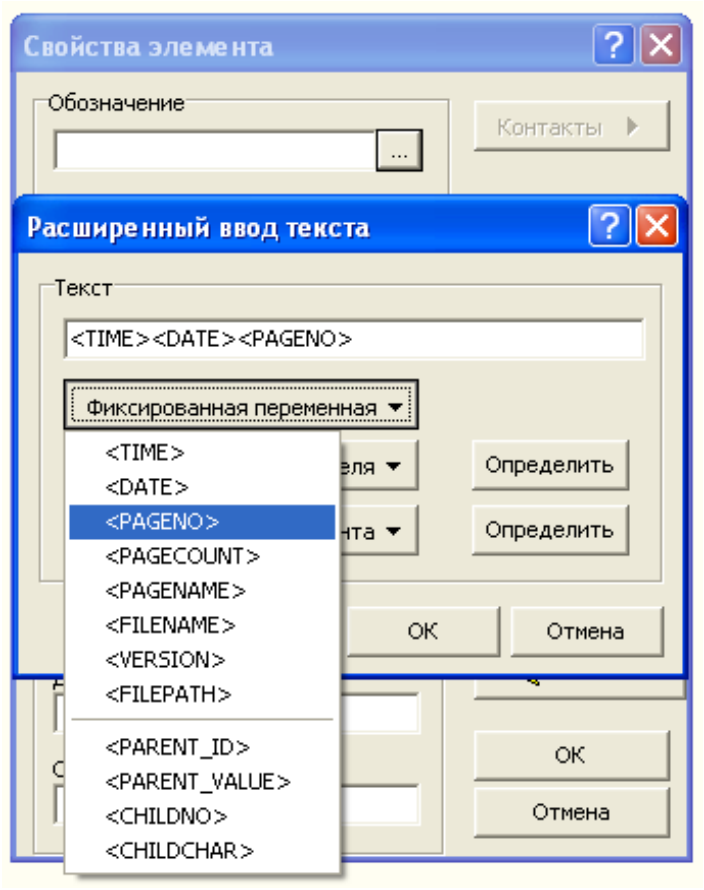

Рисунок 3.6 – Формування тексту із переліку фіксованих змінних

3.6.2 Редагування графічних елементів

Якщо поглянути на «Властивості елементу», то можна помітити, що до властивостей було додано кнопку «Редактор». Натискання цієї кнопки призводить до виклику одного з найголовніших інструментів САПР sPlan 7.x – графічного редактора. Приклад виклику графічного редактора подано на рис. 3.7 для елементу «7-ми сегментовий індикатор».

3.6.3 Перейменування сторінок

Для того, щоб перейменувати вже існуючу сторінку, необхідно:

а) викликати контекстне меню у головному представленні бібліотек;

б) обрати пункт «Перейменувати сторінку» у контекстному меню;

в) внести зміни до діалогового вікна «Властивості розділу».

Подібну структури має процедура «Видалити сторінку бібліотеки», тільки для цього треба обрати відповідний пункт. При цьому, слід звернути особливу увагу на те, що після підтвердження видалення – сторінка буде видалена без можливості відновлення.

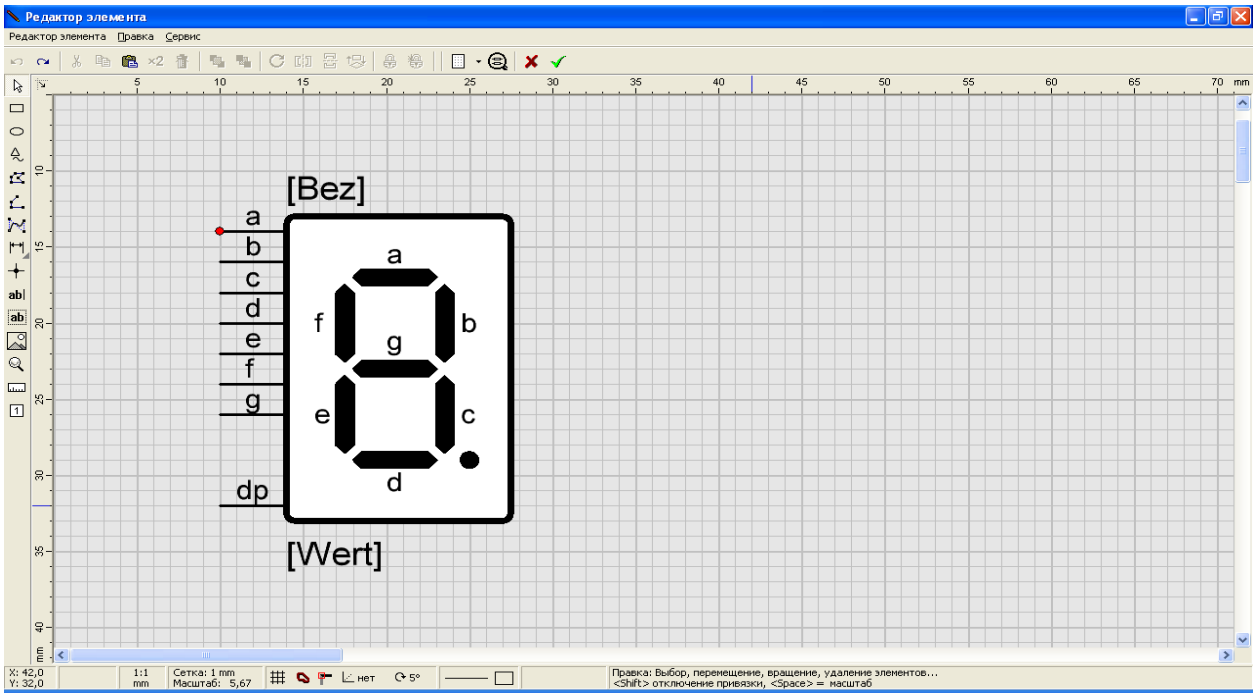

Рисунок 3.7 – Приклад графічного редагування елементу бібліотеки

3.7 Файли бібліотек

При роботі з САПР sPlan, можна використовувати кілька бібліотек, кожна з яких може складатися з декількох сторінок (рис. 3.8).

Бібліотека, що знаходиться у тій чи іншій директорії на диску, містить кілька сторінок, кожна з яких являє собою окремий файл.

Дві бібліотеки інсталюються за замовчуванням разом з програмою sPlan (рис. 3.9):

а) «Standard» – бібліотека за замовчуванням, що знаходиться у директорії BIBO, тобто в папці самої програми sPlan;

б) «User» – бібліотека створюється самим користувачем і знаходиться у піддиректорії USER.

Обидві бібліотеки містять декілька файлів, кожен файл включає в себе кілька сторінок. Коли до бібліотеки додається нова сторінка, то створюється новий файл з ім'ям LIBx.LIB, де «x» – порядковий номер.

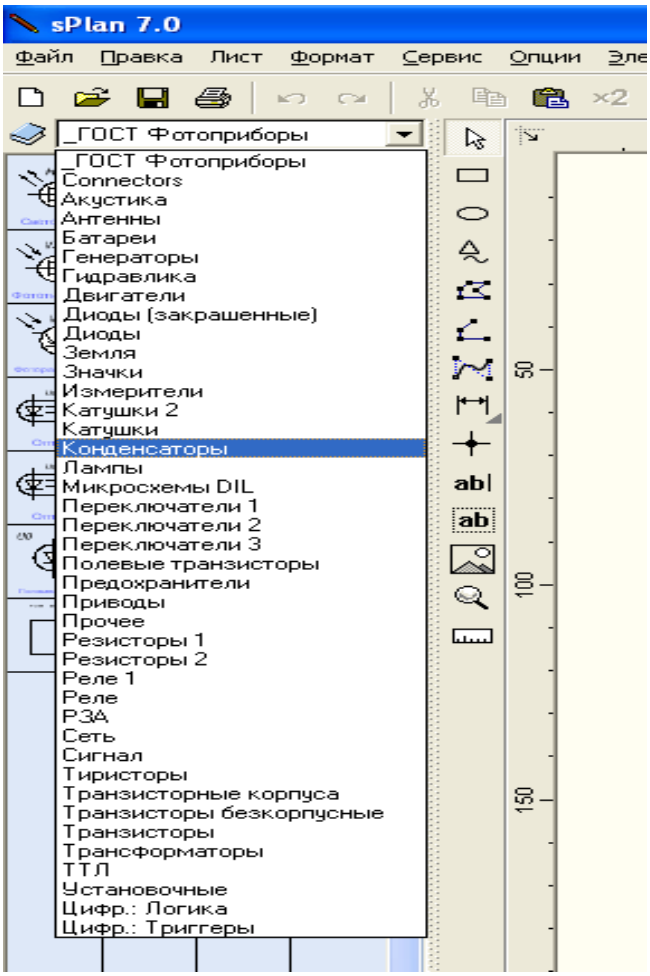

Рисунок 3.8 – Перелік вбудованих у САПР sPlan 7.x бібліотек

3.8 Інформація про сторінки

Для отримання інформації про поточну сторінку бібліотеки – використовується функція «Інфо сторінки бібліотеки», що викликається через контекстне меню головного представлення бібліотек. Екранна форма її приведена на рис. 3.10.

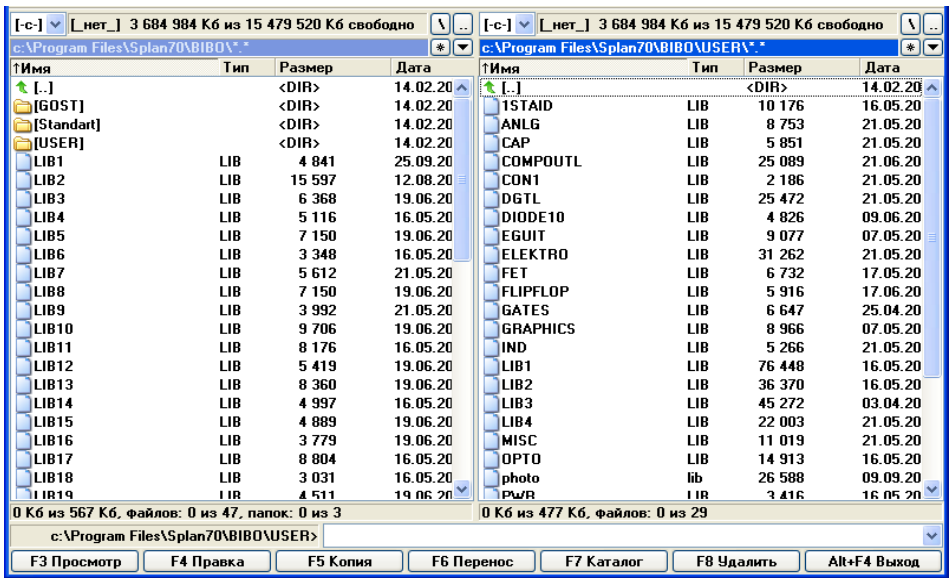

Рисунок 3.9 – Розміщення та склад графічних бібліотек «Standard» та «User»

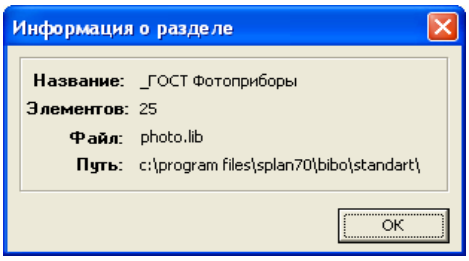

Рисунок 3.10 – Екранна форма функції «Інфо сторінки бібліотеки»

Ця функція несе інформацію про:

а) назву сторінки;

б) кількість елементів;

в) назва файлу;

г) шлях до файлу.

3.9 Імпорт файлів бібліотек

Нові файли бібліотек можна скопіювати до будь-якої директорії з існуючими бібліотеками, але , у цьому випадку, слід бути особливо уважним при заміні вже наявних файлів.

Іноді бібліотека може мати розширення, таке як креслення – «*.SPL». Це означає, що бібліотека виконана у формі креслення, або просто у вигляді групи елементів (рис. 3.11).

У цьому випадку, необхідно відкрити креслення та вручну додати його елементи до бібліотеки користувача.

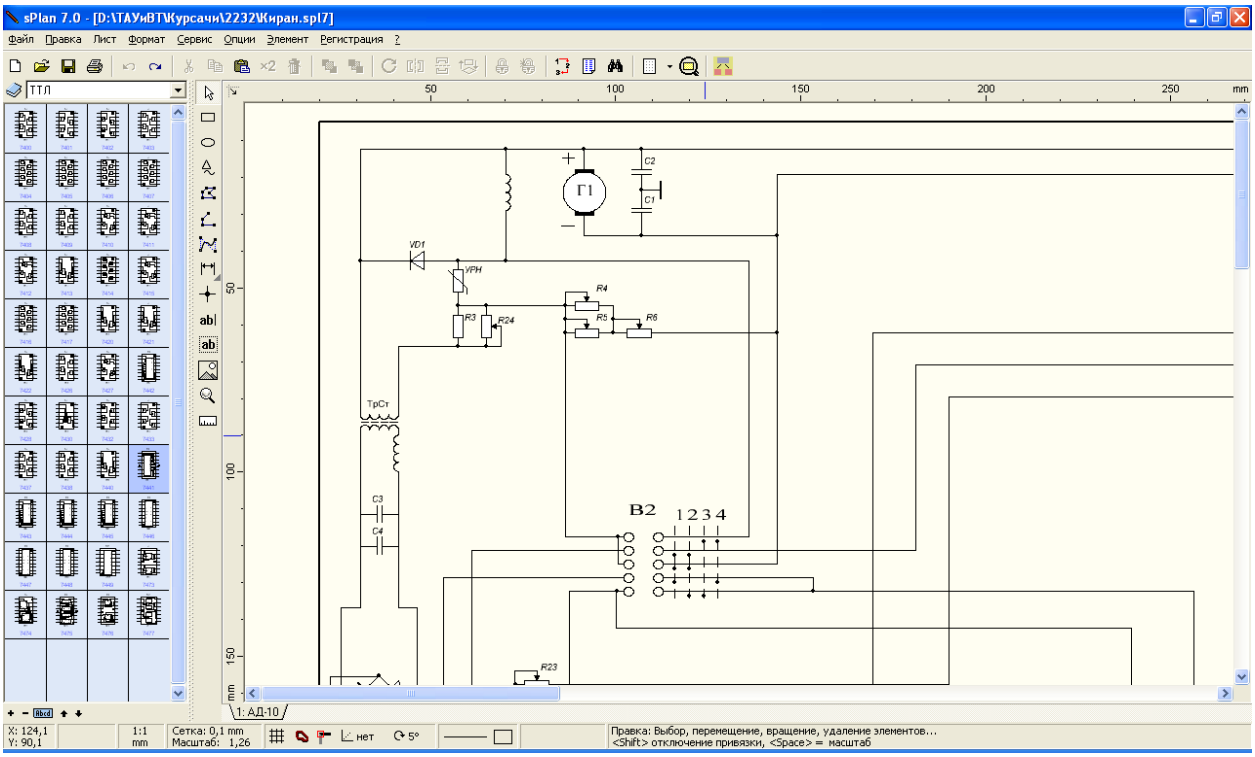

Рисунок 3.11 – Приклад імпортованої бібліотеки, яка виконана у формі креслення, що містить групи електротехнічних елементів

3.10 Операції з бібліотеками

Для того, щоб мати резервну копію бібліотек – необхідно скопіювати всі файли «*. LIB» в окрему директорію, також можна помістити їх до архіву «ZIP» або «RAR», щоб після переінсталяції САПР sPlan, була можливість відновлення бібліотеки з резервної копії архіву.

Якщо було створено власні бібліотеки користувача, то можна знову вказати шлях до них.

Для перейменування файлів бібліотек необхідно використовувати будь-який файловий менеджер. Перейменування файлів може знадобитися при їх подальшому пересиланні.

Наприклад: ім'я «TUBES.LIB» («Лампи.LIB») більш інформативно, ніж «LIB12.LIB». Але треба не забути про збереження розширення «*.LIB» при виконанні перейменування.

3.11 Створення нових бібліотек

Створення принципово нової бібліотеки може знадобитись, якщо вже наявна бібліотека занадто велика (більше 50 елементів). Приклад такої ситуації наведено на рис. 3.12.

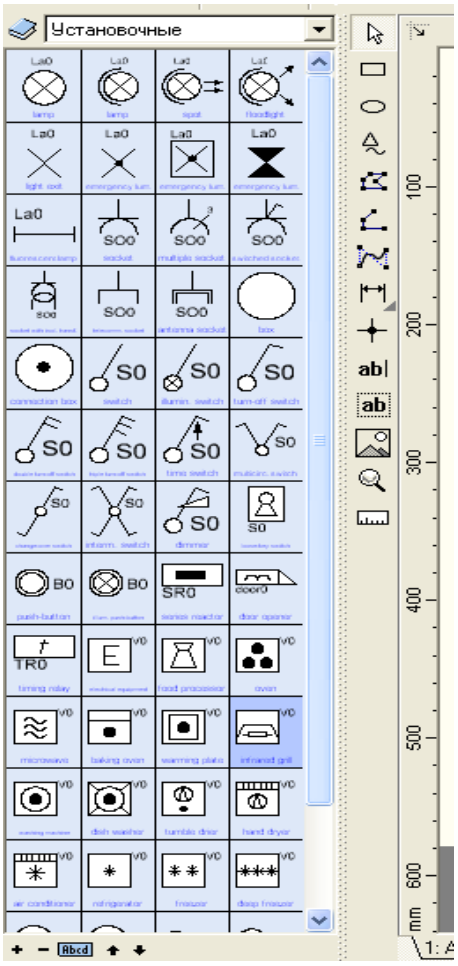

Рисунок 3.12 – Приклад ситуації із занадто великою кількістю елементів

Перше, що необхідно зробити – це створити нову директорію на диску або у мережі, при цьому використовуючи будь-який файловий менеджер.

Друге – скопіювати наявні файли бібліотек «*.LIB» в цю директорію або тимчасово залишити її порожньою – цю директорію можна використовувати як нову бібліотеку користувача.

Після визначення шляху до директорії, слід увести нове потрібне ім'я бібліотеки у формі, що розташована нижче за форму уводу шляху до бібліотеки (рис. 3.13).

Тепер вже можна використовувати нову (створену) бібліотеку користувача, як і решту.

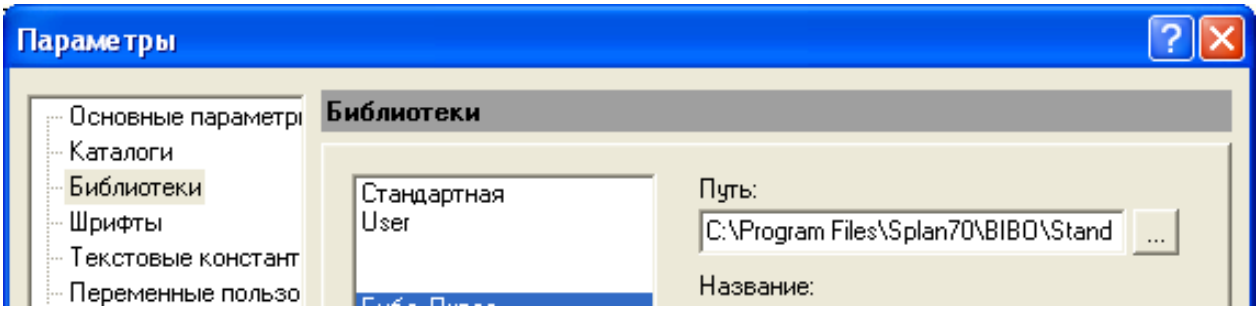

Рисунок 3.13 – Формування нового ім'я бібліотеки користувача

ВИСНОВКИ

В процесі виконання кваліфікаційної роботи – було досягнуто мету, а саме: виконано розробку та спроектовано технологію сполучення бібліотеки електротехнічних елементів із САПР sPlan 7.x.

Для досягнення мети, у спеціальній частині роботи було вирішено низку встановлених задач:

– виконано огляд спеціалізованих САПР для двомірного проектування;

– розглянуто архітектуру sPlan 7.x та складено її проектний каркас;

– визначено взаємозв’язок нової бібліотеки із вже існуючими;

– виконано тестування нової бібліотеки у вигляді виконання контрольних електротехнічних креслень.

Після досліджень САПР sPlan – можна зробити наступні висновки, що подана САПР спеціально розроблена для креслення електричних, радіоелектронних схем та інших простих креслень.

Розглянута САПР, має багату бібліотеку готових радіотехнічних елементів і шаблони для креслень. У САПР є практично всі можливості, необхідні інженеру чи звичайному користувачеві для формування різноманітних схем або креслень.

Для створення електричних схем – є декілька бібліотек, елементи яких можуть редагуватися і додаватися, присутні функції автоматичної нумерації елементів, прив'язки ліній до виводів елементів, групування елементів, прив'язки до сітки, можливість малювання ліній під певним кутом, поворот елементів, імпорт рисунка, експорт сформованого креслення у формати «JPG», «BMP», «GIF» та «EMF».

Як і більшість нинішніх графічних редакторів, САПР sPlan має добру можливість скасування зроблених дій, а також, при необхідності, їх повторення. Позитивним додатком до цього, є те, що кожному компоненту креслення, можливо, привласнити своє ім'я, особистий номінал і бажаний опис.

Особливістю даного графічного редактора є його багатогранність, тобто, одне креслення цілком може містити у собі кілька різних листів, що можна при необхідності зберегти в один єдиний файл, як окремо, так і разом.

Створення власних елементів користувача – не представило суттєвої проблеми, для цього спроектовано спеціальний редактор елементів, крім того, представлено технологію імпортування, за якою зручно переносити графічні елементи і цілі фрагменти схем з раніш створених креслень.

САПР sPlan 7.x містить набори зручних та різноманітних функцій, використання яких може бути розширене і не обмежено тільки потребами створення креслень і схем.

Таким чином, САПР sPlan 7.x – допоможе створити якісне представлення даних електричних та електротехнічних схем. До редактору додано інструменти для креслення і редагування, які роблять розробку схем зручною та ефективною, а простий та зрозумілий інтерфейс та малий інсталяційний розмір – це позитивні переваги цієї САПР.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Dip Trace. Schematic and PCB design software. URL: https://www.diptrace.com/ua/ (дата звернення: 18.03.2023).
2. ElectriCS PRO. URL: https://csoft.shop/catalog/items/item_121122.html (дата звернення: 19.03.2023).
3. nanoCAD. Engineering Ecosystem. URL: https://nanocad.com/ (дата звернення: 20.03.2023).
4. nanoCAD 22 Platform. URL: https://www.youtube.com/watch?v=z2Mv3iNjhUg (дата звернення: 21.03.2023).
5. EAGLE / Fusion 360. URL: https://www.autodesk.com/products/eagle/overview?term=1-YEAR&tab=subscription (дата звернення: 22.03.2023).
6. Static Free Software. Home of the Electric VLSI Design System. URL: https://www.staticfreesoft.com/ (дата звернення: 23.03.2023).
7. KiCad EDA. A Cross Platform and Open Source Electronics Design Automation Suite. URL: https://www.kicad.org/ (дата звернення: 24.03.2023).
8. Великодний С. С., Тимофєєва О. С. Реінжиніринг програмного забезпечення інформаційних систем. Монографія. Одеса: Видавничий дім «Гельветика», 2020. 162 с.
9. Velykodniy S., Burlachenko Zh., Zaitseva-Velykodna S. Modelling the behavioural component of the emergent parallel processes of working with graph databases using Petri net-tools. International Journal of Parallel, Emergent and Distributed Systems. (Scopus) 2021. Vol. 36. Iss. 6. P. 498-515. DOI: https://doi.org/10.1080/17445760.2021.1934836. Taylor & Francis Group, England & Wales. London.

10. Великодний С. С. Ідеалізовані моделі реінжинірингу програмних систем. Радіоелектроніка, інформатика, управління. 2019. № 1. С. 150–156. DOI: 10.15588/1607-3274-2019-1-14.
11. Великодний С. С. Моделі та методи проактивного управління проєктами із розвитку програмних систем і продуктів. Монографія. Одеса: Одеський державний екологічний університет, 2021. 322 с. ISBN 978-966-186-182-3. (URL: http://eprints.library.odeku.edu.ua/id/eprint/ 9595/)
12. Velykodniy S. S. Analysis and synthesis of the results of complex experimental research on reengineering of open CAD systems. Applied Aspects of Information Technology. 2019. Vol. 2. No 3. P. 186–205.
13. Великодний С. С., Бурлаченко Ж. В., Зайцева-Великодна С. С. Реінжиніринг графічних баз даних у середовищі відкритої системи автоматизованого проектування BRL-CAD. Моделювання структурної частини. Вісник Кременчуцького національного університету ім. Михайла Остроградського. 2019. Вип. 3 (116). С. 130–139. (кат. «Б») DOI: 10.30929/1995-0519.2019.3.130-139.
14. Petukhin D.,Velykodniy S., Kozlovskaya V. Modeling the space of possible states of the lesson schedule in higher education institutions. International Scientific and Practical Conference "Intellectual Systems and Information Technologies", 13-19 sep. 2021, Odesa, Ukraine. PP. 230–237.
15. Великодний С. С. Моделювання складних процесів та систем (Частина 1): конспект лекцій. Одеса: Одеський державний екологічний університет, 2021. 92 с. ISBN 978-966-186-181-6. http://eprints.library.odeku.edu.ua/id/eprint/9494/
16. Великодний С. С., Бурлаченко Ж. В., Зайцева-Великодна С. С. Розробка архітектури програмного засобу для управління мережевим плануванням реінжинірингу програмного проекту. Сучасний стан

наукових досліджень та технологій в промисловості. 2019. № 2 (8). С. 25–35.

17. Velykodniy S., Burlachenko Zh., Zaitseva-Velykodna S. Software for automated design of network graphics of software systems reengineering. Scientific Journal Herald of Advanced Information Technology. 2019. No 2 (03). P. 20–32.
18. Великодний С. С., Бурлаченко Ж. В., Зайцева-Великодна С. С. Реінжиніринг графічних баз даних у середовищі відкритої системи автоматизованого проектування BRL-CAD. Моделювання поведінкової частини. Вісник Кременчуцького національного університету ім. Михайла Остроградського. 2019. Вип. 2 (115). С. 117–126.
19. Невлюдов И. Ш., Великодный С. С., Омаров М. А. Использование CAD/CAM/CAE/CAPP при формировании управляющих программ для станков с ЧПУ. Восточно-Европейский журнал передовых технологий. 2010. № 2/2 (44). С. 37–44.
20. Великодний С. С., Тимофєєва О. С., Зайцева-Великодна С. С. Метод розрахунку показників оцінки проекту при виконанні реінжинірингу програмних систем. Радіоелектроніка, інформатика, управління. 2018. № 4. С. 135–142.
21. Великодний С. С., Тимофєєва О. С. Спосіб мультилінгвістичного перекодування програмного забезпечення складних інформаційних систем та технологій. Наукові праці ОНАЗ ім. О. С. Попова. 2017. № 2. С. 153–159.

ДОДАТОК А

НАОЧНІ ПРИКЛАДИ ГРАФІЧНИХ ІНТЕРФЕЙСІВ ТА ПРИНЦИПІВ РОБОТИ САПР-АНАЛОГІВ

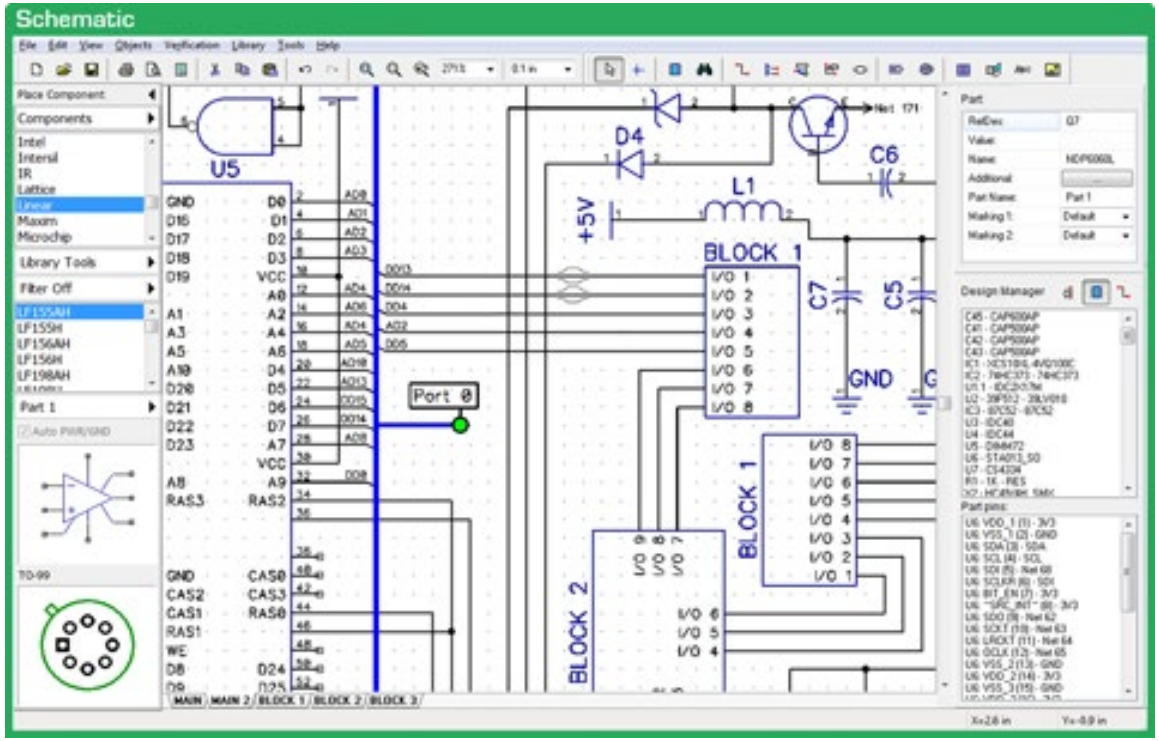

Рисунок А.1 – Приклад розведення зв’язків у програмному комплексі DipTrace

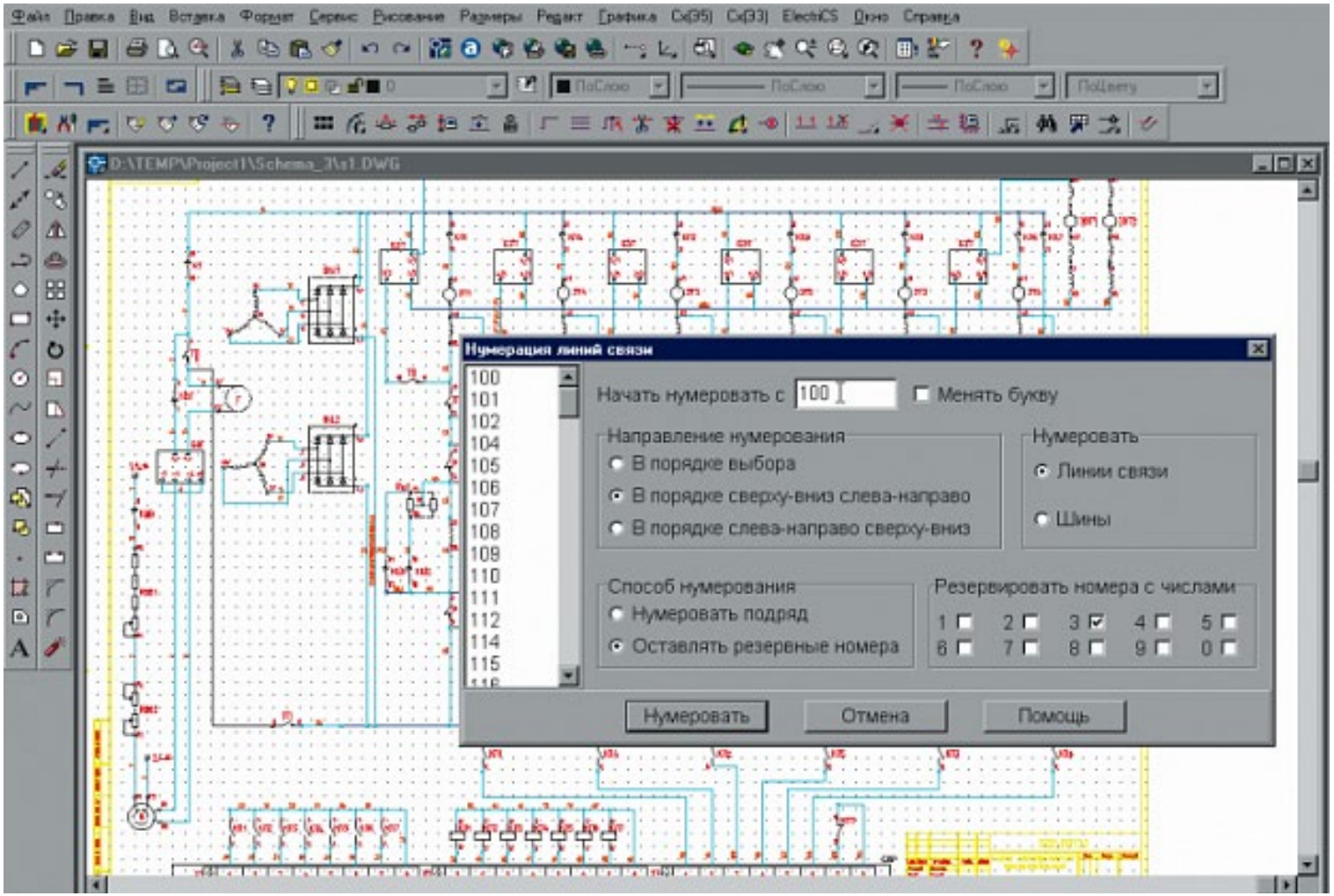

Рисунок А.2 – Нумерація ліній зв’язків електротехнічного обладнання у програмному комплексі ElectriCS, графічна частина якого інтегрована у середовище AutoCAD

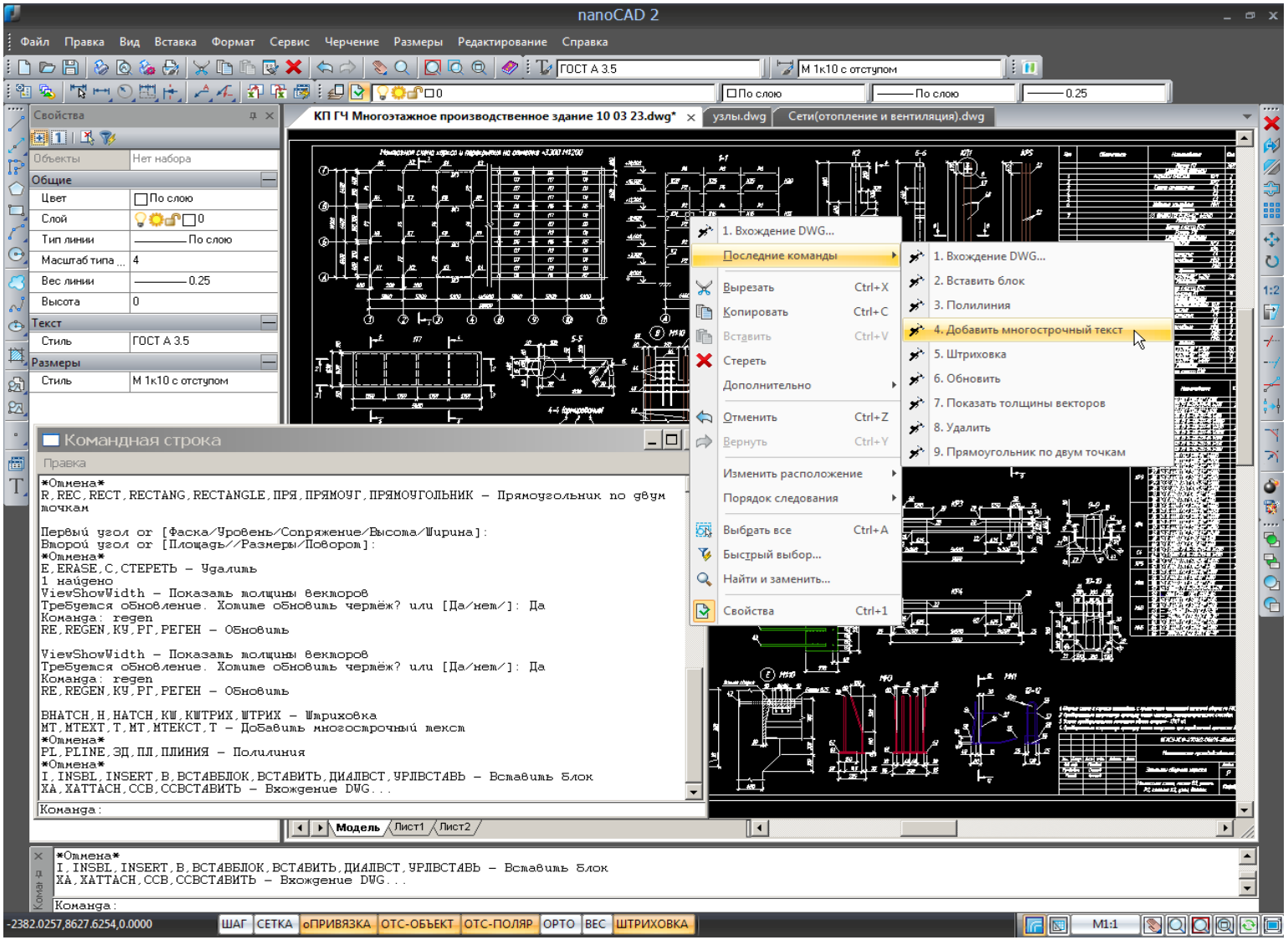

Рисунок А.3 – Графічний інтерфейс користувача базової САПР NanoCAD

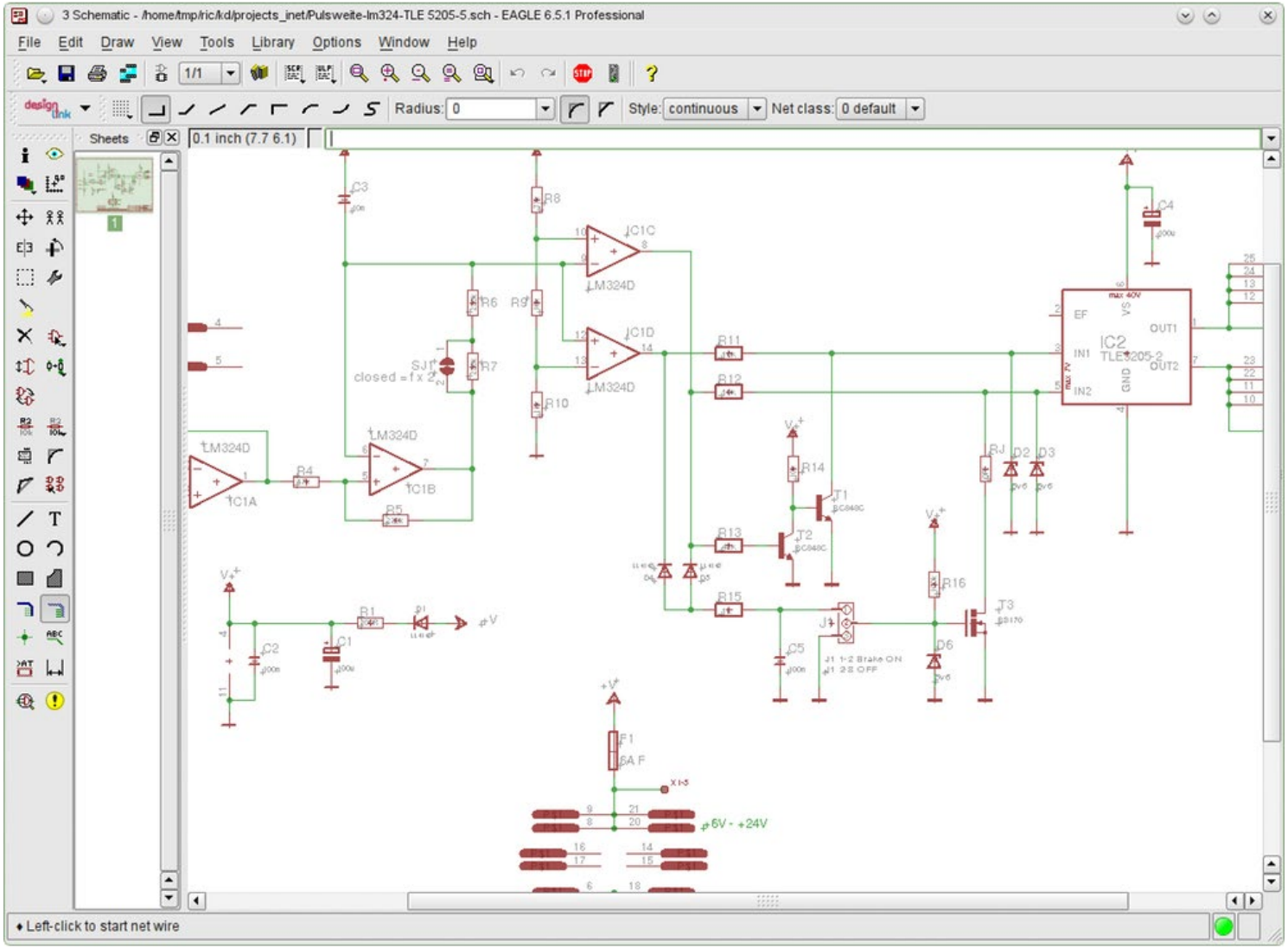

Рисунок А.4 – Графічний редактор схем Schematic Editor, що вбудовано до САПР Cadsoft EAGLE

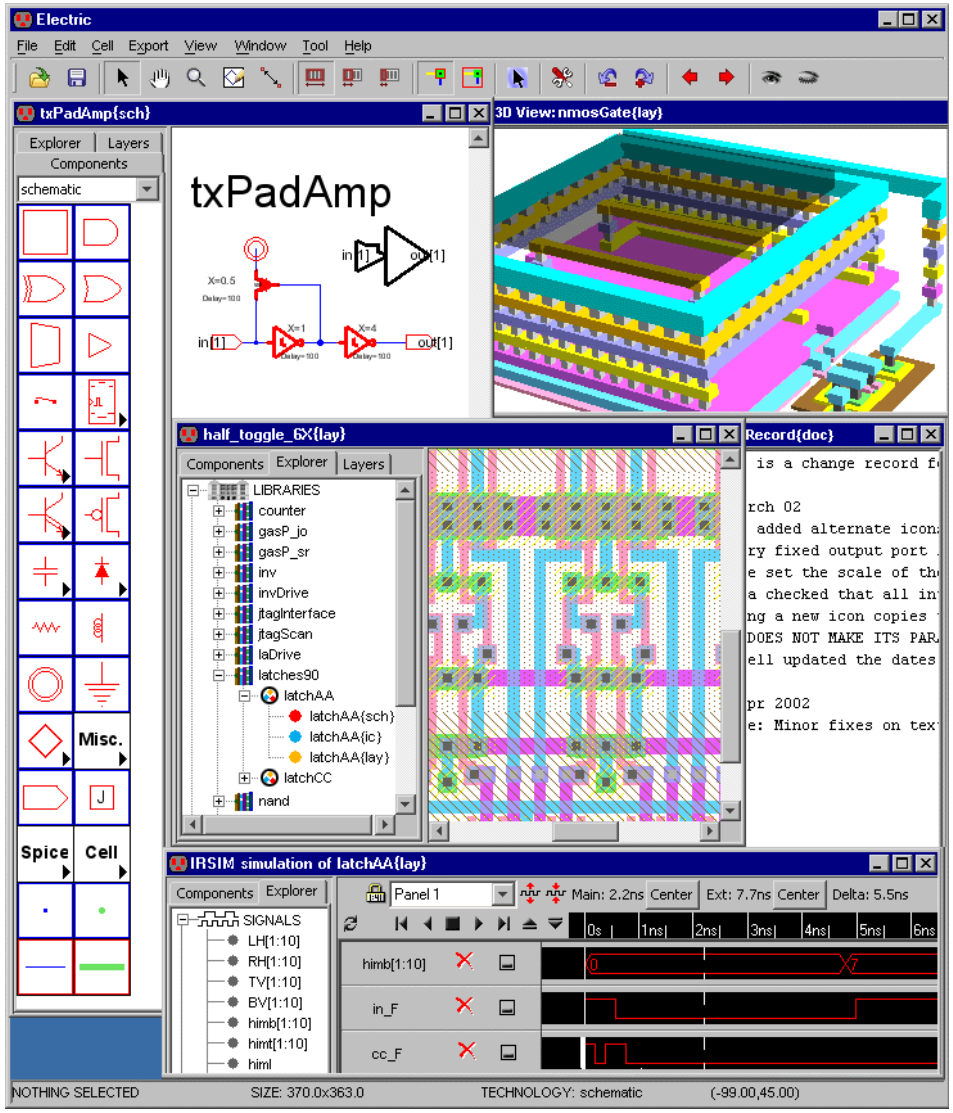

Рисунок А.5 – Архітектурне проектування інтегральних схем у Electric VLSI

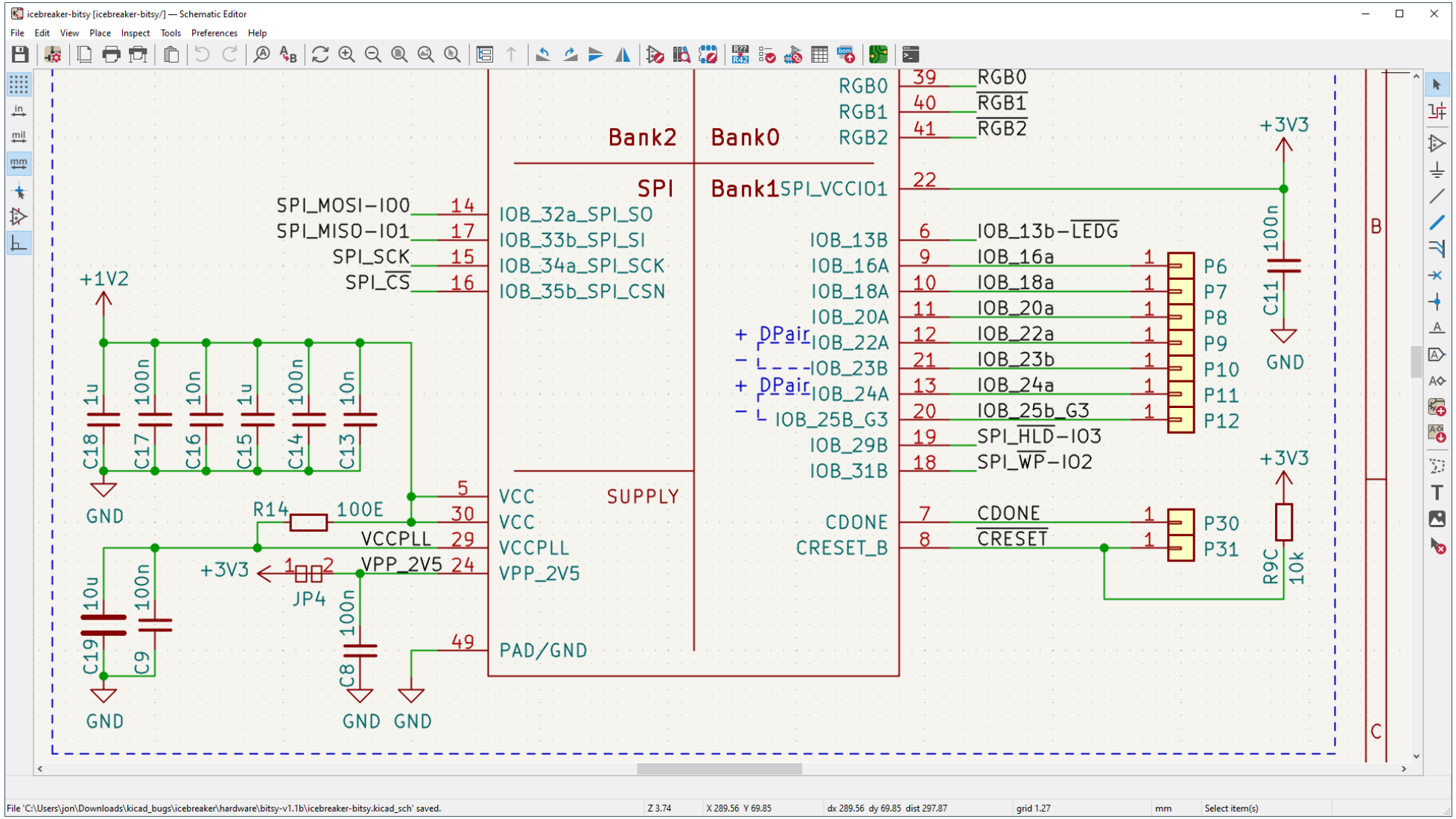

Рисунок А.6 – Редактор електричних схем САПР KiCad