МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____Скарга-Бандурова І.С.
«____»_____ 2018 р.

ДИПЛОМНИЙ ПРОЕКТ (РОБОТА) БАКАЛАВРА

ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

Інструментальні засоби формування комплектів та інсталяцій комплектів компонентів ПТК

Освітньо-кваліфікаційний рівень "бакалавр"
Напрям 6.050102– "Комп'ютерна інженерія"

Керівник проекту:	_____ (підпис)	доц. Ларгін В. А. (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	ст.викл.Критська Я. О. (ініціали, прізвище)
Здобувач вищої освіти:	_____ (підпис)	Фурса П. С. (ініціали, прізвище)
Група:		КІ-14ад

Сєвєродонецьк 2018

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень бакалавр
Напрям підготовки 6.050102 Комп'ютерна інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:

Завідувач кафедри _____
_____ І.С. Скарга-Бандурова
«_____» _____2018 р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) БАКАЛАВРА

Фурса Поліна Сергіївна
(прізвище, ім'я, по батькові)

1. Тема роботи Інструментальні засоби формування комплектів та інсталяцій комплектів компонентів ПТК

керівник проекту (роботи) Ларгін В. А., к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "14 " 05 2018р. №
2. Термін подання студентом роботи
3. Вихідні дані до роботи матеріали переддипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Огляд об'єкту дослідження. Розробка алгоритму інструментальних засобів. Написання комплексу програм. Тестування роботи кожної програми. Охорона праці. Висновки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	ст.викл. кафедри КНІ Критська Я.О.		

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Огляд літератури з теми ДП і постановка задачі	14.05.18-19.05.18	
2	Дослідження матеріалів	20.05.18-25.05.18	
3	Розробка програмної системи	26.05.18-02.06.18	
4	Тестування програмної системи	03.06.18-06.06.18	
5	Розробка розділу охорона праці	07.06.18-09.06.18	
6	Оформлення електронних плакатів	10.06.18-12.06.18	
7	Оформлення пояснювальної записки	13.06.18-15.06.18	

Здобувач вищої освіти _____ Фурса П. С.
(підпис) (прізвище та ініціали)

Керівник _____ Ларгін В. А.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

На тему: «Інструментальні засоби формування комплектів, інсталяції комплектів компонентів програмно-технічного комплексу».

Предметом дослідницької роботи є розробка програмних засобів формування комплектів програмних компонентів ПТК.

У вступі обирається об’єкт дослідження, його теоретична та практична значимість.

У першому розділі містяться загальні відомості про досліджуваний об’єкт, також розкривається актуальність завдання, виконується аналіз останніх розробок, ставиться проблема, мета й задачі дослідження, виконується аналіз апаратних та програмних засобів, використовуваних при вирішенні поставленного завдання.

У другому розділі представлена й детально описана логічна структура інструментальних засобів технологічного програмування.

У третьому розділі описані запропоновані програмні рішення поставленої задачі та представлена алгоритмічна реалізація інструментальних засобів програмування.

Висновки присвячені узагальненню данної роботи та пропозиціям щодо включення цього проекту до виробничого процесу для більшої продуктивності.

Перелік ключових скорочень: ПТК, МСКУ, КС, ІЗП, С, С++.

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ПТК - програмно – технічний комплекс

МСКУ - мікропроцесорні субкомплекси контролю й управління

КС - керуюча система

СДА - сервер діагностування та архівації

ТОУ - технологічний об'єкт управління

АТК - автоматизований технологічний комплекс

ТП - технологічний процес

ВМ - виконавчі механізми

КМП - контролери мікропроцесорні

ІРП - інтерфейс радіальний

МЗВ - модулі зв’язку

МЗО - модулі зв'язку з об'єктом

МКО - модулі контролю обладнання

ПКР - панелі кросові

ПСП - панелі сполучні

НР - нижній рівень

КП ОД - керуюча програма обробки даних

КНР - комплекс нижнього рівняі

ІЗП - інструментальні засоби підготовки

ЦА - центральний архів

ЗМІСТ

ВСТУП ... **8**
1 АНАЛІЗ ЗАДАЧІ Й ПОСТАНОВКА ТЕХНІЧНОГО ЗАВДАННЯ ... **10**
1.1 Загальні відомості ... **10**
1.1.1 Особливості та структура мікропроцесорного субкомплексу контролю і управління (МСКУ-4) ... **10**
1.1.2 Керуюча система (КС) МСКУ ... **12**
1.2 Актуальність розробки ... **14**
1.3 Аналіз останніх розробок ... **14**
1.4 Основні завдання даного дипломного проекту ... **14**
1.5 Вимоги до складу технічних і програмних засобів ... **15**
1.6 Вимоги до засобів розробки ... **16**
Висновки до розділу 1 ... **16**
2 ОПИС ЛОГІЧНОЇ СТРУКТУРИ ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ ... **17**
2.1 Склад інструментальних програм ... **17**
2.1.1 Монітор інструментальної системи підготовки проектів ... **18**
2.1.2 Модулі діалогу групи підменю «Робота з проектами» ... **19**
2.1.3 Модулі діалогу групи підменю «Робота з проектами» -> «Група» ... **19**
2.1.4 Модулі діалогу групи підменю «Робота з проектами» -> «Проект» ... **20**
2.1.5 Архівування ... **21**
2.1.6 Формування завдання на складання комплектів ПЗ ... **21**
2.1.7 Загальні відомості про технологію підготовки КС МСКУ, КП ОД і налаштувань ОБД СДА ... **25**
2.1.8 Модулі діалогу записи у FLASH ПП ... **29**
2.2 Налаштування ОБД нижнього рівня СДА ... **29**
2.3 Структура результуючих каталогів ... **30**
2.4 Структура каталогів групи проектів з робочою інформацією ... **30**
2.5 Файл-заготовка *.xdd для формування дескрипторів ОБД для СДА ... **31**
2.5.1 Файл-дескриптор складу блоків оперативних даних *.xdb ... **33**
2.5.2 Файл-дескриптор конфігурації КНР, що входять в групу проектів *.xcf ... **34**
2.5.3 Файл налаштувань мережевих засобів, що функціонують в СДА ConfEthL*xxx*.xml ... **39**
2.4.4 Файл-заготівка для формування дескрипторів data_attrs ... **44**
2.4.5 Файл-протокол результатів складання проектів ... **46**
Висновки до розділу 2 ... **46**
3 ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ФОРМУВАННЯ КОМПЛЕКТІВ, ІНСТАЛЯЦІЇ КОМПЛЕКТІВ ПЗ КОМПОНЕНТІВ ПТК ... **47**
3.1 Загальні відомості ... **47**
3.2 Інструментальні засоби формування комплектів, інсталяції комплектів компонентів ПТК ... **48**
3.2.1 Програма fpo ... **48**
3.2.1.1 Опис підпрограми main() ... **49**
3.2.1.2 Опис підпрограми init() ... **50**
3.2.1.3 Опис підпрограми GrpDirSet() ... **51**
3.2.1.4 Опис підпрограми NewComplect_clicked() ... **52**
3.2.1.5 Опис підпрограми Params_clicked() ... **53**
3.2.1.6 Опис підпрограми Save_clicked() ... **54**
3.2.1.7 Опис підпрограми Prepare_clicked() ... **55**
3.2.1.8 Опис підпрограми ins_file() ... **56**
3.2.1.9 Опис підпрограми pushButtonOk_clicked() ... **57**

3.2.1.10 Опис підпрограми pushButtonCancel_clicked() ... 57
3.2.2 Програма finalization ... 58
3.2.2.1 Загальні відомості ... 58
3.2.2.2 Опис підпрограми main() ... 58
3.2.3 Програма PackCreator ... 59
3.2.3.1 Загальні відомості ... 59
3.2.3.2 Опис підпрограми main() ... 60
3.2.3.3 Опис підпрограми TC_SupplyCreator() ... 61
3.2.3.4 Опис підпрограми TC_WorkWithPrj() ... 62
3.2.3.5 Опис підпрограми TC_ArchiveCreator() ... 63
3.2.4 Програма PackInstaller ... 66
3.2.4.1 Загальні відомості ... 66
3.2.4.2 Опис підпрограми run() ... 67
3.2.4.3 Опис підпрограми runinstallmode() ... 68
3.2.4.4 Опис підпрограми runverifymode() ... 68
3.2.4.5 Опис підпрограми begininstallprocess() ... 69
3.2.4.6 Опис підпрограми OnCheckPackNum() ... 71
3.2.4.7 Опис підпрограми OnCheckPackNums() ... 71
3.2.4.8 Опис підпрограми oninstall() ... 72
3.2.5 Програма PackComporator ... 73
3.2.5.1 Загальні відомості ... 73
3.2.5.2 Опис підпрограми main() ... 73
3.2.5.3 Опис підпрограми TC_ComparePackages() ... 74
Висновки до розділу 3 ... 75
4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯЇХ ... 76
4.1 Аналіз потенційно небезпечних і шкідливих виробничих факторів, що впливають на персонал ... 76
4.2 Заходи з техніки безпеки ... 77
4.3 Заходи, що забезпечують виробничу санітарію та гігієну праці ... 78
4.4 Рекомендації із пожежної профілактики ... 81
Висновки до розділу 4 ... 83
ВИСНОВКИ ... 84
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ ... 85
ДОДАТОК А ... 87
ПРОГРАМНА РЕАЛІЗАЦІЯ ... 87
Додаток Б ... 97
ПРЕЗЕНТАЦІЯ ... 97

ВСТУП

ПрАТ «СНВО «Імпульс» є провідним виробником програмно-технічних комплексів (ПТК) для АСК ТП. Для організації нижнього рівня ПТК призначені мікропроцесорні субкомплекси контролю й управління (МСКУ), які здійснюють прийом і необхідну обробку вимірювальної інформації від об'єкта, управління технологічним обладнанням, регулювання параметрів технологічного процесу, захист обладнання та блокування [1].

МСКУ використовується для збору, обробки інформації та управління технологічними установками як автономно, так і в складі ПТК.

МСКУ застосовуються в якості:

- підсистем нижнього рівня АСК ТП;
- інтелектуальних автономних систем контролю і управління;
- промислових контролерів відмовостійких систем автоматизації об'єктів атомної енергетики.

МСКУ має всі функції і засоби, необхідні для створення сучасних систем управління технологічними процесами: реєстрацію і обробку параметрів процесу, регулювання, управління, захисту і блокування, сигналізації, обчислювальні операції, оптимізацію, експертні системи, візуалізацію процесу на екранах моніторів, дистанційне керування. Роботу МСКУ відповідно до визначених користувачем функціями організовує керуюча система (КС) МСКУ. КС МСКУ називається сукупність системних програм, налаштувань даних програм (на конфігурацію МСКУ і необхідні функції) і прикладних програм користувача. КС МСКУ створюється під кожне конкретне застосування МСКУ розробником відповідного ПТК. У КС МСКУ обмін з компонентами ПТК може здійснюватися через мережеві контролери Ethernet по протоколу мережі Ethernet нижнього рівня. Для всіх комплексів програмного забезпечення необхідно забезпечувати надійність зберігання і поставки Замовнику, а також зручність у використанні. Для вирішення цих завдань і призначені сервісні програмні засоби, що

забезпечують автоматизацію виконання робіт при підготовці, зберіганні та супроводі комплексів програмного забезпечення. Ця технологія забезпечує виконання повного циклу робіт, пов'язаних з підготовкою комплексів програм для передачі в архів, зберіганням, обліком і контролем схоронності комплектів цих комплексів в архіві, а також їх тиражуванням, внесенням змін і введенням в експлуатацію. Крім того, ця технологія забезпечує надійний облік змін, що вносяться до програмні модулі і експлуатаційну документацію комплексів програм, надійне зберігання комплектів даного комплексу спільно з документацією супроводу в архіві, надійне виготовлення копій комплектів, а також надає надійний механізм перевірки відповідності копій комплекту оригіналу комплекту [2].

Для виконання робіт даного виду, як правило, використовуються інструментальні пакети програм, призначені для створення інсталяційних носіїв програмних комплексів користувача.

Метою індивідуального завдання є розробка інструментальних засобів формування комплектів, інсталяції комплектів програмних компонентів ПТК.

1 АНАЛІЗ ЗАДАЧІ Й ПОСТАНОВКА ТЕХНІЧНОГО ЗАВДАННЯ

1.1 Загальні відомості

1.1.1 Особливості та структура мікропроцесорного субкомплексу контролю і управління (МСКУ-4)

Мікропроцесорні керуючі обчислювальні комплекси МСКУ-4 - сімейство проектно-компонованих, гнучко програмованих логічних промислових контролерів, призначених для застосування в якості:

-промислових контролерів відмово-стійких систем автоматизації особливо відповідальних об'єктів (атомної і теплової енергетики, залізничного транспорту, нафтогазового комплексу та ін.);

- підсистем нижнього рівня АСК ТП;

- інтелектуальних автономних систем контролю і управління.

Функції:

- збір, перетворення, первинна обробка та зберігання інформації, отриманої від об'єкта;

- формування сигналів і видача керуючих впливів на виконавчі механізми (ВМ);

- реалізація алгоритмів контролю та управління, різних законів регулювання, захистів, блокувань, пуску й зупинки устаткування;

- взаємозв'язок із зовнішніми абонентами ПЗ інтерфейсів;

- Ethernet.

Склад конкретного МСКУ-4 визначається особливостями його застосування в системі автоматизації об'єкта.

Складові частини МСКУ-4:

- контролери мікропроцесорні (КМП) на основі інтерфейсу PCI-Express. Кожен КМП оснащений двома портами оптичного інтерфейсу Ethernet, двома портами Ethernet з мідними провідниками і 13 портами внутрішнього радіального

інтерфейсу ІРП-4. Кожен КМП може мати 4, 8 або 12 додаткових портів оптичного Ethernet при установці одного, двох або трьох модулів зв'язку МЗВ-41 відповідно;

- модулі зв'язку з об'єктом (МЗО), призначені для введення/виведення дискретних і аналогових сигналів, оснащені трьома портами ІРП-4;

- модулі зв'язку МЗВ-42, призначені для зв'язку між оптичним Ethernet і 49 портами ІРП-4;

- модулі МЗВ-43, призначені для зовнішніх зв'язків по інтерфейсах RS-422 і RS-485;

- модулі контролю обладнання МКО-4, призначені для контролю працездатності й стану обладнання, розміщеного в шафі МСКУ-4;

- панелі кросові (ПКР), призначені для підключення кабелів від об'єктів;

- панелі сполучні (ПСП), призначені для зв'язку ПКР з МЗО;

- каркаси монтажні (КМ), призначені для установки КМП і МЗО.

Особливості МСКУ-4:

- центральна частина МСКУ-4 може бути нерезервованої (один КМП) або резервованої (три КМП);

- відповідають жорстким вимогам промислових стандартів, мають високу ступінь електромагнітної сумісності, високу стійкість до ударних і вібраційних навантажень;

- відсутність примусової вентиляції у конструкціях;

- самодіагностування з локалізацією несправностей до змінного блоку;

- можливість заміни функціонального блоку, не вимикаючи ("гаряча" заміна);

- електроживлення здійснюється від двох незалежних фідерів напругою 220В як змінного, так і постійного струму;

- середній термін служби - не менше 30 років;

- МСКУ-4 є засобами вимірювань.

ПЗ МСКУ представлене на рис. 1.1 [3].

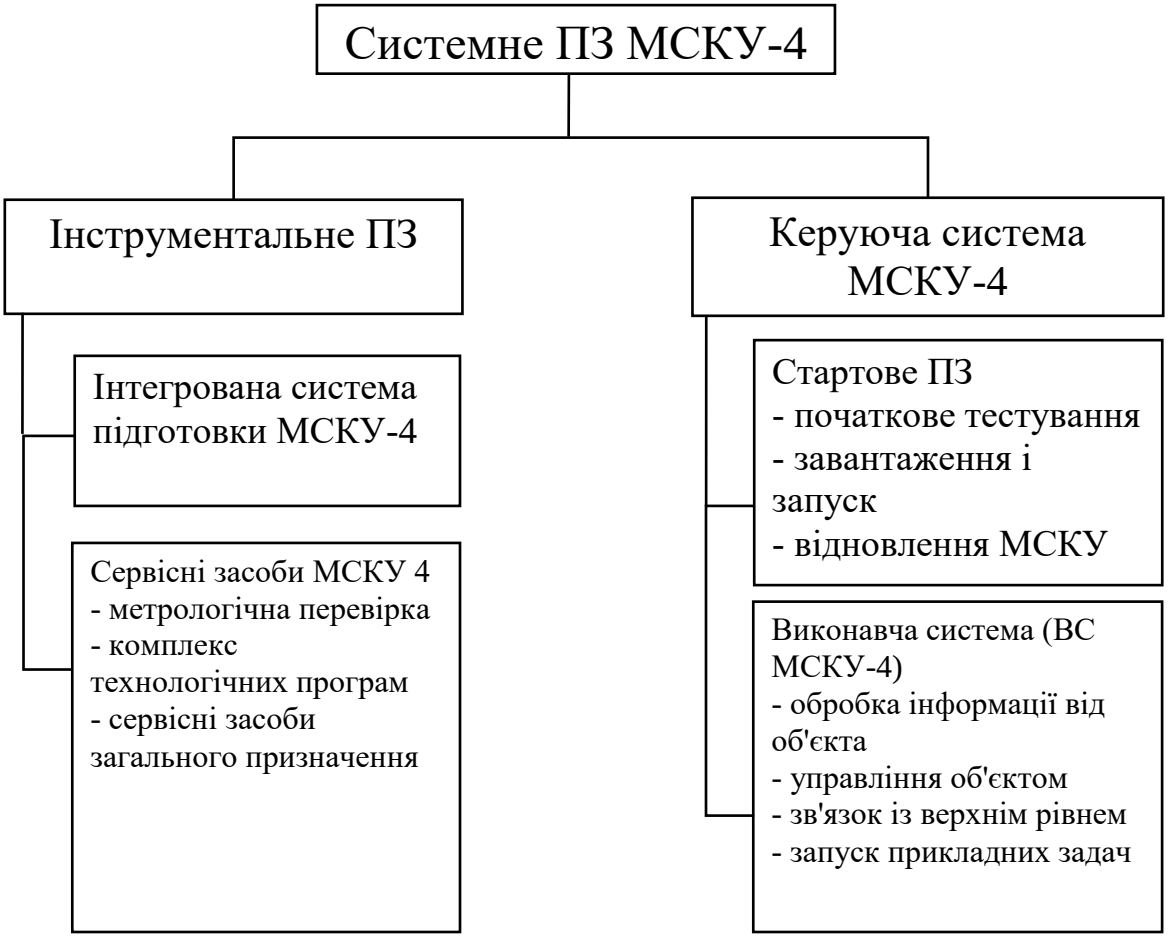


Рисунок 1.1 - ПЗ МСКУ-4

МСКУ-4 компонуються в підлогових шафах зі ступенем захисту IP21. Шафа, що має в своєму складі КМП, називається ведучим, який не має КМП - веденим (шафа розширення).

1.1.2 Керуюча система (КС) МСКУ

КС МСКУ представляють собою сукупність системних програм, налаштувань даних програм (на конфігурацію МСКУ і необхідні функції) і прикладних програм користувача, які організовують роботу МСКУ відповідно до визначених користувачем функціями.

Цільова КС МСКУ створюється під кожне конкретне застосування МСКУ розробником відповідної системи управління, до складу якої входить МСКУ (або КМП).

Цільова КС МСКУ складається з модуля, що включає наступні завантажувальні модулі:

- резидентное ядро КС МСКУ (з реальною або вірутальних адресацією);
- настройки прикладних програм;
- виконуваний код первинної обробки вхідних сигналів;
- виконуваний код формування вихідних сигналів;
- виконуваний код алгоритмів прикладних програм;
- опис оперативно змінюваних даних;
- опис складу завантажувальних модулів.

У КС МСКУ обмін з компонентами ПТК може здійснюватися через мережеві контролери Ethernet по протоколу мережі Ethernet нижнього рівня (далі - мережа Ethernet НР).

Резидентне ядро КС МСКУ включає всі програмні модулі для виконання необхідних функцій для всіх виконань і варіантів застосувань КМП і записується у FLASH ПП КМП.

Обробка даних в робочій станції можлива за допомогою прикладного процесу, організованого за принципом КС МСКУ. Прикладний процес являє собою програму обробки керуючої програми обробки даних (КП ОД), що функціонує в робочій станції у середовищі ОС Linux і забезпечує обмін блоками даних з будь-якими абонентами мереж і іншими процесами (наприклад, ОБД СДА), виконання прикладних програм обробки даних, написаних користувачем на мові технологічного програмування.

КП ОД забезпечує виконання таких функцій:

- виконання прикладних програм, розроблених на мові технологічного програмування, для обробки даних, прийнятих від інших абонентів мережі (мереж) Ethernet НР;

- підготовку і передачу даних іншим абонентам мережі (мереж) Ethernet НР.

КП ОД є віртуальний нерезервованої КНР.

Завантажувальний модуль КП ОД готується інструментальними засобами комплексу програм на підставі файлів вихідних текстів налаштувань і вихідних текстів прикладних програм. Дані модулі встановлюються на жорсткий диск робочої станції.

1.2 Актуальність розробки

Для всіх комплексів програмного забезпечення необхідно забезпечувати надійність зберігання і поставки, а також зручність у використанні. Для вирішення цих завдань призначені інструментальні засоби, що забезпечують автоматизацію виконання робіт при підготовці, зберіганні та супроводі комплексів програмного забезпечення.

1.3 Аналіз останніх розробок

В даний час в ПрАТ «СНВО «Імпульс» розроблений і поставляється Замовникам комплекс інструментальних програм для підготовки цільових проектів керуючої системи МСКУ-4. Для розробки прикладних програм, що функціонують у МСКУ, необхідно надати розробнику програмні засоби формування комплектів програмних компонентів ПТК.

Тому виникла необхідність розробки комплексу інструментальних програм, що функціонують в найбільш поширеному операційному середовищі Windows, що забезпечують формування компонентів ПТК, розроблених на мові технологічного програмування.

1.4 Основні завдання даного дипломного проекту

1. дослідити об’єкт, розкрити актуальність завдання, проаналізувати останні розробки, сформувати мету й задачі дослідження, зробити аналіз

апаратних та програмних засобів, використаних при вирішенні поставленної задачі;

2. розглянути, дослідити та детально описати логічну структуру інструментальних засобів технологічного програмування;

3. представити алгоритм та комплекс програм, що забезпечують формування компонентів ПТК, розроблених на мові технологічного програмування, в ПЕОМ у середовищі Windows.

1.5 Вимоги до складу технічних і програмних засобів

Інструментальні засоби функціонують на інструментальній IBM PC сумісної ПЕОМ під управлінням операційної системи Windows 2000, Linux. В ОС Linux або ОС Implix на робочої станції повинні бути встановлені стандартні засоби обробки make-файлів.

Для запису завантажувального модуля КС МСКУ необхідне підключення інструментальної ПЕОМ до МСКУ по мережі Ethernet HP.

До складу інструментальної ПЕОМ повинні входити:

- процесор архітектури x86 з частотою 3 GHz або вище;
- оперативна пам'ять не менше 1 Gbyte;
- жорсткий диск з об'ємом не менше 60 Gbyte;
- кольоровий монітор стандарту VGA або SVGA з дозволом екрану не менше 1024 на 768 точок (при налаштуванні монітора в ОС Windows рекомендується використовувати встановлюється за умовчанням невеликий розмір шрифту дисплея (96 точок на дюйм) для коректного відображення написів і повідомлень, що виводяться в процесі роботи програм комплексу);
- пристрій введення з CD-ROM, DVD-ROM;
- клавіатура алфавітно-цифрова (далі - клавіатура);
- контролер (контролери) Ethernet.

Всі програмні модулі інструментальних засобів написані на мові С і С++. Процес створення виконуваних модулів (трансляція і компонування) інструментальних засобів автоматизований.

Програми працюють на IBM PC-сумісної ПЕОМ під управлінням ОС Windows, ОС Linux або ОС Implix.

1.6 Вимоги до засобів розробки

В якості інструментальних засобів для розробки підсистеми повинні використовуватися широко застосовувані мови програмування С, С ++.

Висновки до розділу 1

У першому розділі виконані:

- дослідження об’єкту;
- розкриття актуальності завдання;
- аналіз останніх розробок, на основі яких сформувані мета й основні задачі дослідження;
- аналіз апаратних та програмних засобів, використаних при вирішенні поставленної задачі.

2 ОПИС ЛОГІЧНОЇ СТРУКТУРИ ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ

2.1 Склад інструментальних програм

Інструментальні засоби мають структуру, показану на рис. 2.1.

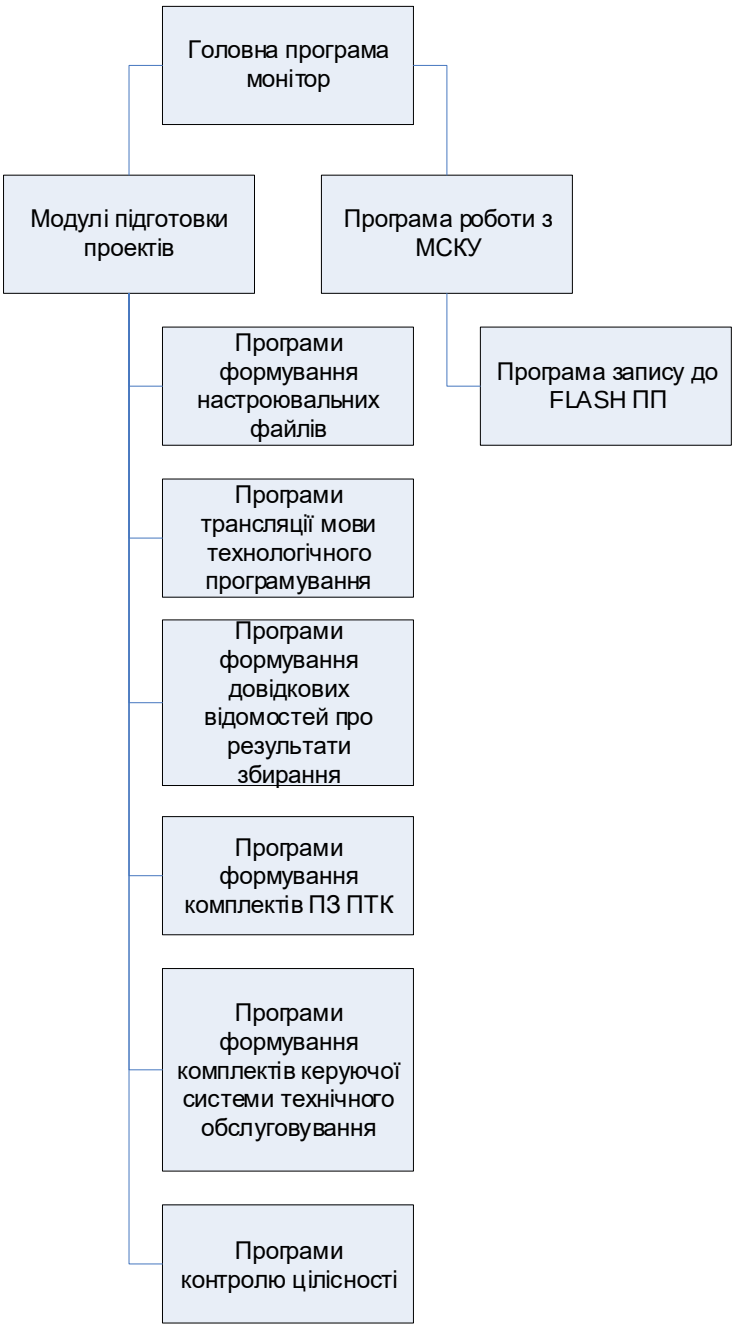


Рисунок 2. 1 - Загальна структура інструментальних засобів

Інструментальні засоби включають такі програмні компоненти:

- головна програма (програма-монітор) виконує:

- виклик програмних модулів в залежності від виконуваних дій;

- виконання контролю цілісності програм;

- до програм діалогу підготовки налаштувань (прикладних програм, СДА) відносяться:

- модулі діалогу групи підменю «робота з проектами»;

- модулі діалогу групи підменю «група проектів»;

- модулі діалогу групи підменю «проект»;

- програма формування виконуваного модуля КС МСКУ, КП ОД;

- до програм діалогу записи у FLASH ПП:

- модулі діалогу записи резидентного ядра під FLASH ПП;

- модулі діалогу запису налаштувань прикладних програм у FLASH ПП.

2.1.1 Монітор інструментальної системи підготовки проектів

Монітор призначений для забезпечення роботи комплекту програмних модулів, що викликаються в міру необхідності виконання обраної операції.

Склад модулів головної програми:

- головна підпрограма інструментальних засобів підготовки;

- підпрограма ініціалізації початкових значень;

- підпрограма виклику діалогу контролю цілісності;

- підпрограма виклику діалогу підготовки налаштувань групи проектів;

- підпрограма виклику довідкової інформації;

- підпрограма виклику діалогу підготовки комплекту програм для керуючої системи технічного обслуговування;

- підпрограма виклику діалогу запису резидентного ядра.

2.1.2 Модулі діалогу групи підменю «Робота з проектами»

Модулі діалогу групи підменю «Робота з проектами» призначені для виклику модулів створення групи проектів або одного проекту з групи, перепідготовки групи проектів або одного проекту і запису проекту під FLASH ПП КМП.

Програма представляє собою набір взаємозв'язаних модулів:

- підпрограма ініціалізації форми пункту меню «робота з проектами»;
- підпрограма фільтрації повідомлень;
- підпрограма включення файлу генерації в проект;
- підпрограма включення файлу бібліотеки в проект;
- підпрограма включення файлу кросування в проект;
- підпрограма виходу з форми пункту меню «Робота з проектами»;
- підпрограма виводу версії генерації проектів;
- підпрограма відображення проектів на формі;
- підпрограма видалення структури попередньої групи;
- підпрограма «підсвічування» змінених файлів у всіх проектах групи;
-підпрограма «підсвічування» файлів всіх проектів червоним кольором, якщо вони були змінені;
-підпрограма «підсвічування» файлів проектів групи червоним кольором, якщо вони були змінені;
- підпрограма видалення директорії проекту;
- підпрограма збереження структури каталогу групи на диску ПЕОМ.

2.1.3 Модулі діалогу групи підменю «Робота з проектами» -> «Група»

Модулі діалогу групи підменю «Робота з проектами» -> «Група» призначені для виклику модулів створення, збереження, відкриття групи проектів, архівування та підготовки групи проектів.

Програма представляє собою набір взаємозв'язаних модулів:

- підпрограма створення групи проектів;
- підпрограма вибору групи проектів;
- підпрограма відображення проектів на формі;
- підпрограма запису проекту у FLASH ПП;
- підпрограма підготовки групи проектів;
- підпрограма збереження групи проектів;
- підпрограма архівування групи проектів;
- підпрограма запуску процесу архівування;
- підпрограма вибору директорії;
- підпрограма вибір директорії для створення архіву;
- підпрограма вибору директорії для групи проектів;
- підпрограма вибору файлу кросування.

2.1.4 Модулі діалогу групи підменю «Робота з проектами» -> «Проект»

Модулі діалогу групи підменю «Робота з проектами» -> «Проект» призначені для виклику модулів створення, збереження, відкриття проектів та підготовки проектів.

Програма представляє собою набір взаємозв'язаних модулів:

- підпрограма створення проекту в групі;
- підпрограма запису проекту у FLASH ПП;
- підпрограма видалення проекту;
- підпрограма видалення директорії проекту;
- підпрограма підготовки поточного проекту;
- підпрограма збереження проекту.

2.1.5 Архівування

Після завершення підготовки групи проектів надається можливість формування архіву для перенесення на інші робочі станції або ПЕОМ.

2.1.6 Формування завдання на складання комплектів ПЗ

Сервісні програмні засоби, що забезпечують функціонування ІЗП в режимі «Формування завдання на складання комплектів ПЗ», призначені для організації технології розробки, обліку, зберігання, постачання та введення в експлуатацію ПЗ компонентів ПТК.

Ця технологія забезпечує виконання повного циклу робіт, пов'язаних із підготовкою комплексів програм для передачі в архів, зберіганням, обліком і контролем збереження комплектів цих комплексів в архіві, а також їх тиражуванням, внесенням змін і введенням в експлуатацію.

Крім того, ця технологія забезпечує надійний облік змін, що вносяться до програмних модулів і експлуатаційну документацію комплексів програм, надійне зберігання комплектів даного комплексу спільно з документацією супроводу в архіві, надійне виготовлення копій комплектів, а також надає надійний механізм перевірки відповідності копій комплекту оригіналу комплекту.

В таблиці 2.1 подані терміни, використовувані при описі технології формування комплектів ПЗ, та їх визначення.

Таблиця 2.1 - Терміни, що використовуються при описі технології формування комплектів ПЗ, та їх визначення.

Термін	Визначення
Комплекс програм	Програмний виріб, що застосовується самостійно й, за необхідністю, спільно з іншими комплексами програм
Компонент комплексу програм	Файл даних будь-якого формата
Інстальований комплекс програм	Комплекс програм, встановлений на жорсткі диски ПЕОМ, у форматі, що забезпечує його використання відповідно до функціонального призначення

Продовження табл. 2.1

Термін	Визначення
Комплект постачання комплексу програм	Комплекс програм на носіях даних (або жорсткому диску ПЕОМ) в запакованому форматі, призначений для постачання користувачу й підготовлений сервісними програмами в режимі «Робота з проектами»> «Група»->«Формування завдання на складання комплектів ПЗ»
Комплект архіву	Комплекс програм на носіях даних (або жорсткому диску ПЕОМ) в запакованому форматі з вихідними текстами розроблених модулів ПО і документації супроводу (опис програм, опис інтерфейсів між програмами і т. д.) В обсязі, достатньому для можливих коригувань програм
Комплект ПЗ комплексу програм	Комплект постачання і комплект архіву комплексу програм
Інсталяція комплекту	Процес перетворення комплекту ПЗ комплексу програм з упакованого формату в формат, що забезпечує його використання відповідно до функціонального призначення, що виконується сервісними програмами, включеними до складу комплекту ПЗ у процесі його створення. В процесі інсталяції виконується контроль цілісності комплекту ПЗ і формування контрольних відомостей для періодичного контролю цілісності інстальованого комплексу програм (комплексів програм)

Для передачі в центральний архів (далі - ЦА) кожному комплексу програм повинно бути присвоєно позначення і найменування.

Позначення комплекту постачання має вигляд:

код_предприятия.номер_комплекса-номер_версии_комплекса, де
код_підприємства - десяткове число (наприклад, 0229767);
номер_комплекса - десяткове число в межах від 1 до 99999;
номер_версії_комплекса - десяткове число в межах від 1 до 999

Комплекту архіву присвоюється таке ж позначення, але з додаванням символу «П».

В окремих випадках позначення комплектів ПЗ для групи компонентів може визначатися за такими правилами:

код_предприятия.xxyyy-zz, де
xx - номер комплекту ПЗ в групі (старша частина). Приймає значення в межах від 0 до 99;
yyy - загальний номер комплекту ПЗ в групі (молодша частина). Приймає значення в межах від 1 до 999;
zz - версія комплексу програм. Приймає значення в межах від 1 до 99

Найменування комплекту постачання повинно мати вигляд:

Наименование предприятия
Наименование комплекта

Найменування комплекту архіву повинно мати вигляд:

Наименование предприятия
Наименование комплекта

Архів компонент постачання і компонент супроводу, де

Наименование предприятия - найменування підприємства-розробника ПЗ;
Наименование комплекта - назва комплексу програм, вибрана розробником і визначальна загальні функціональні властивості ПЗ

Облік позначень і найменувань комплексів програм виконується службами супроводу і зберігання ПЗ.

Комплекс програм, призначений для постачання замовнику, повинен передаватися в ЦА у вигляді двох взаємопов'язаних комплектів ПЗ:

- комплекту поставки і документації згідно специфікації у формі і обсязі, достатньому для освоєння і експлуатації у користувачів;

- комплекту архіву з вихідними текстами розроблених модулів ПЗ і документації супроводу (опис програм, опис інтерфейсів між програмами і т. д.) в обсязі, достатньому для можливих коригувань програм, тобто, супроводу.

ІЗП надають користувачеві можливість автоматизованого формування комплектів ПЗ компонентів ПТК при обробці групи проектів.

Сервісні програми ІЗП включають програмні модулі, що забезпечують виконання таких видів робіт для організації зберігання, постачання, введення в експлуатацію та супроводу комплексів програм:

- підготовку комплекту архіву на носіях в спеціальному запакованому форматі зі збереженням контрольних відомостей про комплекс програм, версії комплексу (наприклад, для передачі в архів);

- отримання комплекту поставки з комплекту архіву;

- періодичний контроль цілісності комплектів ПЗ у процесі зберігання;

- облік змін, що вносяться до програмних модулів і документацію комплексів програм;

- інсталяцію комплектів ПЗ на жорсткі диски з метою введення комплексів програм в експлуатацію з повним контролем у ході інсталяції цілісності комплекту ПЗ і відповідності його еталонному примірнику;

- періодичний контроль цілісності інстальованого комплексу програм у ході його експлуатації;

- можливість виконання на жорсткому диску ПЕОМ всіх службових операцій, пов'язаних із вхідним контролем, обліком, зберіганням, внесенням змін, отриманням копій, що дозволяє спростити і прискорити роботу з комплексом програм в архіві.

Функції сервісних програм ІЗП орієнтовані на користувачів, які здійснюють розробку, супровід і поставку комплектів ПЗ різного функціонального призначення, що функціонують під управлінням ОС Windows/Linux і в КМП.

Формування комплектів ПЗ для групи компонентів ПТК здійснюється наступними діями:

- підготовка вихідних файлів комплектів ПЗ і файлів документації на диску ПЕОМ;

- створення файлу завдання на складання комплектів ПЗ;

- формування комплектів ПЗ.

Із підготовленими комплектами ПЗ можуть бути виконані наступні операції:

- запис на накопичувачі CD-ROM, призначені для передачі в ЦА;

-установка (інсталяція) комплектів ПЗ на жорсткий диск інструментальної ПЕОМ;

-коригування компонентів підготовлених комплексів програм (початкового програмного коду) й формування нових носіїв комплектів ПЗ.

Запис на накопичувачі CD-ROM, призначені для передачі в ЦА, здійснюється штатними засобами.

Установка (інсталяція) комплектів ПЗ поставки на жорсткий диск інструментальної ПЕОМ з контролем цілісності даних здійснюється сервісними засобами, що входять до складу інсталяційного комплекту згідно з інструкцією.

Коригування комплектів ПЗ здійснюється наступними діями:

- інсталяція комплектів архіву всіх компонентів ПТК;
- внесення змін до текстів прикладних програм, коригування документації і т. п.;
- виконання операції підготовки проектів групи, в процесі якої будуть сформовані комплекти ПЗ і сформовані відомості про комплекти ПЗ із зміненими компонентами й список змін по кожному комплекту ПЗ.

2.1.7 Загальні відомості про технологію підготовки КС МСКУ, КП ОД і налаштувань ОБД СДА

Для кожного компонента ПТК вихідні файли, які розробляються користувачем (опису конфігурації, прикладні програми, ...), проміжні й підсумкові файли модулів КС МСКУ й налаштувань СДА, підготовлювані ІЗП групуються у вигляді проекту [4].

ІЗП забезпечують одночасну підготовку результуючих файлів проектів для всіх МСКУ й СДА, що входять до складу ПТК, забезпечуючи тим самим спрощення опису й контроль взаємозв'язків між компонентами ПТК.

Всі вихідні і результуючі файли в процесі підготовки конкретного проекту (проекту КС МСКУ, проекту КП ОД, проекту налаштувань ОБД СДА) формуються в каталозі проекту, яка вказана користувачем при роботі з ІЗП (ім'я каталогу - ім'я проекту).

Проекти, що відносяться до одного ПТК, об'єднуються в одну групу й створюються в загальному каталозі на диску (ім'я групи - це ім'я каталогу), в якому розташовуються підкаталоги проектів (ім'я підкаталогу - це ім'я відповідного проекту). Загальна структура каталогів показана на рис. 2.2.

Проміжні файли в процесі підготовки проектів групи формуються в каталозі *Имя_группы_проектов*/temp.

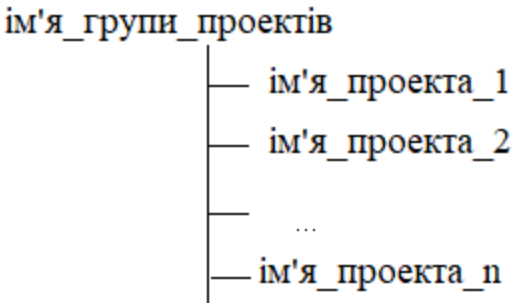


Рисунок 2.2 - Каталог проміжних файлів

Максимальне число проектів в одній групі не повинна перевищувати 100.

Імена груп проектів, проектів і файлів, що включаються до складу проектів, повинні являти собою послідовність літер латинського алфавіту (A - Z, a - z), цифр (0 - 9) і символу підкреслення «_». Послідовність повинна мати довжину не більше восьми символів.

Перед початком роботи з ІЗП у режимі підготовки користувач для кожного проекту повинен підготувати в довільних каталогах на диску необхідні налаштування й прикладні програми для компонентів ПТК (МСКУ, КП ОД, СДА) у вихідних файлах зазначених у таблиці 2.2 типів.

Таблиця 2.2 - Види вихідних файлів для настройки КС МСКУ

Розширення файлу	Вид файлу	Тип проекту	Призначення файлу
***.itg**	Текстовий	Будь-який (У, УО, УТ, УЗ, УЗО, УП, УПО, ОБД/СДА)	Файли з розширенням *.itg включають макрокоманди опису конфігурації МСКУ зі складу ПТК і визначення структури повідомлень обміну даними між компонентами ПТК. Дані відомості можуть бути вказані в файлах з розширенням * itg Кількість файлів даного виду визначається кількістю СДА в складі ПТК.
***.itm**	Текстовий	У, УО, УЗ, УЗО, УП, УПО, ОБД/СДА	Файли з розширенням *itm включають макрокоманди визначення структури блоків даних інформаційної взаємодії між компонентами ПТК. Дані відомості можуть бути визначені в окремих файлах із розширенням *.itm для спрощення внесення змін до системи.
***.itk**	Текстовий	У, УО, УЗ, УЗО, УП, УПО, ОБД/СДА	Файл з розширенням *.itk включає макрокоманди опису відповідності між фізичними каналами введення/виводу й технологічними параметрами системи управління.

Продовження табл. 2.2

Розширення файлу	Вид файлу	Тип проекту	Призначення файлу
***.itx**	Текстовий	У, УО, УЗ, УЗО, УП, УПО	Текстові файли даного виду містять секції прикладних програм на мові технологічного програмування. Повний склад прикладних програм може визначатися в декількох файлах даного типу.
***.dfb**	Текстовий	У, УО, УЗ, УЗО, УП, УПО	Текстовий файл даного виду містить тексти програм оголошення додаткового функціонального блоку (далі - ДФБ), використовуваного прикладною програмою КС МСКУ. Назва файлу повинна відповідати імені ДФБ і бути відмінним від імені проекту, до складу якого включається ДФБ. Повний склад використовуваних ДФБ для прикладних програм конкретної КС МСКУ може визначатися в декількох файлах даного типу.
***.inc**	Текстовий	У, УО, УЗ, УЗО, УП, УПО	Файли даного виду містять оголошення груп атрибутів параметрів, необхідних при описі інтерфейсу оголошення ДФБ. Повний склад оголошень груп атрибутів може визначатися в декількох файлах даного типу. Імена файлів даного типу можуть бути довільними.
***.bdb**	Об'єктний	У, УО, УЗ, УЗО, УП, УПО	Файли даного виду містять бібліотеки ДФБ, підготовлені за допомогою інструментальних засобів комплексу програм. Повний склад використовуваних бібліотек ДФБ для прикладних програм конкретної КС МСКУ може визначатися в декількох файлах даного типу (до складу бібліотек можуть входити як використовувані, так і не використовувані ДФБ у конкретній КС МСКУ).
Визначає користувач	Визначає користувач	ОБД/СДА	Додаткові файли для налаштування СДА. Підсистема відображення даних і ведення архівів на необхідний режим роботи й способи обробки даних, що приймаються ОБД СДА від компонентів ПТК
Визначає користувач	Визначає користувач	Будь-який (У, УО, УЗ, УЗО, УП, УПО, УТ, ОБД/СДА)	Додаткові файли, що включаються користувачем у підготовлювані комплекти постачання і комплекти архіву комплексів програм КС МСКУ і налаштувань ОБД СДА (документація і т. д.).

У результаті виконання підготовки проекту для МСКУ на диску ПЕОМ в каталозі кожного проекту буде сформовано виконуваний модуль КС МСКУ, призначений для запису в FLASH ПП КМП, а також модулі проекту з допоміжними даними. В результаті виконання підготовки проекту для КП ОД на диску ПЕОМ в каталозі кожного проекту буде сформовано виконуваний модуль

КП ОД, призначений для інсталяції на жорсткий диск робочої станції або ПЕОМ. У результаті виконання підготовки проекту ОБД/СДА на диску ПЕОМ у каталозі проекту будуть сформовані модулі налаштувань для ОБД СДА.

Підготовка КС МСКУ, КП ОД і налаштувань ОБД для КНР й СДА складається з наступних етапів:

- перший етап - вхідний контроль файлів *.itg, *.itx, *.itk (результатом є файли замовлення пам'яті *.xml, що містять опис породжуваних змінних для кожного КНР, в проекті якого вказаний файл (файли) *.itx);

- другий етап - генерація налаштувань ОБД (результатом є файли *.xdb, *.xcf, *.xdd для СДА, файли *.asm, *.xmp для КНР). Цей етап складається з декількох підетапів:

1.для кожного з проектів виконується формування блоків фонової діагностики (вхідним файлом є відповідний проекту файл *.itg, вихідним файлом - файл *.itp;

2.для кожного з проектів виконується обробка сформованого на попередньому етапі файлу *.itp замовлення пам'яті для поточного проекту файлу *.xml і файлу відповідності фізичних сигналів технологічними параметрами *.itk з формуванням проміжних файлів, що містять інформацію про локальну ОБД;

3.для кожного з проектів виконується формування блоків фонової діагностики вхідним файлом є відповідний проекту файл *.itg, вихідним файлом - файл *.itp;

4. для кожного з проектів виконується генерація результуючих файлів. Для МСКУ формуються файл інсталяції бази даних *.asm і файл карти пам'яті *.xmp; для СДА - файл опису апаратних засобів *.xcf, файл опису складу переданих блоків *.xdb і файл опису змінних бази даних *.xdd.

- третій етап - генерація кодів програм управління (результатом є файли trans *.asm з описом викликів функціональних блоків для МСКУ, КП ОД) з використанням файлів *.itx і підготовленого на другому етапі файлу *.xmp;

- четвертий етап - формування виконуваних модулів програм управління (результатом є файли *имя_проекта*.bin, *имя_части_згм*.bin для МСКУ, КП ОД);

- п'ятий етап - формування повного файлу з описом налаштувань ОБД для СДА (результатом є файли *имя_проекта_СДА*.csv).

2.1.8 Модулі діалогу записи у FLASH ПП

Модулі діалогу запису в FLASH ПП призначені для виклику модулів запису підготовлених проектів в FLASH ПП КМП.

Програма представляє собою набір взаємозв'язаних модулів:

- підпрограма запису проекту в FLASH ПП КМП;
- підпрограма скасування дії завантаження проекту під FLASH ПП КМП;
- підпрограма відображення повідомлень про помилки;
- підпрограма вибору лінії зв'язку для взаємодії з КМП.

2.2 Налаштування ОБД нижнього рівня СДА

Інструментальні засоби комплексу програм забезпечують одночасно з підготовкою завантажувальних модулів КС для всіх МСКУ і КП ОД підготовку налаштувань для ОБД СДА, що входять до складу ПТК.

Дії опису налаштувань ОБД для СДА включаються в файли з розширенням *.itg. Кількість файлів даного виду визначається кількістю СДА в складі ПТК.

Конфігурація ОБД СДА і дисципліна обміну даними з КНР описуються макрокомандами, послідовність яких в подальшому згадується як програма конфігурації ОБД СДА.

Програма конфігурації ОБД СДА може бути підготовлена будь-яким штатним редактором текстової інформації і є вихідним завданням для створення налаштувань ОБД СДА. Файл, що містить програму конфігурації ОБД СДА, повинен мати розширення *.itg.

2.3 Структура результуючих каталогів

Результуючі каталоги проекту для КНР, створені у відповідному каталозі групи проектів, мають структуру, показану на рис. 2. 3.

(каталог *имя_проекта*)

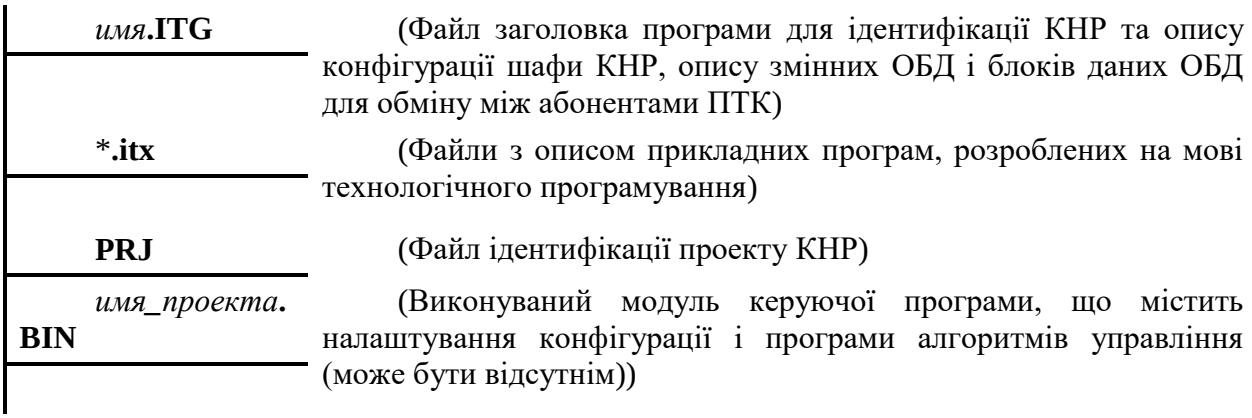


Рисунок 2. 3 – Структура результуючих каталогів проекту КНР

Результуючий каталог проекту для СДА, створений у відповідному каталозі групи проектів, має структуру, показану на рис.2. 4.

(каталог *имя_проекта*)

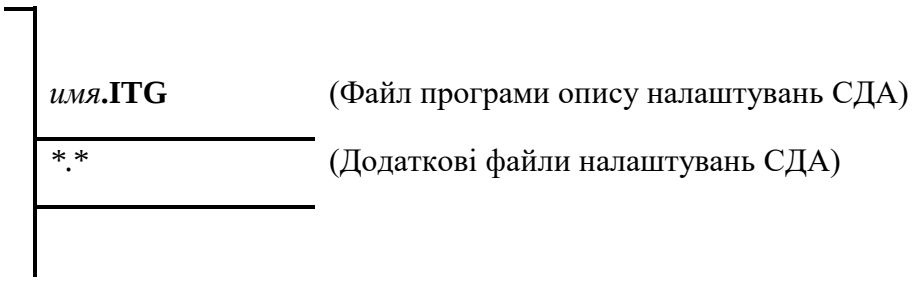


Рисунок 2. 4 – Структура результуючих каталогів проекту СДА

2.4 Структура каталогів групи проектів з робочою інформацією

Структура каталогів групи з робочою інформацією, використовуваної при підготовці групи, показана на рис. 2. 5.

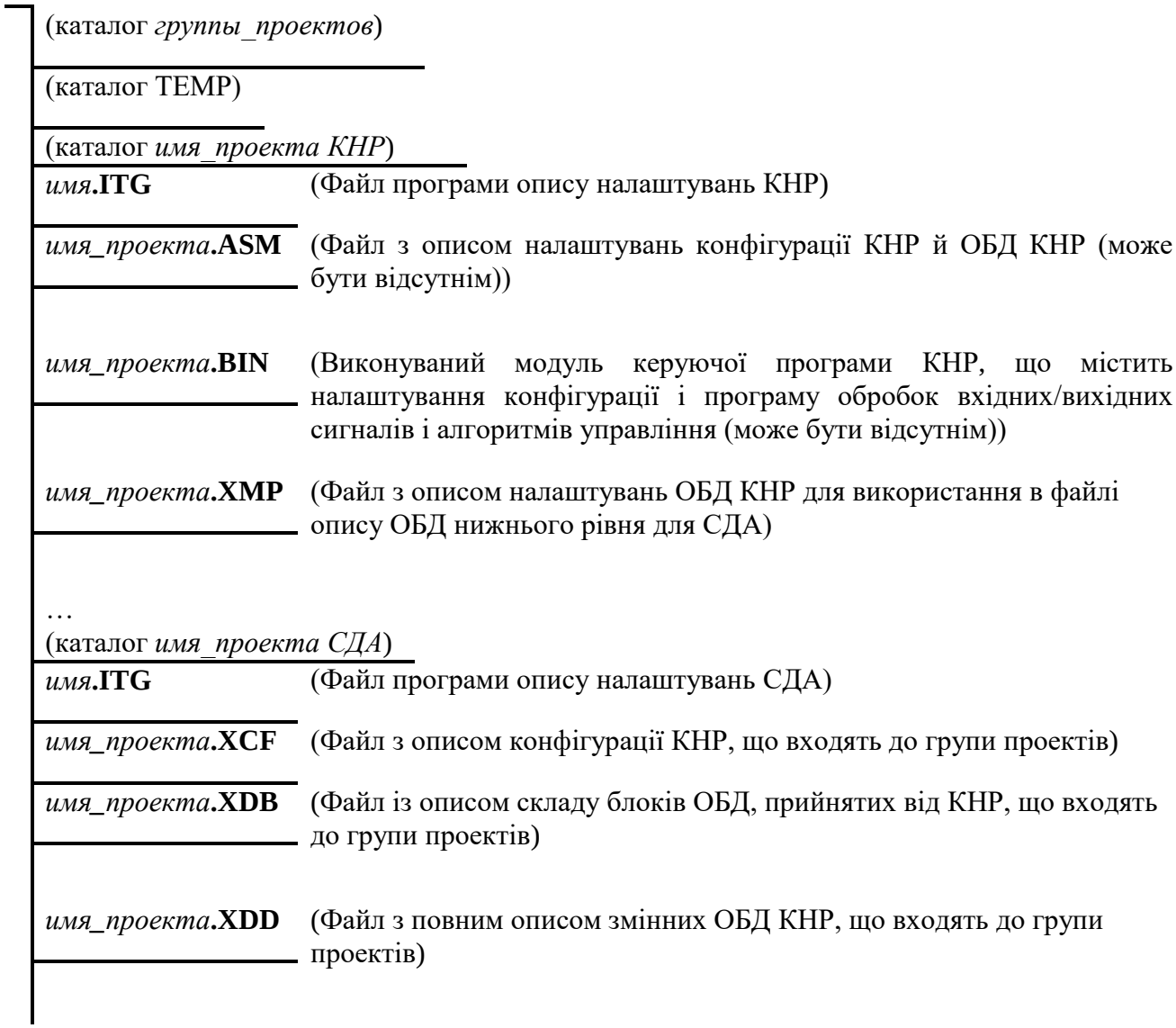


Рисунок 2. 5 – Структура каталогів групи проектів із робочою інформацією

2.5 Файл-заготовка *.xdd для формування дескрипторів ОБД для СДА

Загальний формат файлу *.xdd для формування дескрипторів ОБД для СДА має вигляд:

Назва файлу
Опис СДА
Опис фізичного параметра 1…
Опис фізичного параметра n
Опис технологічного параметра 1…
Опис технологічного параметра m
Закінчення опису СДА

Файл-заготівка *.xdd для формування дескрипторів ОБД для СДА готується за правилами мови XML і складається з обов'язкового константного заголовка, який оголошує відповідність версії мови XML, а також кодування файлу:

<?xml version="1.0" encoding="IBM866"?>

Після заголовка слідує опис СДА в наступному вигляді:

<pc address=”*address*”>
…
</pc>

де address - адреса СДА;
</pc> - тег, що обрамляє

Наприклад, для СДА з адресою 58 дескриптор СДА матиме такий вигляд:

<pc address=”58”>
…
</pc>

Усередині тегів, що обрамляють формується опис фізичних параметрів і технологічних параметрів.

Загальний формат опису фізичного параметра має наступний вигляд:

<phys name="*prefix.parameter.attribute*" type="*type*" [value="*value*"] [alias=”*alias*”]/>

де *prefix* - префікс фізичного параметра, що визначає приналежність до КНР;
parameter - найменування фізичного параметра;
attribute - найменування атрибута фізичного параметра;
type - тип фізичного параметра;
value - значення атрибута фізичного параметра (може бути відсутнім);
alias - вказівник посилання на відповідний технологічний параметр.

Загальний формат технологічного параметра має наступний вигляд:

<tech name="*prefix.parameter.attribute*" type="*type*" [value="*value*"]/>

де *prefix* - префікс технологічного параметра, що визначає приналежність до КНР;
parameter - найменування технологічного параметра;
attribute - найменування атрибута технологічного параметра;
type - тип технологічного параметра;
value - значення атрибута технологічного параметра (може бути відсутнім).

2.5.1 Файл-дескриптор складу блоків оперативних даних *.xdb

Загальний формат файлу-дескриптора складу блоків оперативних даних *.xdb має вигляд:

Назва файлу
Опис СДА
Опис блоку 1
Опис елемента 1 блоку 1

Опис елемента n блоку 1
Опис блоку 2
Опис елемента 1 блоку 2
...
Опис елемента m блоку 2
...
Опис блоку k
Опис елемента 1 блоку k
...
Опис елемента j блоку k
Закінчення опису СДА

Файл-дескриптор складу блоків оперативних даних *.xdb відповідає правилам мови XML з обов'язкового константного заголовка, який оголошує відповідність версії мови XML, а також кодування файлу.

Загальний формат опису блоку має наступний вигляд:

<bd nomer="*number*" type="*bdtype*" size="*size*" id="*id*" cycle="*cycle*" versionPO_knu="*version*" knu="*knu*" crate="*crate*">

де *number* - номер блоку, ціле число від 1 до 255;
bdtype - тип блоку. Набуває таких значень:
input - блок введення (приймається в СДА від абонента);
output - блок виведення (видається з СДА абоненту).
size - розмір блоку, в байтах;
id - тип ідентифікаторів, описаних у блоці. Набуває таких значень:
phys - ідентифікатор фізичного сигналу;
tech - ідентифікатор технологічного параметра.
cycle - цикл прийому/видачі блоку, в мілісекундах;
version - версія програмного забезпечення КНР, що видає/що приймає блок;
knu - адреса КНР, що видає/що приймає блок;
crate - номер крейта КНР, що видає/що приймає блок.

Загальний формат опису елемента блоку має наступний вигляд:

<elem offset="*offset*" name="*prefix.parameter.attribute*" [boffset="*bitoffset*" bsize="*bitsize*"]/>

де *offset* - зміщення елемента в блоці (в байтах);
prefix - префікс фізичного/технологічного параметра, що визначає приналежність до КНР;
parameter - найменування фізичного сигналу/технологічного параметра;
attribute - найменування атрибута фізичного сигналу/технологічного параметра;
bitoffset - бітовий зсув для бітових змінних (ціле число від 0 до 15);
bitsize - розмір (в бітах) для бітових змінних (ціле число від 1 до 7 і від 9 до 15).

2.5.2 Файл-дескриптор конфігурації КНР, що входять в групу проектів *.xcf

Загальний формат файлу- дескриптора конфігурації КНР, що входять у групу проектів *.xcf, має вигляд:

Назва файлу
Опис КНР1
Опис каналу 1 КНР1…
Опис каналу k КНР1
Опис шафи 1 КНР1
Опис сторони 1 шафи 1 КНР1
Опис поверху 1 сторони 1 шафи 1 КНР1
Опис крейта 1 поверху 1 сторони 1 шафи 1 КНР1
Опис блоку МЗО 1 КНР1
Опис каналу 1 блоку МЗО 1 КНР1…
Опис каналу n блоку МЗО 1 КНР1
Опис блоку МЗО 2 КНР1
Опис каналу 1 блоку МЗО 2 КНР1…
Опис каналу m блоку МЗО 2 КНР1…
Опис крейта 2 поверху 1 сторони 1 шафи 1 КНР1
Опис блоку МЗО i КНР1
Опис каналу 1 блоку МЗО i КНР1…
Опис каналу j блоку МЗО i КНР1…
Опис крейта q поверху 1 сторони 1 шафи 1 КНР1
Опис блоку МЗО f КНР1…
Опис поверху 2 сторони 1 шафи 1 КНР1…
Опис сторони 2 шафи 1 КНР1
Опис поверху 1 сторони 2 шафи 1 КНР1
Опис крейта 1 поверху 1 сторони 2 шафи 1 КНР1
Опис шафи 2 КНР1
Опис сторони 1 шафи 2 КНР1
Опис поверху 1 сторони 1 шафи 2 КНР1
Опис крейта 1 поверху 1 сторони 1 шафи 2 КНР1
Опис КНР2

Опис каналу 1 КНР2
Опис каналу m КНР2

Файл-дескриптор складу конфігурації КНР *.xcf відповідає правилам мови XML і включає обов'язковий константний заголовок, що оголошує відповідність версії мови XML, а також кодування файлу.

За заголовком слідують теги опису списку КНР:

<KNULIST> і </ KNULIST>, що обрамляють список дескрипторів КНР.

Всередині тегів <KNULIST> і </ KNULIST> для кожного КНР створюється опис КНР такого вигляду:

<KNU_Config name_knu="*mark*" haveDiag="*hd*" crates="*nc*">…
</KNU_Config>

де *mark* - маркування КНР, задана в параметрі МАРК макрокоманди #КНР;
Hd - ознака наявності в КНР, що видається фоновою діагностикою;
Nc - ступінь резервування КНР.

Всередині тегів <KNU_Config> і </ KNU_Config> поміщаються дескриптори каналів КНР і дескриптори шаф.

Дескриптор каналу КНР має наступний вигляд:

<KNU maps3adr="addrmaps" num_knu="address" num_crate_in_knu="crate" type_knu="tipi" versionPO_knu="verpo" name_crate_knu="mmark" poz="mu" ptk="ptk" havebd="havebd"/>

де
addrmaps - адреса КНР в мережі Ethernet НР;
address - адреса КНР в системі, заданій параметром АДРЕСА макрокоманди # КНР;
crate - номер крейта в КНР;
tipi - тип КНР, заданий параметром ТИПИ макрокоманди # КНР;
verpo - версія ПЗ КНР, задана параметром ВЕР макрокоманди # КНР;
mmark - маркування крейта КНР;
mu - символьний рядок, що визначає шифр (найменування) місця інсталяції;
ptk - ідентифікаційний номер ПТК;
havebd - ознака наявності в КНР блоків даних обміну з СДА.

Дескриптор каналів КНР матиме такий вигляд:

<KNU maps3adr="65" num_knu="1" num_crate_in_knu="1" type_knu="МСКУ 4.0" versionPO_knu="1" name_crate_knu="МСКУ КИП1.1" poz="HZ006" ptk="1" havebd="1"/>

<KNU maps3adr="129" num_knu="1" num_crate_in_knu="2" type_knu="МСКУ 4.0" versionPO_knu="1" name_crate_knu="МСКУ КИП1.2" poz="HZ006" ptk="1" havebd="1"/>

<KNU maps3adr="193" num_knu="1" num_crate_in_knu="3" type_knu="МСКУ 4.0" versionPO_knu="1" name_crate_knu="МСКУ КИП1.3" poz="HZ006" ptk="1" havebd="1"/>

За дескриптором каналів КНР слідують дескриптори шаф, що входять до КНР, які мають такий вигляд:

<Unit nsh="*nsh*" mark="*mark*" tipk="*tk*">…
</Unit>

де *nsh* - номер шафи;
mark - маркування шафи;
tk - тип компоновки шафи;
</unit> - тег, що обрамляє.

Всередині тегів опису шаф поміщаються описи сторін шаф, які мають такий вигляд:

<Storona markst="*mst*">…
</Storona>

де mst - маркування сторони шафи;
</Storona> - тег, що обрамляє

Всередині тегів опису сторін шаф поміщаються описи поверхів шаф, які мають такий вигляд:

<Etazh ne="*ne*" marke="*me*">
…
</Etazh>

де *ne* - номер поверха шафи;
me - маркування сторони шафи;
</Etazh> - тег, що обрамляє.

Всередині тегів опису поверхів шаф поміщаються описи крейти шаф, які мають такий вигляд:

<Crate nkr="*nk*" marke="*me*" [contr="*nkr*"]>
…
</Crate>

де *nk* - номер крейта поверха шафи;
me- маркування крейта поверха шафи;
nkr - ознака інсталяції у крейті контролера КМП, якщо параметр contr відсутній, то контролер не встановлений;
</Crate> - тег, що обрамляє.

Всередині тегів опису крейти поверхів шаф поміщаються опису модулів зв'язку МЗВ 41 і МЗВ-42 або МЗО.

Дескриптор модуля зв'язку МЗВ-41 має наступний вигляд:

<MSV1 nm="*nm*" tb="*tb*" ktb="*ktb*" fd_nb="*nb*" fd_offt="*offt*" fd_size="*size*"/>

де *nm* - номер модуля зв'язку МЗВ-41;
tb - тип блоку модуля зв'язку МЗВ-41;
ktb - код типу блоку модуля зв'язку МЗВ-41;
nb - номер блоку фонової діагностики, в якому передаються відомості про встановлений екземпляр МЗВ-41;
offt - зсув у блоці фонової діагностики, по якому передаються відомості про встановлений екземпляр МЗВ-41;
size - розмір відомостей щодо встановленого екземплярі МЗВ-41 в блоці фонової діагностики.

Дескриптор модуля зв'язку МЗВ-42 має наступний вигляд:

<MSV2 nm="*nks*" tb="*tb*" ktb="*ktb*" fd_nb="*nb*" fd_offt="*offt*" fd_size="*size*" [msv1=”*nm_msv1*”]/>

де nks - номер каналу зв'язку Ethernet, до якого підключений модуль зв'язку МЗВ-42;
tb - тип модуля зв'язку МЗВ-42;
ktb - код типу модуля зв'язку МЗВ-42;
nb - номер блоку фонової діагностики, в якому передаються відомості про встановлений екземплярі МЗВ-42;
offt - зміщення в блоці фонової діагностики, по якому передаються відомості про встановлений екземплярі МЗВ-42;
size - розмір докладну інформацію щодо встановленого екземплярі МЗВ-42 в блоці фонової діагностики;
nm_msv1 - номер місця інсталяції МЗВ-41, до якого підключений МЗВ-42.

Дескриптор МЗО має наступний вигляд:

<Block mesto="*nm*" type_block="*ctb*" name_block="*tb*" id="*id*" fd_nb="*nb*" fd_offt="*offt*" fd_size="*size*" [msv1=”*nm_msv1*”] [msv2=”*nm_msv2*”] [mu=”*mu*”] [flags=”*fl*”]>…

</Block>

де *nm* - номер місця МЗО, що задається в параметрі НМ макрокоманди #БЛОК;
ctb - код типу блоку;
tb - символьне найменування типу блоку;
id - фізичне ім'я блоку;
nb - номер блоку фонової діагностики, в якому передаються відомості про встановлений екземпляр МЗО;
offt - зміщення в блоці фонової діагностики, по якому передаються відомості про встановлений екземпляр МЗО;
size - розмір відомостей щодо встановленого екземпляру МЗО в блоці фонової діагностики;
nm_msv1 - номер місця інсталяції МЗВ-41 до якого підключений МЗО, або проміжний МЗВ-42, до якого підключений МЗО;
nm_msv2 - номер місця інсталяції МЗВ-42, до якого підключений МЗО;
mu - символьне найменування місця інсталяції;
fl - додаткові відомості про блок, якщо в fl міститься слово tblock, то встановлений МЗО є тройованим;
</block> - символьне найменування місця інсталяції.

Наприклад, для блоку ПНК-41/1 з адресою 1 дескриптор матиме такий вигляд:

<Block mesto="1" type_block="7" name_block="ПНК-41/1" id="B0001" fd_nb=”253” fd_offt=”11” fd_size=”49” mu=”1” >

Всередині тегів опису МЗО вкладаються описи каналів МЗО, які мають такий вигляд:

<Channel num_channel="*a*" ktk="*ktk*" kts="*kts*"/>

де *a* - адреса каналу в МЗО;
ktk - код типу каналу;
kts - символьне найменування типу каналу.

Наприклад, для каналу ПН2.5В з адресою 1 дескриптор матиме такий вигляд:

<Channel num_channel="1" ktk="1" kts="ПН2.5В"/>

Після опису шаф вкладаються описи груп взаємно резервованих МЗО, які мають такий вигляд:

<RezbGroup pngrb="pngrb" id="id">…

</RezbGroup>

де *pngrb* - порядковий номер групи взаємно резервованих МЗО;
id - фізичне ім'я групи взаємно резервованих МЗО;
RezbGroup - тег, що обрамляє.

Всередині тегів опису групи взаємно резервованих МЗО вкладаються опису посилань на фізичні імена МЗО, які мають такий вигляд:

<RezBlock id="id"/>

де id – фізичне ім'я МЗО, що входить до групи взаємно резервованих МЗО.

2.5.3 Файл налаштувань мережевих засобів, що функціонують в СДА ConfEthL*xxx*.xml

Загальний формат файлу ConfEthLxxx.xml налаштувань мережевих засобів

```
<?xml version="1.0" encoding="IBM866"?>
<root version="2">
<!—Опис мережі(ей) Ethernet нижнього рівня -->
<PCSettings myAdrName="адр" myName="марк" NumberNet="ксет" />
<NETLIST>
<!— Опис підключення робочої станції до першої мережі Ethernet нижнього рівня -->
        <NET_Config numberNET="нсет" MagCount="кмаг"
PTK_name="имя_ПТК" dtTime="ивр"
ptk="нптк" ptk_in_addr="аптк">
                <EthAdapter nm="нмаг1" name="ипкл1" MAC_addr="мак_адр">
<!— Список груп («поштових скриньок»), до складу яких входить робоча станція -->
                        <GROUPLIST GroupItemsCount="кгрупадр">
                                <GROUP Addr="адргр1" MAC_addr="мак_адр"/>
```

```
                    …
                    <GROUP Addr="адргрk" MAC_addr="мак_адр"/>
                </GROUPLIST>
            </EthAdapter>
…
            <EthAdapter nm="нмаг3" name="ипкл3" MAC_addr="мак_адр">
<!— Список груп («поштових скриньок»), до складу яких входить робоча
станція -->
                <GROUPLIST GroupItemsCount="кгрупадр">
                    <GROUP Addr="адргр1" MAC_addr="мак_адр"/>
                    …
                    <GROUP Addr="адргрk" MAC_addr="мак_адр"/>
                </GROUPLIST>
            </EthAdapter>
<!— Список абонентів і спосіб їх підключення -->
        <ABONENTLIST>
            <ABONENT address="адраб" crates="урез" name_KNU="имя_аб"
mark_KNU="марк_аб" type_Abn="тип_аб">
                <KNU num_crate_in_knu="нкрейт" maps3adr="адр_крейт"
name_crate_knu="марк_крейт"/>
                    <EthAdapter [" num_mag="нмагк"]
MAC_addr="мак_адр"/>
                </KNU>
        </ABONENT>
            …
        </ABONENTLIST>
        </NET_Config>
<!— Опис підключення робочої станції до другої мережі Ethernet нижнього
рівня -->
…
<!— Опис підключення робочої станції до n-ої мережі Ethernet нижнього
рівня -->
</NETLIST>
</root>
```

де *адр* - адреса робочої станції в мережах Ethernet HP (від 1 до 253);
марк - маркування робочої станції (символьний рядок, наприклад, "PC");
ксет - кількість мереж Ethernet HP, конфігурація яких описується в даному файлі (від 1 до 255);
нсет - номер мережі Ethernet HP (від 1 до 255);
кмаг - рівень резервування (кількість «магістралей») мережі Ethernet HP (1, 2, 3);
имя_птк - найменування ПТК (символьний рядок, наприклад, "МДЦ");
ивр - величина тайм-ауту очікування прийому даних для контролю конфігурації мережі (мереж) Ethernet HP, в секундах. За замовчуванням тайм-аут приймається рівним 5 s;
нптк - номер ПТК (від 0 до 255);
аптк - ознака використання номера ПТК при формуванні мережевих адрес абонентів мережі Ethernet HP;
нмагі (i=1, 2, 3) - номер «магістралі» мережі Ethernet HP (1, 2, 3);
ипкл, (i=1, 2, 3) - ідентифікатор, закріплений за Ethernet-адаптером підключення до мережі Ethernet HP в ОС Linux;
мак_адр - MAC-адреса, що встановлюється в Ethernet-контролер, який працює з магістраллю "нмаг";
адраб - адреса абонента в ПТК. Може приймати значення:
від 1 до 62 - для резервованих КНР;
від 1 до 253 - для нерезервованих КНР.
урез - рівень резервування КНР. Може приймати значення:
3 - для тройованих КНР;
6 - для шестиканального КНР;
1 - для нерезервованих КНР і СДА.
имя_аб - тип виконання КНР (символьний рядок, наприклад, "МСКУ-4.0/001");
марк_аб - маркування КНР МСКУ (символьний рядок, наприклад, "1 КІВ");
тип_аб - тип абонента мережі Ethernet HP. Може приймати значення:
0 - для трьохканального МСКУ-4;
1 - для нерезервованого КНР МСКУ;
2 - для робочої станції.
нкрейт - номер крейта КНН. Може приймати значення:
від 1 до 3 - для тройованих КНР;
1 - для нерезервованих КНР.
адр_крейт - адреса крейта в мережі Ethernet HP (від 1 до 253);
марк_крейт - маркування і-го (i = 1, 2, 3) крейта/шафи (символьний рядок, наприклад, "К1.1");
нмагк - номер «магістралі» мережі Ethernet HP (1, 2), до якої підключений крейт. Параметр відсутній, якщо дані приймаються за двома магістралями;
кгрупадр - кількість груп («поштових скриньок»), до складу яких входить робоча станція (від 1 до 62);
адргрk (k=1-10)- адреса групи («поштової скриньки»), до складу якої входить робоча станція (1≤ *адргрk* ≤ 63).

Файл налаштувань мережевих засобів ConfEthLxxx.xml готується за правилами мови XML і складається з обов'язкового константного заголовка, який оголошує відповідність версії мови XML, а також кодування файлу:

<? Xml version = "1.0" encoding = "IBM866"?>

Після константного заголовка випливає опис версії формату файлу в наступному вигляді:

<root>

…

</root>

де root – тег, що обрамляє.

Після заголовка випливає опис СДА в наступному вигляді:

<PCSettings myAdrName=”address” myName="name" NumberNet="nn" />

де *address* - адреса СДА;
name - маркування СДА;
nn - число мереж, в які входить СДА;
PCSettings - тег, що обрамляє.

Після опису СДА випливає дескриптор списку мереж в наступному вигляді:

<NETLIST…

</NETLIST>

де NETLIST - тег, що обрамляє.

Усередині дескриптора списку мереж присутні один або більше дескрипторів мережі наступного виду:

<NET_Config numberNET="nn" MagCount="nm" PTK_name="nptk" dtTime="dt">

де NET_Config – тег, що обрамляє;
nn – номер мережі Ethernet HP;
nm – число магістралей мережі Ethernet HP;
nptk – найменування ПТК;
dt – величина тайм-ауту очікування прийому даних для контролю конфігурації мережі (мереж) Ethernet HP, в мілісекундах.

За дескриптором мережі слідують дескриптории Ethernet-адаптерів робочої станції такого вигляду:

<EthAdapter nm="nm" name="eth">

де *nm* - номер магістралі;
eth - найменування мережевого інтерфейсу в ОС;
EthAdapter - тег, що обрамляє.

Якщо СДА має кілька групових адрес, то після дескриптора Ethernet-адаптера випливає дескриптор списку групових адрес виду:

<GROUPLIST GroupItemsCount="ngr">

де GROUPLIST – тег, що обрамляє;

ngr – число групових адрес для адаптера.

Усередині тега GROUPLIST розташовується кілька дескрипторів групових адрес виду:

<GROUP Addr="addr" MAC_addr="mcaddr">

де GROUP - тег, що обрамляє;
addr - адреса групи;
mcaddr - унікальний мережевий ідентифікатор групи (MAC-адреса).

Після опису Ethernet-адаптерів випливає дескриптор списку абонентів такого вигляду:

<ABONENTLIST>

…

</ABONENTLIST>

де ABONENTLIST – тег, що обрамляє.

Усередині дескриптора списку абонентів присутні один або більше абонентів такого вигляду:

<ABONENT address="addr" crates="ncr" name_KNU="tipi" mark_KNU="mark">

…

</ABONENT>

де ABONENT - тег, що обрамляє;
addr - адреса абонента в ПТК;
ncr - ступінь резервування абонента;
tipi - найменування типу абонента;
mark - маркування.

Усередині дескриптора абонента присутні один або кілька дескрипторів крейта/контролерів/напрямків КНР такого вигляду:

<KNU num_crate_in_knu="nc" maps3adr="ma" name_crate_knu="mark">

де KNU - тег, що обрамляє;
nc - номер крейта/контролера/напрямку КНР;
ma - адреса в мережі Ethernet НР;
mark - маркування крейта/контролера/напрямку.

2.4.4 Файл-заготівка для формування дескрипторів data_attrs

Загальний формат файлу для формування дескрипторів data_attrs має вигляд:

Заголовок файлу
Опис технологічного параметра 1
Опис атрибута 1 технологічного параметра 1
…
Опис атрибута й технологічного параметра 1
…
Опис технологічного параметра n
Опис атрибута 1 технологічного параметра n
…
Опис атрибута j технологічного параметра n
…
Опис робочого параметра 1
…
Опис робочого параметра m
Опис елемента обміну 1
…
Опис елемента обміну k

Файл-заготівка для формування дескрипторів data_attrs готується за правилами мови XML і складається з обов'язкового константного заголовка, який оголошує відповідність версії мови XML, а також кодування файлу:

<?xml version="1.0" encoding="koi8-r"?>

За заголовком слідують теги опису списку параметрів:

<rootentry name=”description”>

…

</rootentry>

де rootentry – тег, що обрамляє;

description – опис.

Всередині тегів <rootentry> і </ rootentry> зберігаються описатели технологічних, робочих параметрів і елементів обміну.

Опис технологічного параметра подається в наступному вигляді:

<tech name=”name”>...
</tech>

де tech – тег, що обрамляє;
name – назва параметра.

Усередині тегів, що обрамляють формується опис атрибутів технологічного параметра.

Загальний формат опису атрибута має наступний вигляд:

<attr name="*name*" type="*type*" [value="value"] [flags="*flag1, flagn*"] prefix="*prefix*" group="*group*" />

де *name* - ім’я атрибута;
type - тип атрибута;
value - значення атрибута;
flag1, flagn - прапори стану атрибута;
prefix - префікс технологічного параметра, що визначає приналежність до КНР;
group - назва групи, до якої відноситься атрибут.

Опис технологічних параметрів може чергуватися з описом робочих параметрів. Загальний формат опису робочого параметра:

<work name="*name*" type="*type*" flags="*flag1, flagn*" prefix="*prefix*" group="*group*"/>

де *name* - назва робочого параметра;
type - тип параметра;
flag1, flagn - прапори стану робочого параметра;
prefix - префікс робочого параметра, що визначає приналежність до КНР;
group - назва групи, до якої відноситься параметр.

Загальний формат опису елемента обміну:

<exch name="*name*" flags="*flag1*, *flagn*" [cycle="*cycle*" phase="*phase*"] />

де *name* - назва елемента обміну;
flag1, flagn - прапори елемента обміну;
cycle - цикл видачі елемента обміну, якщо відсутній, то елемент видається з циклом системи;
phase - фаза видачі елемента обміну, якщо відсутній, то вважається, що фаза має значення, рівне 0.

2.4.5 Файл-протокол результатів складання проектів

При виконанні підготовки групи проектів в каталозі групи проектів створюється файл протоколу з ім'ям build.log, в який записується результат збірки проектів.

Файл-протокол результатів складання проектів з ім'ям build.log розміщується в кореневому каталозі групи. Формат файлу:

Початок підготовки (дата: *день.місяць.рік* час: *години*:*хвилини*)
Ім'я_програми: *повідомлення* 1
…
Ім'я_програми: *повідомлення* N

Висновки до розділу 2

У другому розділі виконані:

- огляд логічної структури інструментальних засобів технологічного програмування;
- дослідження компонентів структури інструментальних засобів;
- опис кожного компонента структури інструментальних засобів.

3 ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ФОРМУВАННЯ КОМПЛЕКТІВ, ІНСТАЛЯЦІЇ КОМПЛЕКТІВ ПЗ КОМПОНЕНТІВ ПТК

3.1 Загальні відомості

Склад інструментальних програм формування комплектів, інсталяції комплектів ПЗ компонентів ПТК наведено в таблиці 3.1.

Таблиця 3.1 - Список інструментальних програм формування комплектів, інсталяції комплектів ПЗ компонентів ПТК

Ім'я файлу	Виконувані функції
fpo.exe	Програма формування файлу опису збірки комплектів ПЗ компонентів ПТК
finalization.exe	Програма управління процесом створення комплектів під час створення генерації групового проекту
PackCreator.exe	Програма підготовки комплектів ПЗ компонентів ПТК

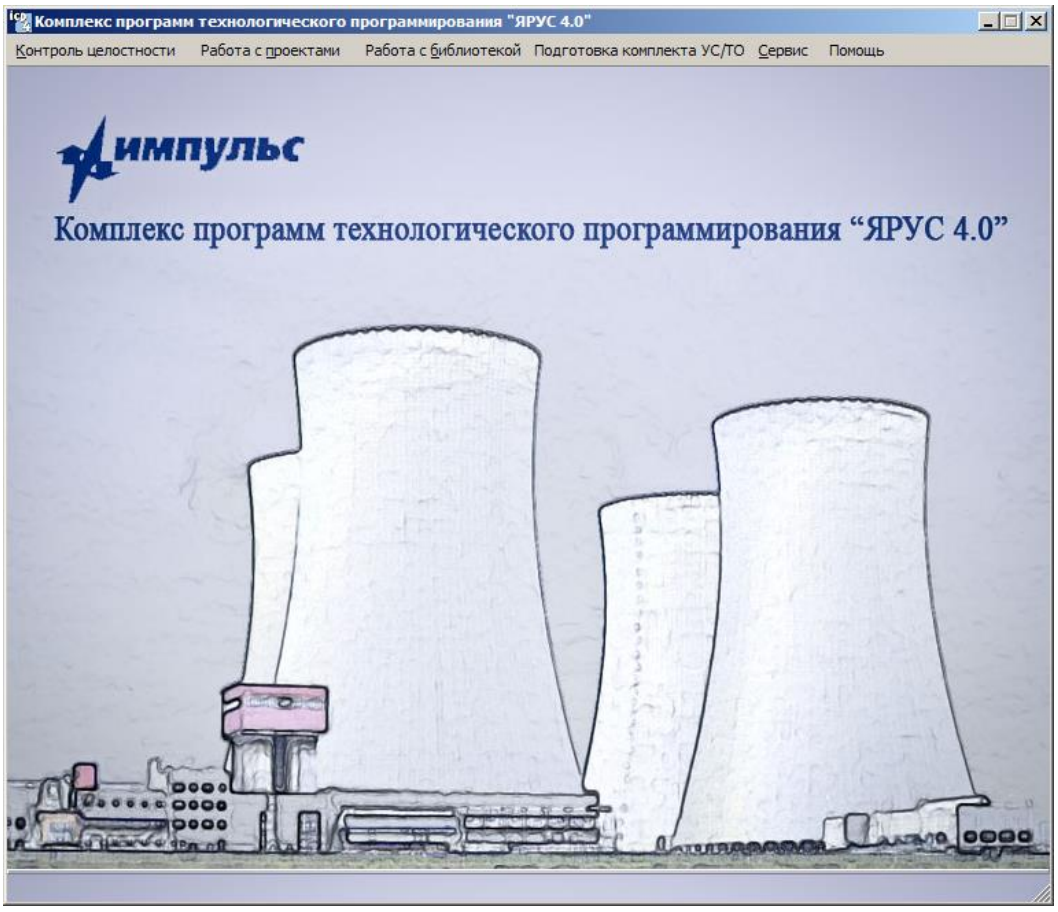


Рисунок 3.1 – Основний кадр інструментальних засобів підготовки

Інструментальні програми формування комплектів, інсталяції комплектів ПЗ компонентів ПТК використовують такі структури даних:

- каталог ia, що містить допоміжні засоби підготовки, інсталяції, контролю цілісності комплектів архіву й поставки комплектів ПЗ компонентів ПТК, а також засоби запису в FLASH ПП КМП;

- файл завдання на складання комплектів ПЗ PackList.xml;

- файл опису проекту prj. Для формування комплектів ПЗ компонентів ПТК із файлу prj використовуються поля TYPE, KMPVER, LARGE, PATHJDR, PATHWRFLASH.

3.2 Інструментальні засоби формування комплектів, інсталяції комплектів компонентів ПТК

3.2.1 Програма fpo

Програма формування файлу опису збірки комплектів із функціями «меню віконного» діалогу викликається з програмного монітора icp у режимі «Робота з проектами»> «Група»> «Формування завдання на складання комплектів ПЗ» й являє собою набір взаємозв'язаних модулів, перелік яких наведено в таблиці 3.2.

Таблиця 3.2 - Список модулів програми формування файлу опису збірки комплектів

Ім'я файлу	Ім'я підпрограми	Виконувані функції
main.cpp	main()	Головна підпрограма
pack2xml.ui.h	init()	Підпрограма ініціалізації основного вікна «Формування завдання на складання комплектів ПЗ»
pack2xml.ui.h	GrpDirSet()	Підпрограма зміни каталогу групового проекту
	NewComplect_clicked()	Підпрограма обробки натискання кнопки «Новий комплект»
	Params_clicked()	Підпрограма обробки натискання кнопки «Налаштування комплекту»
	Save_clicked()	Підпрограма обробки натискання кнопки «Зберегти»
	Prepare_clicked()	Підпрограма обробки натискання кнопки «Вихід»

Продовження табл. 3.2

Ім'я файлу	Ім'я підпрограми	Виконувані функції
cparams.ui.h	ins_file()	Підпрограма вибору файлів комплекту ПЗ компонентів ПТК
	pushButtonOk_clicked()	Підпрограма збереження налаштувань комплекту ПЗ компонентів ПТК
	pushButtonCancel_clicked()	Підпрограма скасування змін комплекту ПЗ компонентів ПТК

3.2.1.1 Опис підпрограми main()

Функціональне призначення

Підпрограма призначена для забезпечення діалогового режиму з користувачем і виводить на екран основне вікно «Формування завдання на складання комплектів ПЗ».

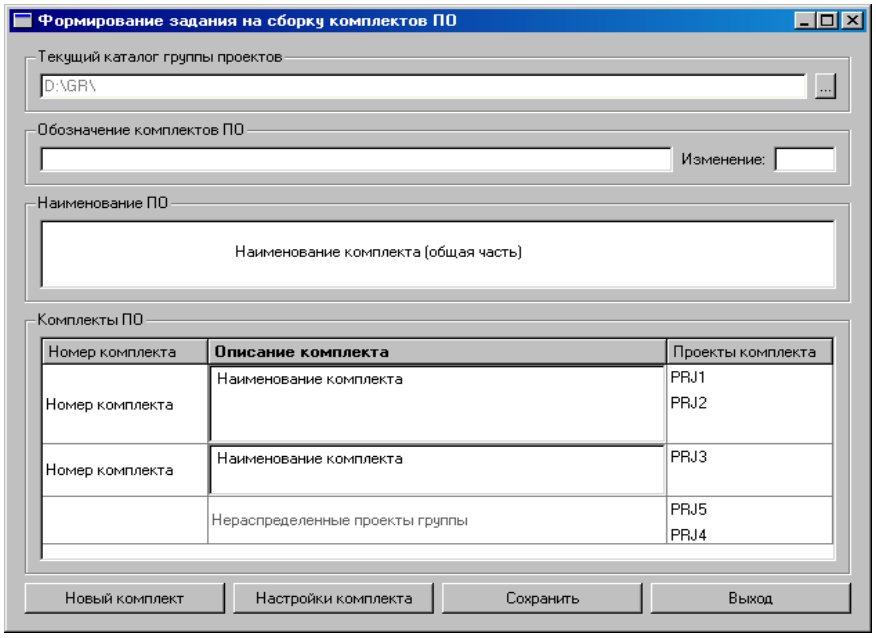


Рисунок 3.2 - Формат вікна «Формування завдання на складання комплектів ПЗ»

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Ініціалізація вхідних параметрів підпрограми.

Крок 2

Завантаження файлу pack2xml.qm з настройками написів російською мовою.

Крок 3

Ініціалізація і виведення на екран вікна «Формування завдання на складання комплектів ПЗ» і виконання основного циклу обробки, що забезпечує діалоговий режим роботи з користувачем.

Крок 4

Якщо параметр є заданим при запуску, виконуються ініціалізація вказаного каталогу в якості каталогу групового проекту й заповнення елементів вікна «Формування завдання на складання комплектів ПЗ».

Вхідні дані

Вхідними даними підпрограми є шлях до каталогу групового проекту.

Вихідні дані

Вихідними даними підпрограми є сформований файл опису збірки комплектів із ім'ям PackList.xml в каталозі групового проекту.

3.2.1.2 Опис підпрограми init()

Функціональне призначення

Підпрограма init() виконує ініціалізацію даних вікна «Формування завдання на складання комплектів ПЗ».

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Ініціалізація змінних підпрограми.

Крок 2

Виведення на екран вікна «Формування завдання на складання комплектів ПЗ».

Вхідні дані

Вхідними даними підпрограми є змінна оточення P40400_01.

Вихідні дані

Вихідними даними підпрограми є сформовані змінні.

3.2.1.3 Опис підпрограми GrpDirSet()

Функціональне призначення

Підпрограма GrpDirSet() виконує зміну каталогу групового проекту.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Виведення на екран додаткового вікна «Вибір групи» для вибору каталогу групового проекту.

Крок 2

Аналіз обраного каталогу групового проекту. Якщо каталог не є груповим проектом, виконується виведення на екран повідомлення про помилку.

Крок 3

Чистка елементів вікна «Формування завдання на складання комплектів ПЗ» від інформації попереднього групового проекту.

Крок 4

Читання з файлу grp списку файлів проектів із секції[prj] і повного шляху до вихідних файлів з секціїі[KROSS].

Крок 5

Читання даних із файлу опису PackList.xml і заповнення елементів вікна «Формування завдання на складання комплектів ПЗ».

Вхідні дані

Вхідними даними підпрограми є каталог групового проекту.

Вихідні дані

Вихідними даними підпрограми є заповнені елементи вікна «Формування завдання на складання комплектів ПЗ».

3.2.1.4 Опис підпрограми NewComplect_clicked()

Функціональне призначення

Підпрограма NewComplect_clicked() виконує обробку натиснення кнопки «Новий комплект» вікна «Формування завдання на складання комплектів ПЗ».

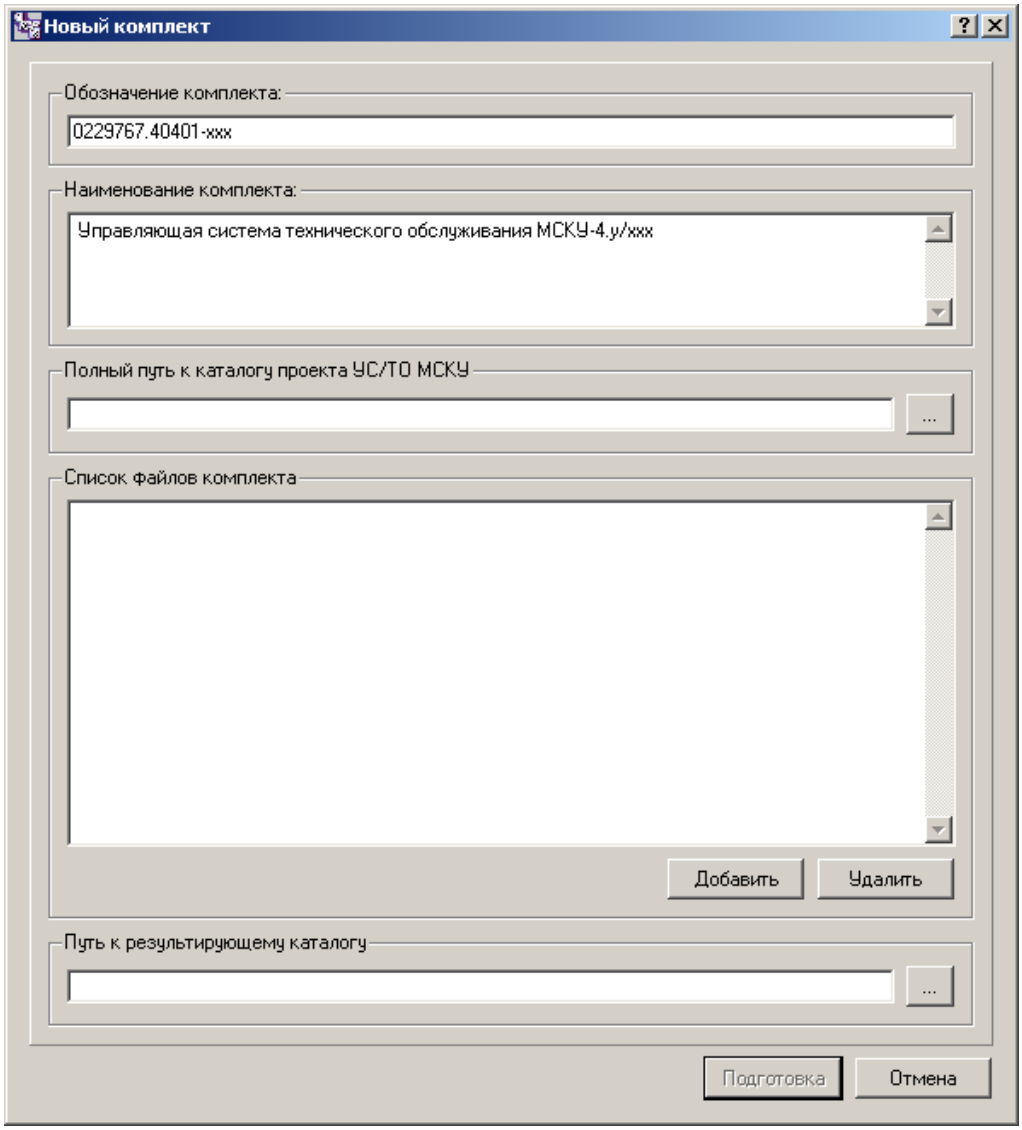


Рисунок 3.3 - Формат вікна «Створити комплект»

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Доповнення рядка в таблицю «Комплекти ПЗ», що надає можливість введення інформації про новий комплект.

Крок 2

Установка ознаки внесення змін, що дорівнює 1.

Вхідні дані

Вхідні дані відсутні.

Вихідні дані

Вихідними даними підпрограми є доданий рядок таблиці «Комплекти ПЗ».

3.2.1.5 Опис підпрограми Params_clicked()

Функціональне призначення

Підпрограма Params_clicked() виконує обробку натиснення кнопки «Налаштування комплекту» вікна «Формування завдання на складання комплектів ПЗ».

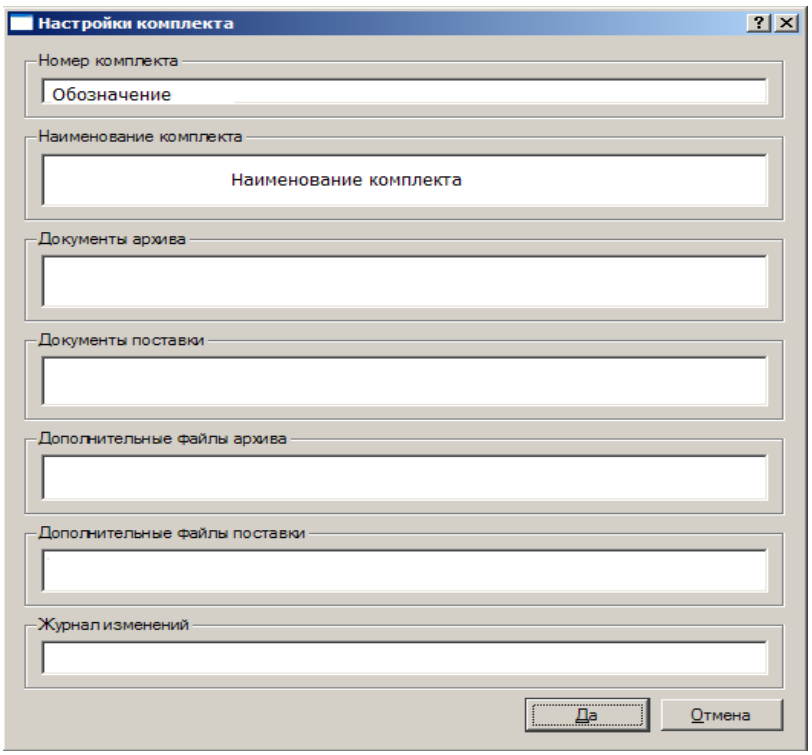


Рисунок 3.4 - Формат вікна «Налаштування комплекту»

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Аналіз вхідного параметра. Якщо поточний номер рядка таблиці «Комплекти ПЗ» відповідає елементу «Нерозподілені комплекти групи», слід завершити виконання.

Крок 2

Аналіз номера комплектів ПЗ на коректність. Якщо номер комплектів ПЗ вказаний невірно, виводиться повідомлення про помилку.

Крок 3

Ініціалізація даних і висновок на екран вікна «Налаштування комплекту» з заповненими елементами.

Вхідні дані

Вхідними даними підпрограми є номер рядка таблиці «Комплекти ПЗ».

Вихідні дані

Вихідними даними підпрограми є сформовані списки з іменами файлів документації поставки й архіву, а також списки додаткових файлів користувача.

3.2.1.6 Опис підпрограми Save_clicked()

Функціональне призначення

Підпрограма Save_clicked() виконує обробку натиснення кнопки «Зберегти» вікна «Формування завдання на складання комплектів ПЗ».

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Створення файлів readme для всіх комплектів ПЗ, зазначених у таблиці «Комплекти ПЗ».

Крок 2

Збереження даних таблиці «Комплекти ПЗ» у файл з ім'ям PackList.xml у поточному каталозі групового проекту.

Крок 3

Встановити ознаку внесення змін рівним 0.

Вхідні дані

Вхідними даними підпрограми є дані таблиці «Комплекти ПЗ» вікна «Формування завдання на складання комплектів ПЗ».

Вихідні дані

Вихідними даними підпрограми є сформований файл опису PackList.xml в поточному каталозі групового проекту.

3.2.1.7 Опис підпрограми Prepare_clicked()

Функціональне призначення

Підпрограма Prepare_clicked() виконує обробку натиснення кнопки «Вихід» вікна «Формування завдання на складання комплектів ПЗ».

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Аналіз ознаки внесення змін. Якщо ознака не дорівнює 0, виводиться на екран додаткове вікно з повідомленням «Виявлені зміни. Зберегти й вийти?» і кнопками «Так»,«Ні» і «Скасувати».

Крок 2

Якщо натиснули кнопку «Так», виконуються збереження даних вікна «Формування завдання на складання комплектів ПЗ» у файл опису PackList.xml і вихід з підпрограми.

Крок 3

Якщо натиснули кнопку «Ні», виконується вихід ыз підпрограми без збереження даних.

Крок 4

Якщо натиснули кнопку «Скасувати», додаткове вікно з повідомленням про зміни закривається й виконується повернення в вікно «Формування завдання на складання комплектів ПЗ».

Вхідні дані

Вхідними даними підпрограми є дані вікна «Формування завдання на складання комплектів ПЗ».

Вихідні дані

Вихідними даними підпрограми є сформований файл опису PackList.xml у каталозі групового проекту.

3.2.1.8 Опис підпрограми ins_file()

Функціональне призначення

Підпрограма ins_file() надає можливість користувачу в діалоговому режимі виконати вибір файлів комплектів ПЗ.

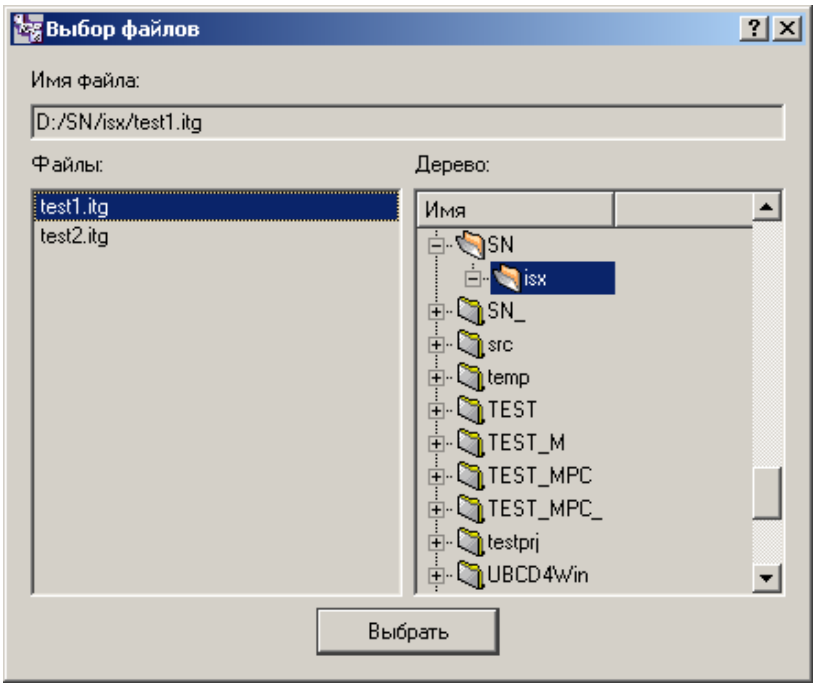


Рисунок 3.5 - Формат вікна «Вибір файлів»

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Ініціалізація каталогу розміщення вихідних файлів і висновок на екран додаткового вікна «Вибір файлів».

Крок 2

Формування списку вибраних файлів після натиснення кнопки «Вибір».

Вхідні дані

Вхідними даними підпрограми є каталог групового проекту.

Вихідні дані

Вихідними даними підпрограми є сформований список файлів.

3.2.1.9 Опис підпрограми pushButtonOk_clicked()

Функціональне призначення

Підпрограма pushButtonOk_clicked() виконує закриття вікна «Налаштування комплекту» зі збереженням налаштувань.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Збереження списків обраних файлів і закриття вікна «Налаштування комплекту».

Крок 2

Установлення ознаки внесення змін.

Вхідні дані

Вхідними даними підпрограми є списки файлів вікна «Налаштування комплекту».

Вихідні дані

Вихідними даними підпрограми є збережені списки файлів і сформована ознака внесення змін.

3.2.1.10 Опис підпрограми pushButtonCancel_clicked()

Функціональне призначення

Підпрограма pushButtonCancel_clicked() виконує закриття вікна «Налаштування комплекту» без збереження змін.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Очищення ознаки змін вікна «Налаштування комплектів».

Крок 2

Закриття вікна «Налаштування комплектів». Повернення у вікно «Формування завдання на складання комплектів ПЗ».

Вхідні дані

Вхідні дані підпрограми відсутні.

Вихідні дані

Вихідні дані підпрограми відсутні.

3.2.2 Програма finalization

3.2.2.1 Загальні відомості

Програма управління процесом підготовки комплектів ПЗ призначена для запуску на виконання засобів підготовки комплектів ПЗ компонентів ПТК у процесі створення генерації групового проекту.

Програма представляє собою модуль з ім'ям main().

3.2.2.2 Опис підпрограми main()

Функціональне призначення

Підпрограма main() є головною підпрограмою, яка запускає на виконання послідовність програм, певну в файлі конфігурації finalization.cfg.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Виконання розбору настроювальних параметрів виклику програми.

Крок 2

Перевірка необхідних для роботи параметрів, переданих користувачем через параметри запуску програми.

Крок 3

Читання даних із файлу конфігурації finalization.cfg.

Крок 4

Запуск на виконання послідовності програм, певної у файлі конфігурації. Кожна наступна програма із списку файлу конфігурації запускається на виконання тільки після завершення виконання попередньої програми.

Вхідні дані

Вхідними даними підпрограми є параметри запуску, передані через командний рядок, і дані, що містяться в файлі конфігурації finalization.cfg.

Вихідні дані

Вихідними даними підпрограми є відомості про успішність виконання запущених програм.

3.2.3 Програма PackCreator

3.2.3.1 Загальні відомості

Програма підготовки комплектів ПЗ призначена для формування комплектів ПЗ на носіях у спеціальному упакованому форматі зі збереженням контрольних відомостей про конкретні комплекси програм (далі - комплекс програм), визначених у файлі опису PackList.xml, версії комплексів програм (наприклад, для передачі в архів).

Програма представляє собою набір взаємозв'язаних модулів, перелік яких наведено в таблиці 3.3.

Таблиця 3.3 - Список модулів програми підготовки комплектів ПЗ

Ім'я файлу	Ім'я підпрограми	Виконувані функції
main.cpp	main()	Головна підпрограма
SupplyCreator.cpp	TC_SupplyCreator()	Підпрограма формування комплектів поставки для послідовності комплексів програм, певної у настроювальному файлі PackList.xml
	TC_WorkWithPrj()	Підпрограма формування в комплекті поставки даних про окремий проект групового проекту, що входить до складу комплексу програм
ArchiveCreator.cpp	TC_ArchiveCreator()	Підпрограма формування комплектів архіву для послідовності комплексів програм, певної у настроювальному файлі PackList.xml

3.2.3.2 Опис підпрограми main()

Функціональне призначення

Підпрограма призначена для формування у директорії INSTALL групового проекту каталогів із файлами комплектів ПЗ для кожного певного в файлі опису PackList.xml комплексу програм.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Виконання розбору настроювальних параметрів виклику головною підпрограми.

Крок 2

Визначення режиму роботи, перевірка необхідних для роботи параметрів, переданих користувачем через параметри запуску програми.

Крок 3

Формування комплектів ПЗ у залежності від обраного режиму роботи програми.

Вхідні дані

Вхідними даними підпрограми є параметри запуску, передані через командний рядок.

Вихідні дані

Вихідними даними підпрограми є каталоги з файлами комплектів ПЗ, розташовані в директорії INSTALL групового проекту.

3.2.3.3 Опис підпрограми TC_SupplyCreator()

Функціональне призначення

Підпрограма призначена для формування даних комплекту поставки для кожного певного в файлі опису PackList.xml комплексу програм.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Читання даних із файлу опису комплексів програм PackList.xml.

Крок 2

Для кожного комплексу програм, визначеного в файлі опису PackList.xml, виконання кроків з 3 по 8.

Крок 3

Отримання переліку проектів групового проекту, що входять до складу комплексу програм.

Крок 4

Додавання до комплекту поставки даних про кожний проект групового проекту, що входить до складу комплексу програм, за допомогою функції TC_WorkWithPrj().

Крок 5

Додавання до комплекту поставки файлів експлуатаційної документації, журналу змін, а також короткої інструкції по контролю цілісності та інсталяції комплексу програм.

Крок 6

Додавання до комплекту поставки додаткових виконуваних файлів, що входять до складу комплексу програм.

Крок 7

Створення в комплекті поставки файлів сервісних програм інсталяції, контролю цілісності та файлу з коротким описом комплекту.

Крок 8

Створення файлу контрольних сум файлів, що входять до складу комплекту поставки.

Вхідні дані

Вхідними даними підпрограми є файл опису комплексів програм PackList.xml.

Вихідні дані

Вихідними даними підпрограми є каталоги з файлами комплектів поставки комплексів програм, визначених у файлі опису PackList.xml.

3.2.3.4 Опис підпрограми TC_WorkWithPrj()

Функціональне призначення

Підпрограма призначена для формування в комплекті поставки даних про окремий проект групового проекту, що входить до складу комплексу програм.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Читання файлу опису проекту.

Крок 2

Визначення типу проекту. Якщо тип проекту «ОБД/СДА», виконується перехід до кроку 3, інакше - виконання кроку 4.

Крок 3

Пошук у кореневому каталозі групового проекту файлу Имя_проекта.zip. Якщо файл знайдений, для кожного файлу, що входить до архіву, виконується формування файлу контрольних сум, який додається до zip-архіву, а сам zip-архів копіюється в каталог комплекту поставки.

Крок 4

Додавання до архівного файлу Рномер_комплекса_программ.pak, що розташований у каталозі формованого комплекту поставки, файлу опису проекту prj, файлу виконуваних модулів програми управління Імя_проекта.bin і засобів для запису програм управління в постійний запам'ятовуючий пристрій відповідного компонента ПТК.

Вхідні дані

Вхідними даними підпрограми є:

- ім'я проекту;

- шлях до каталогу проекту;

- номер комплексу програм, до складу якого входить даний проект.

Вихідні дані

Вихідними даними підпрограми є заархівовані файли з даними проекту, які зберігаються в каталозі формованого комплекту поставки.

3.2.3.5 Опис підпрограми TC_ArchiveCreator()

Функціональне призначення

Підпрограма призначена для формування даних комплекту архіву для кожного певного в файлі опису PackList.xml комплексу програм.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Визначення шляху до каталогу з файлами комплекту архіву групового проекту.

Крок 2

Читання даних із файлу опису комплексів програм PackList.xml.

Крок 3

Для кожного комплексу програм, визначеного у файлі опису PackList.xml, виконання кроків із 4 по 17.

Крок 4

Отримання переліку проектів групового проекту, що входять до складу комплексу програм.

Крок 5

Для кожного проекту, що входить до складу комплексу програм, виконання кроків із 6 по 8.

Крок 6

Додавання до файлу Sномер_комплекса_программ.pak, який розташований у каталозі комплекту архіву, файлів, перерахованих у секціях [info], [scr], [vis], [rez] файлу опису проекту prj.

Крок 7

Додавання в файл Sномер_комплекса_программ.pak файлу опису проекту prj, в якому абсолютні шляхи до каталогу файлів комплекту архіву групового проекту замінюються на рядок SRC_DIR.

Крок 8

Визначення типу проекту. Якщо тип проекту «ОБД/СДА», проводиться пошук у кореневому каталозі групового проекту файлу Имя_проекта.zip. Якщо файл знайдений, здійснюється копіювання zip-архіву в каталог комплекту архіву.

Крок 9

Додавання до файлу Sномер_комплекса_программ.pak файлу кросування.

Крок 10

Отримання переліку файлів проектів для підсистеми відображення *.xprj, що входять до складу групового проекту.

Крок 11

Для кожного файлу *.xprj, що входить до складу групового проекту, виконання кроків з 12 по 13.

Крок 12

Виконання перевірки, чи використовується файл *.xprj компонентами ПТК, що входять до складу комплексу програм, для якого формується комплект архіву.

Крок 13

Якщо файл *.xprj використовується компонентами ПТК, відбувається додавання в файл Sномер_комплекса_программ.pak перерахованих в *.xprj відповідних їм файлів *.xml, використовуваних бібліотек *.bdb, а також самого файлу *.xprj, в якому абсолютні шляхи до каталогу файлів вихідних текстів групового проекту замінюються на рядок SRC_DIR.

Крок 14

Додавання до файлу Sномер_комплекса_программ_DOC.pak, розташований у каталозі комплекту архіву, файлів документації.

Крок 15

Додавання до файлу Sномер_комплекса_программ.pak додаткових файлів як текстових, так і двійкових, що входять до складу комплексу програм.

Крок 16

Створення в комплекті архіву файлів сервісних програм інсталяції, контролю цілісності та файлу з коротким описом комплекту.

Крок 17

Створення файлу контрольних сум файлів, що входять до складу комплекту архіву.

Вхідні дані

Вхідними даними підпрограми є файл опису комплексів програм PackList.xml.

Вихідні дані

Вихідними даними підпрограми є каталоги з файлами комплектів архіву комплексів програм, визначених у файлі опису PackList.xml.

3.2.4 Програма PackInstaller

3.2.4.1 Загальні відомості

Програма інсталяції комплекту архіву призначена для виконання наступних функцій:

- установки файлів комплекту архіву компонентів ПТК до цільового групового проекту;

- додавання необхідних даних для реєстрації проектів, створених при установці комплекту архіву в цільовому груповому проекті;

- перевірки цілісності в груповому проекті встановлених комплектів.

Програма інсталяції працює в діалоговому режимі з користувачем і має два режими запуску:

- інсталяція комплекту архіву в груповий проект;

- перевірка цілісності комплекту архіву, встановленого в груповому проекті.

Основне вікно програми є класом PackInstaller. Перелік основних модулів класу PackInstaller наведено в таблиці 3.4.

Таблиця 3.4 - Список основних модулів класу PackInstaller

Ім'я файлу	Ім'я підпрограми	Виконувані функції
PackInstaller.cpp	run()	Підпрограма підготовки даних для основного діалогового вікна програми інсталяції комплектів архіву
	runInstallMode()	Підпрограма формування даних, необхідних для виконання інсталяції комплекту архіву
	runVerifyMode()	Підпрограма формування даних, необхідних для виконання перевірки цілісності встановленого комплекту архіву
	beginInstallProcess()	Підпрограма, яка виконує інсталяцію комплекту архіву
PackInstallerHandlers.cpp	OnCheckPackNum()	Підпрограма перевірки цілісності окремого комплекту архіву
	OnCheckPackNums()	Підпрограма перевірки цілісності встановлених комплектів архіву
	OnInstall()	Підпрограма, що запускає на виконання процес інсталяції комплекту архіву

3.2.4.2 Опис підпрограми run()

Функціональне призначення

Підпрограма призначена для встановлення початкових значень основного вікна відповідно до заданого користувачем режимом роботи програми.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Виконання розбору настроювальних параметрів виклику головною підпрограми.

Крок 2

Визначення режиму роботи, перевірка необхідних для роботи параметрів, переданих користувачем через параметри запуску програми.

Крок 3

Установка початкових значень основного вікна відповідно до заданого користувачем режимом роботи програми інсталяції за допомогою підпрограми runinstallmode() або підпрограми runverifymode(), в залежності від режиму запуску програми.

Вхідні дані

Вхідними даними підпрограми є параметри запуску, передані через командний рядок.

Вихідні дані

Вихідними даними є діалогове вікно, відповідне заданому користувачем режиму роботи програми.

3.2.4.3 Опис підпрограми runinstallmode()

Функціональне призначення

Підпрограма призначена для підготовки інформації про комплект архіву, що встановлюється, яка відображається в основному діалоговому вікні програми в режимі інсталяції.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Читання інформації про комплект архіву(номер комплекту архіву, опис та дата складання комплекту архіву, список проектів, що входять до складу комплекту архіву), що встановлюється.

Крок 2

Відображення в основному вікні програми інсталяції інформації про комплект архіву, що встановлюється.

Вхідні дані

Вхідні дані відсутні.

Вихідні дані

Вихідними даними є інформація про комплект архіву, що встановлюється, яка відображається в основному вікні програми.

3.2.4.4 Опис підпрограми runverifymode()

Функціональне призначення

Підпрограма призначена для підготовки інформації про вже встановлені в груповому проекті комплекти архіву, яка відображається в основному діалоговому вікні програми інсталяції у режимі перевірки цілісності.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Читання інформації про вже встановлені в груповому проекті комплекти архіву (номер комплекту архіву, опис, дата інсталяції та дата останньої збірки генерації комплекту архіву).

Крок 2

Відображення в основному вікні програми інсталяції інформації про вже встановлені в груповому проекті комплекти архіву.

Вхідні дані

Вхідні дані відсутні.

Вихідні дані

Вихідними даними є інформація про вже встановлені в груповому проекті комплекти архіву, яка відображається в основному вікні програми.

3.2.4.5 Опис підпрограми begininstallprocess()

Функціональне призначення

Підпрограма призначена для виконання установки файлів комплекту архіву в груповий проект і додавання необхідних даних для реєстрації проектів, створених при встановленні комплекту архіву в груповому проекті.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Виконання перевірки, встановлений в груповому проекті даний комплект архіву. Якщо встановлений, виконання кроку 2, інакше - перехід до кроку 3.

Крок 2

Запит у користувача на підтвердження про продовження інсталяції із заміною файлів раніше встановленого комплекту архіву. Якщо підтвердження отримано, виконання кроку 3, інакше - завершення виконання підпрограми.

Крок 3

Підготовка файлів встановлюваного комплекту архіву для копіювання в каталог групового проекту й каталог файлів, що містить вихідні тексти програм

групового проекту. В процесі порядкової обробки файлів у кожному рядку перевіряється наявність підрядка «$ SRC_DIR $». Якщо дана підстрока присутня в рядку, то в цьому рядку вона замінюється значенням абсолютного шляху до каталогу файлів, який містить вихідні тексти програм групового проекту.

Крок 4

Копіювання файлів встановлюваного комплекту архіву до каталогу групового проекту й каталог файлів, що містить вихідні тексти програм групового проекту.

Крок 5

Додавання даних про комплект архіву, що встановлюється в файл опису комплексів програм packlist.xml.

Крок 6

Додавання даних про проекти, що входять до складу комплекту архіву, що встановлюється в файл опису групового проекту grp.

Крок 7

Додавання даних для перевірки цілісності встановлюваного комплекту архіву в директорію CHKSUM, розташовану в каталозі файлів, що містить вихідні тексти програм групового проекту.

Крок 8

Додавання еталонної копії встановленого комплекту архіву в каталог ETALONS групового проекту.

Крок 9

Висновок на екран результатів виконання інсталяції комплекту архіву.

Вхідні дані

Вхідними даними підпрограми є файли встановлюваного комплекту архіву і шлях до каталогу групового проекту, в який інсталюється комплект архіву.

Вихідні дані

Вихідними даними підпрограми є каталоги з файлами комплекту архіву, додані до групового проекту.

3.2.4.6 Опис підпрограми OnCheckPackNum()

Функціональне призначення

Підпрограма призначена для перевірки цілісності комплекту архіву, встановленого в груповому проекті.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Визначення шляху до файлу з контрольними сумами файлів комплекту архіву, що перевіряється.

Крок 2

Порівняння контрольних сум файлів комплекту архіву з відповідними контрольними сумами, що містяться в файлі, визначеному при виконанні кроку 1.

Крок 3

Відображення в основному вікні програми інсталяції інформації про результати перевірки цілісності.

Вхідні дані

Вхідними даними підпрограми є номер комплекту архіву, для якого виконується перевірка цілісності.

Вихідні дані

Вихідними даними підпрограми є інформація про результати перевірки цілісності комплекту архіву.

3.2.4.7 Опис підпрограми OnCheckPackNums()

Функціональне призначення

Підпрограма призначена для перевірки цілісності комплектів архіву, встановлених у груповому проекті.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Ініціалізація вхідних параметрів підпрограми.

Крок 2

Для кожного комплекту архіву виконання перевірки цілісності за допомогою функції oncheckpacknum().

Вхідні дані

Вхідними даними підпрограми є список номерів комплектів архіву, для яких виконується перевірка цілісності.

Вихідні дані

Вихідними даними підпрограми є інформація про результати перевірки цілісності комплектів архіву.

3.2.4.8 Опис підпрограми oninstall()

Функціональне призначення

Підпрограма призначена для запуску процесу інсталяції у груповому проекті комплекту архіву.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Перевірка, чи є директорія, шлях до якої переданий у вхідних даних, каталогом групового проекту. Якщо є, перехід до кроку 3, інакше - перехід до кроку 2.

Крок 2

Створення шаблону групового проекту.

Крок 3

Генерація події, що приводить до початку виконання процесу інсталяції комплекту архіву.

Вхідні дані

Вхідними даними підпрограми є шлях до директорії, в яку необхідно проінсталювати комплект архіву.

Вихідні дані

Вихідні дані відсутні.

3.2.5 Програма PackComporator

3.2.5.1 Загальні відомості

Програма порівняння проінстальованих комплектів ПЗ із новими підготовленими призначена для визначення відмінності даних у проінстальованих (еталонних) комплектах і в комплектах, створених засобами програми підготовки комплектів у процесі створення генерації групового проекту.

Програма представляє собою набір взаємозв'язаних модулів, перелік яких наведено в таблиці 3.5.

Таблиця 3.5 - Список модулів програми порівняння проінстальованих комплектів ПЗ із новими підготовленими комплектами

Ім'я файлу	Ім'я підпрограми	Виконувані функції
main.cpp	main()	Головна підпрограма
packcomporatorhelper.cpp	tc_comparepackages()	Підпрограма визначення переліку файлів комплекту ПЗ, створеного програмою підготовки, змінених у порівнянні з відповідним проінстальованим (еталонним) комплектом ПЗ

3.2.5.2 Опис підпрограми main()

Функціональне призначення

Підпрограма призначена для визначення відмінності в комплектах ПЗ, створених програмою підготовки, й відповідних проінстальованих (еталонних) комплектах, розташованих у каталозі ETALONS групового проекту.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Виконання розбору настроювальних параметрів виклику головною підпрограми.

Крок 2

Перевірка необхідних для роботи параметрів, переданих користувачем через параметри запуску програми.

Крок 3

Читання даних із файлу опису комплексів програм packlist.xml.

Крок 4

Для кожного комплексу програм, визначеного в файлі опису packlist.xml, відбувається отримання переліку файлів комплектів ПЗ, створених програмою підготовки, які були змінені в порівнянні з відповідними проінстальованими (еталонними) комплектами за допомогою функції TC_ComparePackages().

Вхідні дані

Вхідними даними підпрограми є параметри запуску, передані через командний рядок.

Вихідні дані

Вихідними даними підпрограми є перелік змінених файлів для кожного комплексу програм, визначеного в файлі опису packlist.xml.

3.2.5.3 Опис підпрограми TC_ComparePackages()

Функціональне призначення

Підпрограма призначена для визначення переліку файлів комплекту ПЗ, створеного програмою підготовки, в який були внесені зміни в порівнянні з відповідним проінстальованим(еталонним) комплектом, розташованим в каталозі ETALONS групового проекту.

Опис алгоритму

Алгоритм представляє собою наступну послідовність кроків.

Крок 1

Створення файлу контрольних сум файлів комплекту ПЗ, створеного програмою підготовки.

Крок 2

Створення файлу контрольних сум файлів проінсталювати (еталонного) комплекту ПЗ.

Крок 3

Порівняння файлів контрольних сум комплекту ПЗ, створеного програмою підготовки та проінстальованого(еталонного) комплекту.

Вхідні дані

Вхідними даними підпрограми є:

- номер комплексу програм;
- тип комплекту ПЗ.

Вихідні дані

Вихідними даними підпрограми є перелік змінених файлів комплекту ПЗ.

Висновки до розділу 3

У третьому розділі виконані:

- складання й написання алгоритмів програм, що входять до комплексу, які забезпечують формування й інсталяції комплектів програмних компонентів програмно-технічного комплексу на мові технологічного програмування, в ПЕОМ, у середовищі Windows;
- складання й написання програм за вищевказаними алгоритмами, які входять до комплексу, що забезпечують формування й інсталяції комплектів програмних компонентів програмно-технічного комплексу на мові технологічного програмування, в ПЕОМ, у середовищі Windows.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯЇХ

У даному розділі проаналізовані потенційно небезпечні й шкідливі виробничі факторів. На підставі цього аналізу розроблені заходи з техніки безпеки, а також заходи, що забезпечують виробничу санітарію та гігієну праці, а також рекомендації із пожежної безпеки.

Так як завданням дипломного проекту є розробка програмних засобів формування комплектів програмних компонентів програмно-технічного комплексу, то аналіз потенційно небезпечних і шкідливих виробничих чинників виконується для персонального комп'ютера на якому знаходитимуться інструментальні засоби програмування.

4.1 Аналіз потенційно небезпечних і шкідливих виробничих факторів, що впливають на персонал

Обчислювальна техніка при функціонуванні має наступні експлуатаційні характеристики:

- робоче живлення 220 В;
- частота живильної мережі 50 Гц;
- споживана потужність в межах 300 Вт.

При роботі на персональних ПЕОМ користувач наражається на небезпеку ураження електричним струмом. Приміщення для обчислювальної техніки за ступенем небезпеки ураження людини електричним струмом відноситься до приміщень без підвищеної небезпеки [18]. Тяжкість роботи персоналу, що обслуговує і працює на ПЕОМ, відноситься до категорії 1а - легкі фізичні навантаження [19]. При обслуговуванні обчислювальної техніки мають місце фізичні і психофізіологічні небезпечні та шкідливі виробничі фактори [20]:

- підвищене значення напруги в електричному ланцюзі, замикання якого може відбутися через тіло людини;
- підвищена або знижена температура повітря робочої зони;

- підвищена або знижена рухливість повітря;
- підвищена або знижена вологість;
- підвищений рівень електромагнітних полів у робочій зоні;
- відсутність або нестача природного світла;
- підвищена пульсація світлового потоку;
- розумове перенапруження;
- монотонність праці;
- емоційні навантаження;
- підвищений рівень шуму;
- недостатнє освітлення робочого місця;
- підвищена статична електроенергія.

4.2 Заходи з техніки безпеки

Для захисту людей від ураження електричним струмом при дотику до металевих неструмоведучих частин, які можуть опинитися під напругою в результаті пошкодження ізоляції, передбачаються наступні заходи:

- захисне заземлення або занулення металевих частин електроустановок, які доступні для дотику людини й не мають інших видів захисту, що забезпечують електробезпеку [21]:
- захисне відключення;
- електричний поділ мереж;
- використання малої напруги;
- ізоляція струмоведучих частин;
- огорожу електроустановок;
- шина заземлення виконується провідником з опором не більше 4-х Ом.

4.3 Заходи, що забезпечують виробничу санітарію та гігієну праці

У виробничому приміщенні на організм людини nf його працездатність впливають мікрокліматичні фактори. Мікроклімат виробничих приміщень визначається поєднанням температури, вологості q швидкості руху повітря, а також температури навколишніх поверхонь.

Для робіт категорії 1а для робочої зони виробничих приміщень забезпечуються наступні метеорологічні умови:

− в холодний і перехідний період року температура повітря 22 ÷ 24°С;

− відносна вологість повітря 40 ÷ 60%, швидкість руху повітря не більше 0.1 м / с;

− у теплий період року температура повітря 23 ÷ 25°С, відносна вологість повітря 40-60%, швидкість руху повітря не більше 0.1 м/с.

Для підтримки оптимальних мікрокліматичних умов, при відсутності надлишкового тепла, вологи, шкідливих речовин досить природної вентиляції, при наявності надлишкового тепла і для припливу зовнішнього повітря використовується кондиціонер типу БК-1500.

Рівень шуму не перевищує санітарних норм. Тому застосування захисту від шуму в роботі не передбачається.

Ще одна проблема полягає в тому, що спектр випромінювання комп'ютерного монітора включає в себе рентгенівську, ультрафіолетову та інфрачервону області, а також широкий діапазон хвиль інших частот. Небезпека рентгенівських променів мізерно мала, оскільки цей вид випромінювання поглинається речовиною екрану. Однак велику увагу слід приділяти біологічним ефектам низькочастотних електромагнітних полів.

Проектом передбачається час роботи персоналу з ПЕОМ не більше 3 годин. Відстань від дисплея до людини - 1,2 метра.

Для зниження стомлюваності обслуговуючого персоналу в приміщеннях, де розташовані обчислювальні засоби, передбачається використовувати кольорові поєднання й покриття, що не дають відблисків.

У проекті, що розробляється передбачається використовувати сполучне освітлення. У світлий час доби приміщення буде висвітлюватися через віконні прорізи, в решту часу буде використовуватися штучне освітлення. Штучне освітлення створюється лампами розжарювання або газорозрядними лампами. Штучне освітлення в робочому приміщенні передбачається здійснювати з використанням люмінесцентних джерел світла в світильниках загального освітлення, оскільки люмінесцентні лампи мають високу світловою віддачею (до 75 лм/Вт і більше), тривалим терміном служби (до 10000 годин), спектральним складом випромінюваного світла, близьким до сонячного. При експлуатації комп'ютерів виробляється зорова робота IV в розряду точності (середня точність). При цьому нормована освітленість на робочому місці дорівнює 200 лк. Джерелом природного світла (освітлення) є сонячне світло. У приміщенні, де розташовані комп'ютери, передбачається природне бічне освітлення, рівень якого відповідає ДБН В.2.5-28-2015 [22] "Будівельні норми і правила".

Регулярно проводиться контроль освітленості, який підтверджує, що рівень освітленості задовольняє ДБН і для даного приміщення в світлий час доби достатньо природного освітлення. Для організації освітлення в темний час доби передбачається обладнати приміщення, довжина якого становить 10 м, ширина 8 м, висота 4 м, світильниками ЛП02П, облаштованими лампами типу ЛБ (дві по 80 Вт) зі світловим потоком 5400 лм кожна.

Розрахунок штучного освітлення проводиться за коефіцієнтами використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні. Розрахунок кількості світильників здійснюється за формулою:

$$N = E \cdot S \cdot Z \cdot K / (F \cdot U \cdot M) \qquad (4.1)$$

де N - число світильників;

E - нормоване освітлення;

S - площа підлоги, $м^2$;

Z - поправний коефіцієнт світильника (для стандартних світильників Z=1.1÷1.3);

K - коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації;

U - коефіцієнт використання, що залежить від типу світильника, показника індексу приміщення і т. п. (U= 0.55 ÷ 0.6);

M - число люмінесцентних ламп у світильнику;

F - світловий потік.

Згідно вимог ДБН В.2.5-28-2015, освітлення робочого місця оператора обчислювальної техніки повинно бути не менше 200 лк.

$$N = 200 \cdot 80 \cdot 1.1 \cdot 1.5 / (5400 \cdot 0.6 \cdot 2) = 4.07 \approx 4$$

Обираємо кількість світильників, що дорівнює 4.

Схема розташування світильників показана на рис. 4.1.

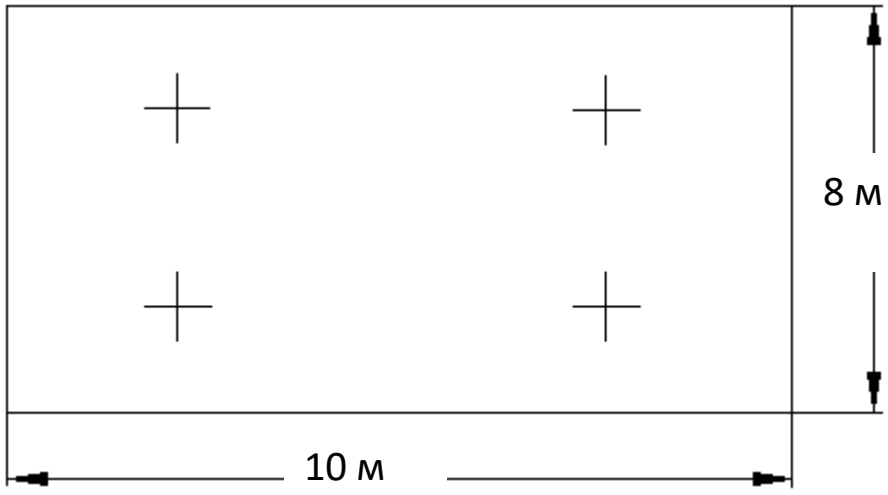


Рисунок 4.1 - Схема розташування світильників

Для захисту людини від електромагнітних випромінювань передбачається використовувати захисні екрани, регламентований час роботи не більше 4 год, відстань від екрану монітора не менше 1,2 м.

4.4 Рекомендації із пожежної профілактики

Пожежі в робочому приміщенні становлять небезпеку, тому що пов'язані як з матеріальними втратами, так і з відмовою засобів обчислювальної техніки.

Пожежа може виникати при внесенні джерела запалювання в горючу середу. Горючими матеріалами в приміщенні, де розташовані обчислювальні засоби є будівельні матеріали, віконні рами, двері, підлоги, меблі, ізоляція силових і сигнальних кабелів, радіотехнічні деталі, конструктивні елементи з пластичних матеріалів, рідини для очищення елементів і вузлів ПЕОМ від забруднень.

- поліамід - матеріал корпусу мікросхеми, горюча речовина, температура самозаймання 420°С;

- полівінілхлорид - ізоляційний матеріал, горюча речовина, температура займання 335°С, температура самозаймання 530°С, теплота згоряння 18000-20700 кДж/кг;

- склотекстоліт ДЦ - матеріал друкованих плат, важко-горючий матеріал, показник горючесті1.74, не схильний до температурного самозаймання; пластик кабельний №489 - матеріал ізоляції кабелю, горючий матеріал, показник горючості більше 2.1;

- деревина - будівельний і оздоблювальний матеріал, матеріал з якого виготовлені меблі, горючий матеріал, показник горючості більше 2.1, теплота згоряння 18731-20853 кДж/кг, температура займання 399°С, схильна до самозаймання [23].

Згідно НАПБ Б.03.002-2007 таке приміщення належить до категорії "В" (пожежонебезпечна), як пожежонебезпечна. Відповідно до класифікації за ПБЕ приміщення відноситься до класу П-IIа. Пожежа може виникнути в результаті утворення джерела запалювання (іскри й дуги короткого замикання, порушення ізоляції, що приводить до короткого замикання, перегріву радіодеталей внаслідок тривалого перевантаження) та внесення його до горючої середи. При повному згорянні органічних сполук утворюється CO_2, SO_2, H_2O, N_2, а при згорянні

неорганічних сполук - оксиди. Залежно від температури плавлення продукції реакції диму можуть або знаходитися у воді розплаву (Al_2O_3, TiO_2), або підійматися в повітря у вигляді диму (P_2O_5, Na_2O, MgO). Розплавлені тверді частинки створюють світимість полум'я. Склад продуктів неповного згоряння горючих речовин складний і різноманітний. Це можуть бути горючі речовини - H_2, CO, CH_4 та ін.; атомарний водень і кисень; різні радикали - OH, CH і інші. Продуктами неповного згорання можуть бути також оксиди азоту, спирти альдегіди, кетони й високотоксичні з'єднання, наприклад, синильна кислота.

Для захисту персоналу від можливого отруєння проектом передбачається застосування протигаза з коробкою марки «В» (жовта).

Пожежна безпека при застосуванні ЕОС забезпечується:

- системою запобігання пожеж;
- системою протипожежного захисту;
- організаційно-технічними заходами.

До системи запобігання пожежі відносяться:

- запобігання утворенню горючого середовища;
- електроживлення ПЕОМ має автоматичне блокування відключення електроенергії на випадок зупинки системи охолодження й кондиціонування;
- система вентиляції обчислювальних центрів обладнуються блокувальними пристроями, що забезпечують її відключення в разі пожежі. Система обладнується вогнеперепинюваючими клапанами;
- освіту в займистою середовищі джерел запалювання;
- застосування обладнання задовольняє вимогам;
- забезпечення пожежної безпеки обладнання;
- після закінчення роботи, перед закриттям приміщення, все електроустановки відключаються і ПК від мережі електроживлення.

Зменшити горюче навантаження не є можливим, тому проектом передбачається застосування наступних способів і їх комбінацій для запобігання утворення (внесення) джерел запалювання:

- застосування обладнання задовольняє вимогам електростатичної безпеки;

- застосування в конструкції швидко діючих засобів захисного відключення можливих джерел запалювання;

- виключення можливості появи іскрового заряду статичної електрики в займистому середовищі з енергією, що дорівнює і вище мінімальної енергії запалювання;

- підтримання температури нагріву поверхні машин механізмів устаткування пристроїв речовин і матеріалів, які можуть увійти в контакт з горючим середовищем, нижче гранично допустимої, що становить 80% найменшої температури самозаймання пального.

Для зниження пожежної небезпеки проектом передбачається використовувати систему автоматичної пожежної сигналізації з одним димовим датчиком- оповіщувачем типу ВДМ-1М, який розрахований для контролю площі до 100 $м^2$ при висоті стелі до чотирьох метрів, а також первинні засоби пожежогасіння. В якості первинних засобів пожежогасіння передбачається використовувати:

- ручний вуглекислотний вогнегасник ОУ-5 - 1 шт .;
- повітряно-пінний вогнегасник ОВП-5 - 1 шт .;
- азбестове полотно 1.5 × 2 м.

В якості організаційно-технічних заходів рекомендується проводити навчання робочого персоналу пожежної безпеки.

Висновки до розділу 4

У четвертому розділі виконані:

- аналіз потенційно небезпечних і шкідливих виробничих факторів при роботі з ПЕОМ;
- розробки заходів із техніки безпеки й заходи, що забезпечують виробничу санітарію та гігієну праці;
- аналіз штучного освітлення;
- розрахунки й розробки рекомендацій щодо пожежної безпеки.

ВИСНОВКИ

У даному дипломному проекті розроблені інструментальні засоби формування комплектів, інсталяції комплектів програмних компонентів програмно– технічних комплексів, а також забезпечено взаємодію з розробленим на ПрАТ «СНВО «Імпульс» програмним забезпеченням, яке встановлюється на мікропроцесорних субкомплексах контролю й управління.

В поданих розділах здійснений аналіз об’єкту дослідження, наведена його теоретична та практична значимість. Також представлена й детально описана логічна структура інструментальних засобів технологічного програмування. Й, нарешті, представлені програмні рішення разом із алгоритмічною реалізацією інструментальних засобів програмування.

В розділі «Охорона праці та безпека в надзвичайних ситуаціяїх» виконаний аналіз потенційно небезпечних і шкідливих виробничих факторів при роботі з ПЕОМ, розроблені заходи з техніки безпеки, заходи, що забезпечують виробничу санітарію та гігієну праці, виконаний аналіз штучного освітлення, розроблені рекомендації та розрахунки щодо пожежної безпеки.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. «Програмно-технічні комплекси АСК ТП: Навч. Посібник-К» - Єлісєєв В.В., Ларгін В.А., Пивоваров Г.Ю. Видавничо-поліграфічний центр «Київський університет», 2003. - 429 с.

2. «Системи контролю та управління технологічними процесами: Збірник наукових статей» Єлісєєв В. В. - Луганськ: Світлиця, 2006. - 440 с.

3. 0229767.40402-01 13 01 Комплекс стартових програм МСКУ-4. Архів компонент поставки і компонент супроводу. Опис програм.

4. 0229767.40400-03 34 01 Комплекс програм технологічного програмування «ЯРУС 4.0». Інструментальні засоби підготовки цільових програм, що управляють і налаштувань ОБД. Керівництво оператора.

5. 0229767.40402-01 13 01 Комплекс стартових програм МСКУ-4. Архів компонент поставки і компонент супроводу. Опис програм.

6. 0229767.40400-03 31 01 Комплекс програм технологічного програмування «ЯРУС 4.0». Опис застосування.

7. 0229767.40400-03 13 04 Комплекс програм технологічного програмування «ЯРУС 4.0». Архів компонент поставки і компонент супроводу. інструментальні програми.

8. 0229767.40400-03 35 01 Комплекс програм технологічного програмування «ЯРУС 4.0». Опис конфігурації технічних і програмних засобів. Опис мови.

9. 0229767.40400-03 35 02 Комплекс програм технологічного програмування «ЯРУС 4.0». Текстова мова програмування прикладних програм. Опис мови.

10. 0229767.40400-03 30 01 Комплекс програм технологічного програмування «ЯРУС 4.0». Формуляр.

11. 0229767.40400-03 95 01 Комплекс програм технологічного програмування «ЯРУС 4.0». Архів компонент поставки і компонент супроводу. Інструкція по використанню.

12. 0229767.40400-03 13 05 Комплекс програм технологічного програмування «ЯРУС 4.0». Архів компонент поставки і компонент супроводу. Сервісні засоби налагодження. Опис програм.

13. 0229767.40400-03 34 02 Комплекс програм технологічного програмування «ЯРУС 4.0». Сервісні засоби налагодження. Керівництво оператора.

14. 0229767.00446 01 30 01 Системне ПЗ мережі МАПС/Ethernet для ОС Windows. Формуляр.

15. http://www.imp.lg.ua

16. RPM Package Manager http://rufus.w3.org/linux/RPM

17. Методичні вказівки до виконання і захисту дипломного проекту (роботи) бакалавра спеціальностей 122 "Комп'ютерні науки та інформаційні технології", 123 "Комп'ютерна інженерія", 125 "Кібербезпека" (за напрямами 6.050101 "Комп’ютерні науки", 6.050102 "Комп’ютерна інженерія") для здобувачів вищої освіти денної і заочної форм навчання / Уклад.: Скарга-Бандурова І.С., Барбарук В.М., Кардашук В.С. – Сєвєродонецьк: СНУ ім. В. Даля, 2018. – 60 с.

18. ДСТУ Б А.3.2-13: 2011. Будівництво. Електробезпека. Загальні вимоги.

19. ДСТУ 12.1.005-88 ССБП. Загальні санітарно-гігієнічні вимоги до повітря робочої зони.

20. ДСТУ 12.0.003-74 ССБП. Небезпечні і шкідливі виробничі фактори. Класифікація.

21. ДСТУ 12.1.009-76 ССБП. Електробезпека. терміни та визначення.

22. ДБН В.2.5-28-2015 "Будівельні норми і правила".

23. ДСТУ 12.1.004-91 ССБП. Пожежна безпека. Загальні вимоги.

24. Пожежовибухонебезпека речовин і матеріалів, і засоби їх гасіння. Довідник, під ред. Баратова А.Н. в 2-х томах М .: Хімія, 1990.

ДОДАТОК А
ПРОГРАМНА РЕАЛІЗАЦІЯ

Підпрограма main()

```
#include <qapplication.h>
#include <qdir.h>
#include <qtranslator.h>
#include <qtextcodec.h>
#include <qmessagebox.h>
#include "pack2xml.h"
#include "struct.h"
#ifdef linux
#include <stdlib.h>
#endif
QTextCodec *my_codec_dos, *my_codec;
QTextCodec *my_codec_koi8;
Pack2xml *w2;
QApplication *app=NULL;
int version=DEFAULT_VERSION;
int main( int argc, char ** argv )
{#ifdef WIN32
      char *ptr;
      char ss[1024];
#endif
      my_codec=QTextCodec::codecForName("CP1251");
      my_codec_dos=QTextCodec::codecForName("IBM 866");
      my_codec_koi8=QTextCodec::codecForName("KOI8-R");

      QTranslator translator(0);
      QApplication a( argc, argv );
      app=&a;
      QString grdir="";
#ifdef WIN32
      strcpy(ss,argv[0]);
      for(ptr = ss; *ptr; ptr++) { if (*ptr == '\\') *ptr = '/'; };
      if((ptr=strrchr(ss,'/'))!=0)
      {
       *(ptr+1)=0;
      translator.load( "pack2xml", ss );
      }
      else translator.load( "pack2xml", "." );
#else
      translator.load( "pack2xml",
QString("/usr/share/%1/icp").arg(getenv(MSKU)));
#endif
      a.installTranslator( &translator );
      Pack2xml w;
      w2=&w;
      w.show();
      if (argc>1)
      {
            grdir.sprintf("%s",argv[1]);
            w.GrpDirSet(grdir);
      }
      a.connect( &a, SIGNAL( lastWindowClosed() ), &a, SLOT( quit() ) );
      return a.exec();
}
```

Набір підпрограм cparams

```
void cparams::init()
{
      docarch->setColumns(1);
      docpost->setColumns(1);
```

```
5         otherfiles->setColumns(1);
6         otherbins->setColumns(1);
7         modlog->setColumns(1);
8         tarch = new listtable(docarch);
9         tpost = new listtable(docpost);
10        tother = new listtable(otherfiles);
11        tbins = new listtable(otherbins);
12        tbull = new listtable(modlog);
13  }
14  void cparams::pushButtonOk_clicked()
15  {
16        accept();
17  }
18  void cparams::pushButtonCancel_clicked()
19  {
20        reject();
21  }
22  int cparams::TablesInit( QString & ishdir, QString & packnum, const QString &
23  packdir, QString & doc_post, QString & doc_arch, QString & bull_dir)
24  {
25        QString ss_a, ss_p, ss_b;
26        int r;
27        ss_p=doc_post;
28        ss_p.replace(QChar('&'), packnum.ascii());
29        r=ss_p.findRev("/");
30        if(r>0)
31              ss_p.truncate(r);
32        ss_a=doc_arch;
33        ss_a.replace(QChar('&'), packnum.ascii());
34        r=ss_a.findRev("/");
35        if(r>0)
36              ss_a.truncate(r);
37          ss_b = bull_dir;
38        ss_b.replace(QChar('&'), packnum);
39        char *s1=(char *)ss_b.ascii();
40        // ...
41        if(ishdir.isEmpty()) return -1;
42        char*st;
43  //    tarch->listdir =
44  QString("%1%2DOCUMENTS%3Pack_%4%5").arg(ishdir).arg(QDir::separator()).arg(QDir::s
45  eparator()).arg(packnum).arg(QDir::separator());
46        tarch->listdir =
47  QString("%1%2%3%4").arg(ishdir).arg(QDir::separator()).arg(ss_a).arg(QDir::separat
48  or());
49        st=(char*)tarch->listdir.ascii();
50        tpost->listdir =
51  QString("%1%2%3%4").arg(ishdir).arg(QDir::separator()).arg(ss_p).arg(QDir::separat
52  or());
53        st=(char*)tpost->listdir.ascii();
54        tother->listdir = ishdir+QDir::separator();
55        st=(char*)tother->listdir.ascii();
56        tbins->listdir = ishdir+QDir::separator();
57        st=(char*)tbins->listdir.ascii();
58        tbull->listdir =
59  QString("%1%2%3%4").arg(ishdir).arg(QDir::separator()).arg(ss_b).arg(QDir::separat
60  or());
61        st=(char*)tbull->listdir.ascii();
62        tarch->subdir = (packdir.isNull() ? tarch->listdir : packdir);
63        tpost->subdir = (packdir.isNull() ? tpost->listdir : packdir);
64        tother->subdir= (packdir.isNull() ? tother->listdir: packdir);
65        tbins->subdir = (packdir.isNull() ? tbins->listdir : packdir);
66        tbull->subdir = (packdir.isNull() ? tbull->listdir : packdir);
67        st=(char*)tpost->subdir.ascii();
```

```
68          st=(char*)tbull->subdir.ascii();
69          return 0;
70    }
```

Підпрограма parse

```
1     #include "parse.h"
2     extern QTextCodec *my_codec_dos, *my_codec;
3     int parser::open(int m)
4     {QString sline="";
5      if (file.exists())
6      {
7         if (!file.open(IO_ReadOnly))
8        return -1;
9       listLine.clear();
10      while(!ts.atEnd())
11      {sline = ts.readLine();
12       listLine.append(sline);
13      }
14      file.close();
15     }
16    if (!file.open(m))
17      return -1;
18     return 0;
19    int parser::open()
20    {
21    if (!file.open(IO_ReadOnly))
22      return -1;
23     return 0;
24    }
25    int parser::ReadFieldSec(QString section, struct j * k, int n)
26    {
27     QString l="", s1, s2;int f=0;
28    if (k == NULL)
29      return -1;
30     file.reset();
31    ts.setCodec(QTextCodec::codecForName("IBM 866"));
32     while(!ts.atEnd())
33     {l = my_codec->fromUnicode(ts.readLine());
34      if (l == ("["+section+"]") )
35       f=1;
36      else
37       if (f)
38       {if ( (l.find('[') == 0))
39         break;
40        s1 = l.section('=', 0,0);
41        s2 = l.section('=', 1,-1);
42          for (int i=0;i<n;i++)
43          {
44           if (QString(k[i].t) == s1)
45           {memset(k[i].r, 0, k[i].s);
46            if (!s2.isEmpty())
47             strcpy( k[i].r, (char*)s2.ascii());
48            else
49            strcpy(k[i].r, "");
50          break;
51           }
52          }
53       }
54     }
55     return 0;
56    }
```

```
int parser::ReadStrSec(QString section, QStringList&list)
{
 QString l, s1, s2;int f=0;
 file.reset();list.clear();
 ts.setCodec(QTextCodec::codecForName("IBM 866"));
 while(!ts.atEnd())
 {l = my_codec->fromUnicode(ts.readLine());
  if (l == ("["+section+"]") )
   f=1;
  else
   if (f){
      if ( (l.find('[') == 0))
     break;
    if (!l.isEmpty())
       list.append(l);
   }
 }
 return 0;
}
void parser::wrSec(QString&sec, QString&var, QString&val)
{QString s="", v="", sval="";
 int fsec=0, fvar=0;
 char *j = (char*)val.ascii();
 sval=var+"="+val;
 for ( QStringList::Iterator it = listLine.begin(); it != listLine.end(); ++it )
 {s=*it;
  char* k = (char*)s.ascii();
  if (s == ("["+sec+"]") )
   fsec++;
  else
   if (fsec==1)
   {
       v=s.section("=", 0,0);
     char *o =(char*)v.ascii();
       if (v.find('[')==0)
       {
        if (!val.isEmpty())
        {listLine.insert(it, sval);fvar=1;}
        break;
       }
       if (v==var)
       {
        if (val.isEmpty())
 for ( QStringList::Iterator its = listLine.begin(); its != listLine.end(); ++its
)
 {s=*its;
  char* k = (char*)s.ascii();
  ts<<s<<"\n";
 } */
}
void parser::wrSec(QString&sec, QStringList list)
{QString s="", v="", sval="";
 int fsec=0, fvar=0;
 QStringList::Iterator it;
 it = listLine.begin();
 while(it != listLine.end())
 {s=*it;
  char* k = (char*)s.ascii();
  if (s == ("["+sec+"]") )
   fsec++;
  else
  {
   if (fsec==1)
```

```
120      {
121         if (s.find('[')==0)
122       {
123        for ( QStringList::Iterator its = list.begin(); its != list.end(); ++its )
124         listLine.insert(it, *its);
125          fvar=1;
126          break;
127         }
128       else
129         {
130        it=listLine.remove(it);
131          if (it == listLine.end())
132          break;
133          continue;
134         }
135      }
136     }
137     it++;
138    }
139   if (!fsec)
140    {listLine.append(QString("[%1]").arg(sec));
141     for ( QStringList::Iterator its = list.begin(); its != list.end(); ++its )
142      listLine.append(*its);
143    }
144    //есть секция но небыло полей секции
145    if (fsec && !fvar)
146    {
147     for ( QStringList::Iterator its = list.begin(); its != list.end(); ++its )
148      listLine.append(*its);
149    }
150   }
151
152   void parser::flush()
153   {QString s="";
154    file.reset();
155    for ( QStringList::Iterator its = listLine.begin(); its != listLine.end(); ++its
156   )
157    {s=*its;
158     ts<<s<<"\n";
159    }
160   }
161   void parser::close()
162   {listLine.clear();
163    file.close();
164   }
```

Підпрограма ErrMessages

```
1   #include "ErrMessages.h"
2   unsigned long ErrMsgCount = sizeof(ErrMsg)/sizeof(TC_Strings);
3   void TC_PrintMessage( unsigned int MessageCode, unsigned int MessageKind, ... )
4   {
5         int _MessageKind = MessageKind;
6         if ( _MessageKind > TC_MAX_MESSAGE_KIND ) _MessageKind = TC_INFORMATION;
7         if ( MessageCode < ErrMsgCount )
8         {
9               char s[TC_MAX_LENSTR];
10              va_list va;
11              memset( &s[0], 0, sizeof( s ) );
12              char *format = (char *)&ErrMsg[MessageCode].String[0];
13
14              va_start(va, MessageKind );
15              vsprintf(&s[0], format, va);
```

```
16              va_end(va);
17              QTextCodec *QTCodec1 = QTextCodec::codecForName( "CP1251" );
18              QString t = QTCodec1->toUnicode( QString( s ) );
19              QTextCodec *QTCodec = QTextCodec::codecForName("IBM 866");
20              t = QTCodec->fromUnicode( t );
21              fprintf( stdout, t.data() ); fflush( stdout );
22              if ( TC_ERROR == MessageKind )
23              {
24                    exit( 1 );
25              }
26        }
27  }
```

Програма PackCreator

```
 1  #include "PackCreator.h"
 2  #include "ErrMessages.h"
 3  #include <qtextcodec.h>
 4
 5  #ifdef WIN32
 6  #include <windows.h>
 7  #else
 8  #include <time.h>
 9  #endif
10  #include <stdlib.h>
11  QString TC_QStrToUnicode(const QString &qstr)
12  {
13       QTextCodec *QTCodec = QTextCodec::codecForName("CP1251");
14       return QTCodec->toUnicode(qstr);
15  }//QStrToUnicode
16  void TC_Wait(uint msec)
17  {
18  #ifdef WIN32
19        Sleep(msec);
20  #else
21        timespec req;
22        req.tv_sec = msec / 1000;
23        req.tv_nsec = (msec % 1000)*1000000;
24        nanosleep(&req, NULL);
25  #endif
26  }
27  QString TC_GetEnv_P20400()
28  {
29        QString environment;
30        environment = getenv( ENVVARNAME );
31        if( environment.isEmpty() )
32        {
33              TC_PrintMessage( TC_ErrGetEnv_P20400, TC_ERROR, "e", ENVVARNAME );
34        }
35        return environment;
36  }
37  #ifndef _SUPPLY_CREATOR_H_
38  #define _SUPPLY_CREATOR_H_
39  #include <qstringlist.h>
40  #include <qthread.h>
41  #include <qmap.h>
42  typedef QMap< QString, QStringList > TMapFileFromListFilesLst;
43  typedef enum _e_stateWorking { old_mode = 0, new_mode };
44  // Создание поставки
45  void  TC_SupplyCreator();
46  // Поток для создания
47  class TSupplyCreator : public QThread
48  {
```

```
49  public:
50        virtual void run();
51        bool haveError;
52
53  protected:
54  private:
55  };
56  // Работа с проектами
57  void  TC_WorkWithPrj(         const QString &namePrj,
58                                      const QString &pathPackDir,
59                                      const QString &packageName );
60  // Работа с документами
61  void  TC_WorkWithSupplyDoc(   const QString &nameDoc,
62                                            const QString &pathToSourceDoc,
63                                            const QString &Package,
64                                            const QString &pathPackDocDir );
65  // Работа с BIN файлами
66  void TC_WorkWithOtherBin(     const QStringList &lstBinFile,
67                                      const QString &pathPackDir,
68                                      const QString &packageName,
69                                      const QString &pathToSource );
70  // Создание поставки не для проекта ОБД/СДА
71  void  TC_CreatePatternSupplyKIP(    const QString &filePrj,
72                                                  const QString &namePrj,
73                                                  const QString &pathPackDir,
74                                                  const QString &packageName );
75  //    Извлечь ZIP в указанную папку
76  int         TC_ExtractZIP( const QString &nameZIPFile, const QString
77  &pathToExtract );
78  //    Получение списка файлов директории
79  int         TC_GetListFileDir(      const QString &dirIn,
80                                      QStringList &lstFile );
81  //    Сохранение списка файлов в файл chk_file
82  int         TC_SaveListFileDir(     const QStringList &lstFile,
83                                      const QString &nameFile  );
84  //    Запаковать содержимое временного каталога в файл <имя проекта>.ZIP
85  int         TC_AddDirectoryToZIP(   const QString &nameZIPFile,
86                                            const QString &pathArchDir,
87                                            const QString &CreateDir="",
88                                            const QString &PathToAdd="" );
89  //    Формирование контрольных сумм файлов
90  int         TC_RunCheck(      const QString &tmpDir,
91                                const QString &nameFile=TC_FILENAME_CHK_SUM );
92  //    Получение каталога с исходниками
93  QString     TC_GetPathToSource();
94  //    Создает сервисный файл install.bat в каталоге Windows
95  int         TC_CreateInstallForWindows( const QString &pathPackDirWindows );
96  //    Создает сервисный файл install в каталоге Linux
97  int         TC_CreateInstallForLinux( const QString &pathPackDirLinux );
98  //    Создает сервисный файл verify.bat в каталоге Windows
99  int         TC_CreateVerifyForWindows( const QString &pathPackDirWindows );
100 //    Создает сервисный файл verify в каталоге Linux
101 int         TC_CreateVerifyForLinux( const QString &pathPackDirLinux );
102 //    Копирование приложения "check"
103 int         TC_CopyCheck( const QString &pathPackDirWindows, const QString
104 &pathPackDirLinux );
105 //    Копирование приложения bat для КИП
106 int         TC_CopyBAT( const QString &pathPackDirWindows, const QString
107 &pathPackDirLinux, const QString &packageName );
108 //    Создает сервисный файл winstdir.bat
109 int         TC_CreateWinstdirForWindows( const QString &pathPackDirWindows, const
110 QString &packageName );
111 //    Создает сервисный файл linstdir
```

```
int         TC_CreateLinstdirForLinux( const QString &pathPackDirLinux, const
QString &packageName );
//    получить имя BIN файла
QStringList TC_GetNameBINFile( const QString &namePRJ );
//    Замена в пути к файлу пути к искходникам
bool TC_ReplacePathToSource(
                                        const QString &pathFile,
                                        const QString &pathSource,
                                        const QString &strReplace,
                                        const QString &prjName,
                                        QString &new_pathFile );
// Если хотя бы в одном проекте находится файл PackFiles[n].lst, то работаем по
новому
_e_stateWorking GetStateWorking( const QString Package, const QStringList _prjList
);
// Чтение всех файлов по маске PackFiles*.lst, и занесение информации о файлах в
структуру TC_Supply
bool GetFileSupply( const QString &Package, const QStringList &_prjList, TC_Supply
&FileSupply);
```

Програма SupplyCreator

```
#include <qstringlist.h>
#include <qthread.h>
#include <qmap.h>
typedef QMap< QString, QStringList > TMapFileFromListFilesLst;
typedef enum _e_stateWorking { old_mode = 0, new_mode };
void  TC_SupplyCreator();
class TSupplyCreator : public QThread
{
public:
      virtual void run();
      bool haveError;

protected:
private:
};
void  TC_WorkWithPrj(        const QString &namePrj,
                                        const QString &pathPackDir,
                                        const QString &packageName );
void  TC_WorkWithSupplyDoc(  const QString &nameDoc,
                                              const QString &pathToSourceDoc,
                                              const QString &Package,
                                              const QString &pathPackDocDir );
void TC_WorkWithOtherBin(    const QStringList &lstBinFile,
                                        const QString &pathPackDir,
                                        const QString &packageName,
                                        const QString &pathToSource );

void  TC_CreatePatternSupplyKIP(   const QString &filePrj,
                                                    const QString &namePrj,
                                                    const QString &pathPackDir,
                                                    const QString &packageName );
int         TC_ExtractZIP( const QString &nameZIPFile, const QString
&pathToExtract );
int         TC_GetListFileDir(     const QString &dirIn,
                                        QStringList &lstFile );

int         TC_SaveListFileDir(    const QStringList &lstFile,
                                        const QString &nameFile  );
int         TC_AddDirectoryToZIP(  const QString &nameZIPFile,
                                              const QString &pathArchDir,
```

```
                                                const QString &CreateDir="",
                                                const QString &PathToAdd="" );
int         TC_RunCheck(      const QString &tmpDir,
                                    const QString &nameFile=TC_FILENAME_CHK_SUM );
QString     TC_GetPathToSource();
int         TC_CreateInstallForWindows( const QString &pathPackDirWindows );
int         TC_CreateInstallForLinux( const QString &pathPackDirLinux );
int         TC_CreateVerifyForWindows( const QString &pathPackDirWindows );
int         TC_CreateVerifyForLinux( const QString &pathPackDirLinux );
int         TC_CopyCheck( const QString &pathPackDirWindows, const QString
&pathPackDirLinux );
int         TC_CopyBAT( const QString &pathPackDirWindows, const QString
&pathPackDirLinux, const QString &packageName );
int         TC_CreateWinstdirForWindows( const QString &pathPackDirWindows, const
QString &packageName );
int         TC_CreateLinstdirForLinux( const QString &pathPackDirLinux, const
QString &packageName );
QStringList TC_GetNameBINFile( const QString &namePRJ );
bool TC_ReplacePathToSource(
                                          const QString &pathFile,
                                          const QString &pathSource,
                                          const QString &strReplace,
                                          const QString &prjName,
                                          QString &new_pathFile );
```

Підпрограма ArchiveCreator

```
#ifndef _ARCHIVECREATOR_H_
#define _ARCHIVECREATOR_H_
#include <qthread.h>
void TC_ArchiveCreator();
// Поток для создания
class TArchiveCreator : public QThread
{
public:
      virtual void run();
      bool haveError;
protected:
private:
};
#endif // _ARCHIVECREATOR_H_
```

Додаток Б

ПРЕЗЕНТАЦІЯ

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

Дипломний проект на тему:

«ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ФОРМУВАННЯ КОМПЛЕКТІВ ТА ІНСТАЛЯЦІЙ КОМПЛЕКТІВ КОМПОНЕНТІВ ПТК»

виконала ст. групи КІ-14ад:
Фурса Поліна
Наук. керівник:
доц. Ларгін В. А.

Сєвєродонецьк 2018

1

ВСТУП

ПрАТ «СНВО «Імпульс» є провідним розробником, виробником і постачальником високонадійних програмно-технічних комплексів (ПТК) для АСК ТП для атомної енергетики.

Автоматизована система керування технологічним процесом(АСК ТП) - автоматизована система у вигляді комплексу програмних і технічних засобів, призначена для вироблення та реалізації керувальної дії на технологічний об'єкт керування згідно з прийнятими критеріями керування.

2

МІКРОПРОЦЕСОРНІ СУБКОМПЛЕКСИ КОНТРОЛЮ Й УПРАВЛІННЯ (МСКУ)

МСКУ використовується для збору, обробки інформації та управління технологічними установками як автономно, так і в складі ПТК. Роботу МСКУ відповідно до визначених користувачем функцій організовує керуюча система МСКУ.

КС МСКУ називається сукупність системних програм, налаштувань даних програм і прикладних програм користувача. КС МСКУ створюється під кожне конкретне застосування МСКУ розробником відповідного ПТК. У КС МСКУ обмін з компонентами ПТК може здійснюватися через мережеві контролери Ethernet по протоколу мережі Ethernet нижнього рівня.

3

АНАЛІЗ ЗАДАЧІ Й ПОСТАНОВКА ТЕХНІЧНОГО ЗАВДАННЯ

Актуальність виконаної роботи полягає в забезпеченні надійності зберігання й поставки, а також зручності у використанні інструментальних засобів, що забезпечують автоматизацію виконання робіт при підготовці, зберіганні та супроводі комплексів програмного забезпечення.

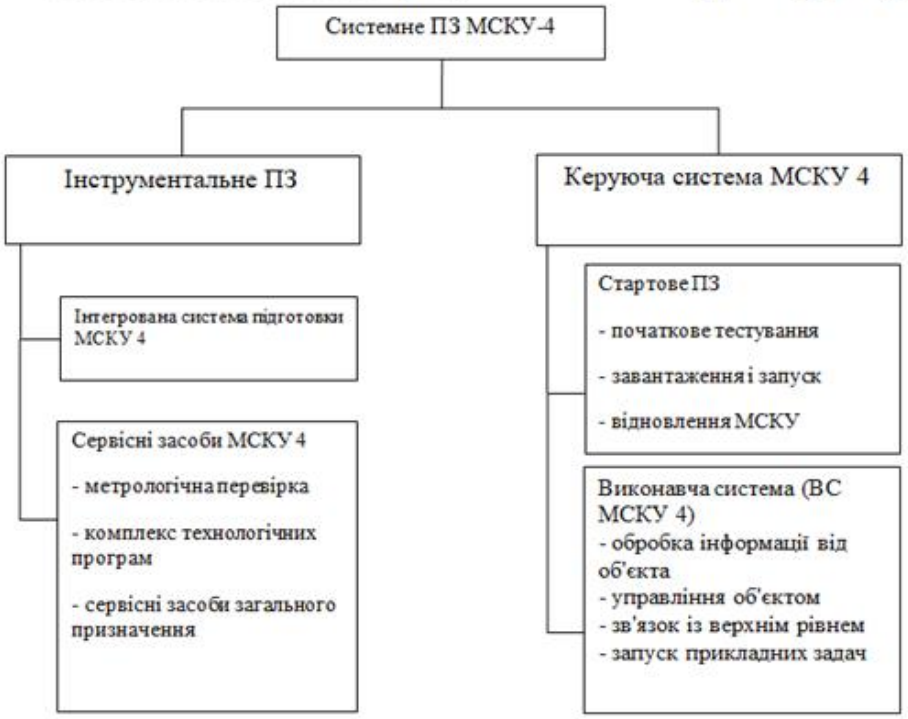
Метою індивідуального завдання є розробка інструментальних засобів формування комплектів, інсталяції комплектів програмних компонентів програмно-технічного комплексу.

В якості інструментальних засобів для розробки підсистеми використовувалися мови програмування **C, C++**.

4

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ МСКУ

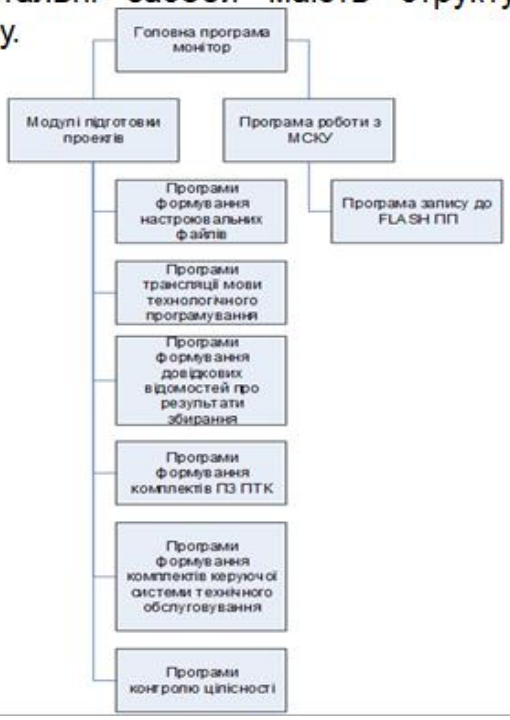
Системне ПЗ МСКУ представлене на поданому рисунку.



5

ОПИС ЛОГІЧНОЇ СТРУКТУРИ ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ

Інструментальні засоби мають структуру, показану на поданому рисунку.



6

ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ФОРМУВАННЯ КОМПЛЕКТІВ, ІНСТАЛЯЦІЇ КОМПЛЕКТІВ ПРОГРАМНИХ КОМПОНЕНТІВ ПТК

Склад інструментальних програм формування комплектів, інсталяції комплектів програмних компонентів ПТК наведений у поданій таблиці.

Ім'я файлу	Виконувані функції
fpo.exe	Програма формування файлу опису збірки комплектів ПЗ компонентів ПТК
finalization.exe	Програма управління процесом створення комплектів під час створення генерації групового проекту
PackCreator.exe	Програма підготовки комплектів ПЗ компонентів ПТК

7

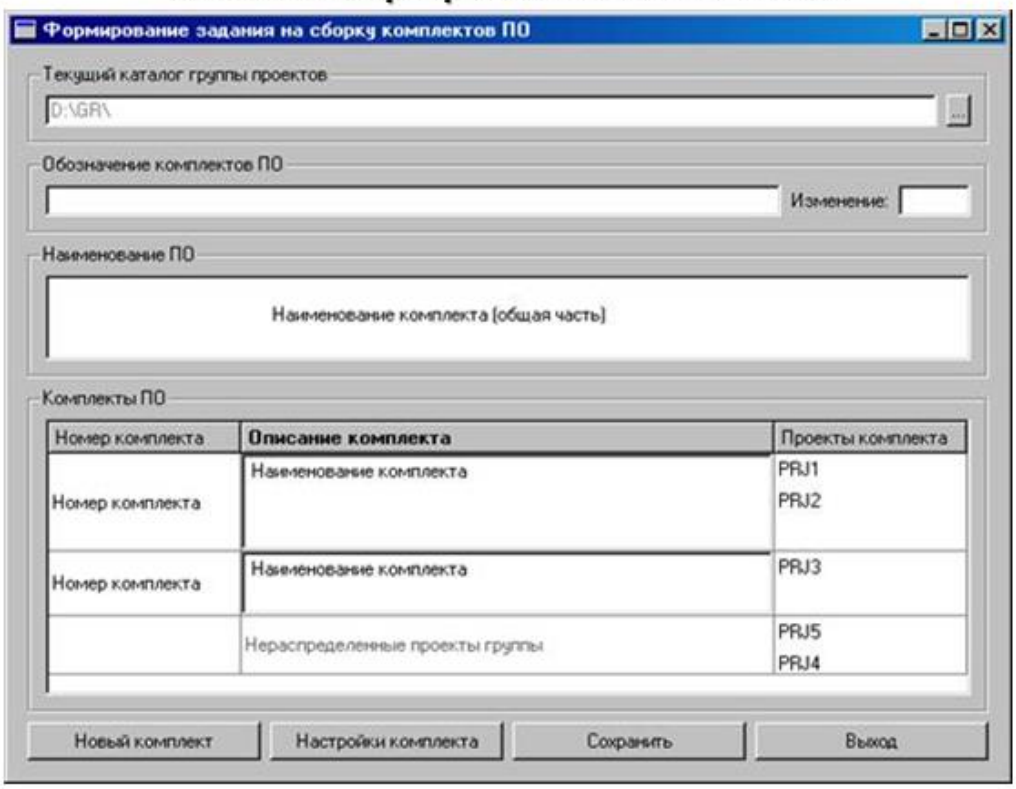
ПРОГРАМА FPO

Програма fpo являє собою набір взаємозв'язаних модулів, перелік яких наведено в поданій таблиці.

Ім'я файлу	Ім'я підпрограми	Виконувані функції
main.cpp	main()	Головна підпрограма
pack2xml.ui.h	init()	Підпрограма ініціалізації основного вікна «Формування завдання на складання комплектів ПЗ»
pack2xml.ui.h	GrpDirSet()	Підпрограма зміни каталогу групового проекту
	NewComplect_clicked()	Підпрограма обробки натискання кнопки «Новий комплект»
	Params_clicked()	Підпрограма обробки натискання кнопки «Налаштування комплекту»
	Save_clicked()	Підпрограма обробки натискання кнопки «Зберегти»
	Prepare_clicked()	Підпрограма обробки натискання кнопки «Вихід»
cparams.ui.h	ins_file()	Підпрограма вибору файлів комплекту ПЗ компонентів ПТК
	pushButtonOk_clicked()	Підпрограма збереження налаштувань комплекту ПЗ компонентів ПТК
	pushButtonCancel_clicked()	Підпрограма скасування змін комплекту ПЗ компонентів ПТК

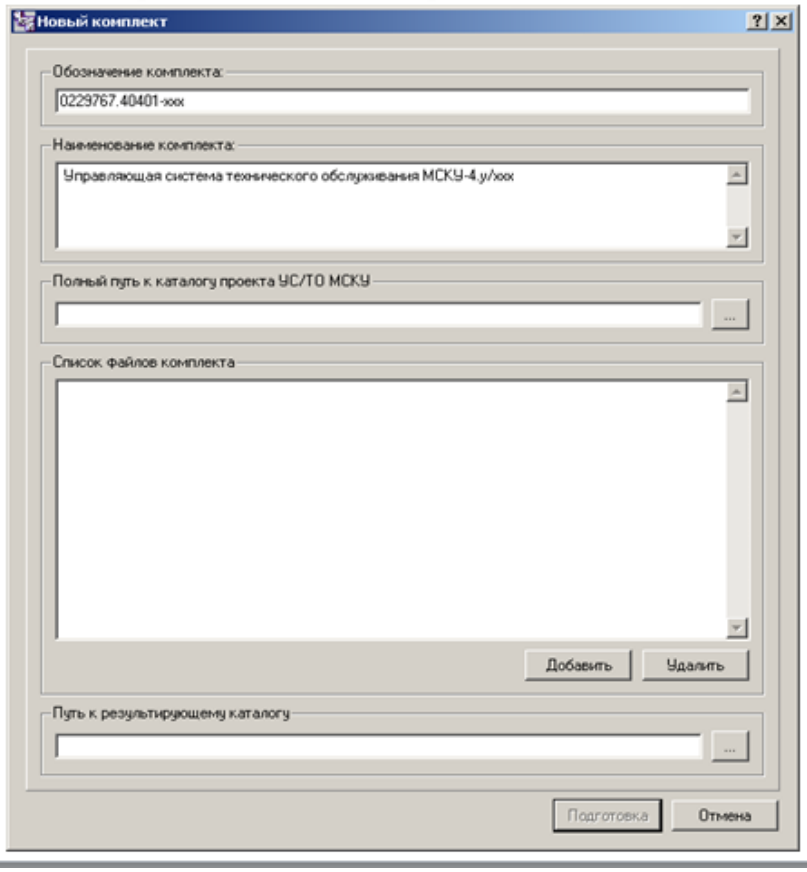
8

Формат вікна «Формування завдання на складання комплектів програмного забезпечення»



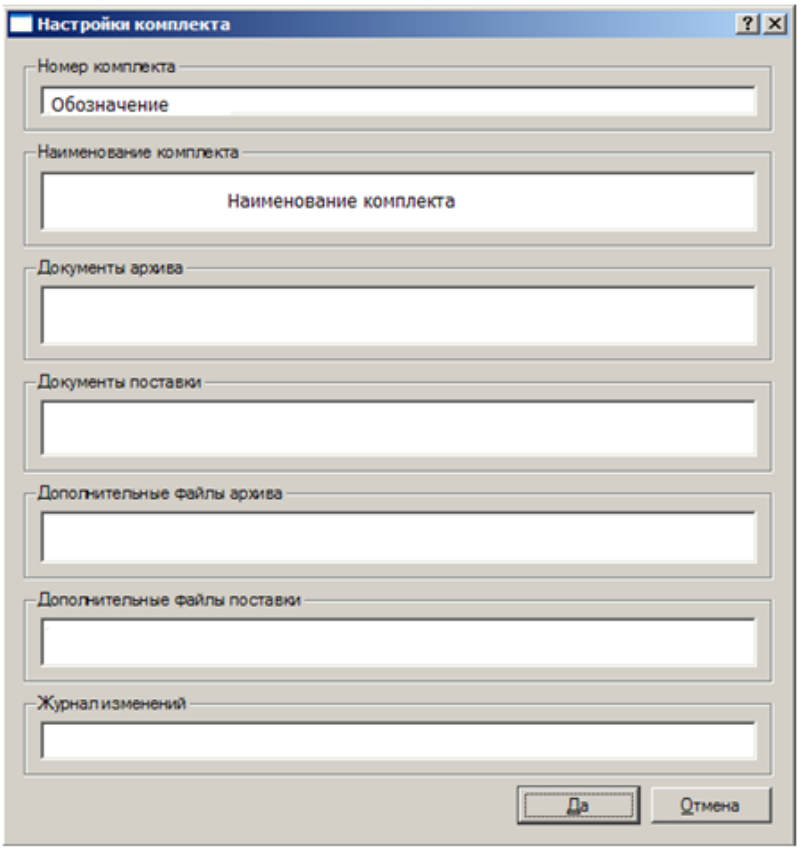
9

Формат вікна «Створити комплект»



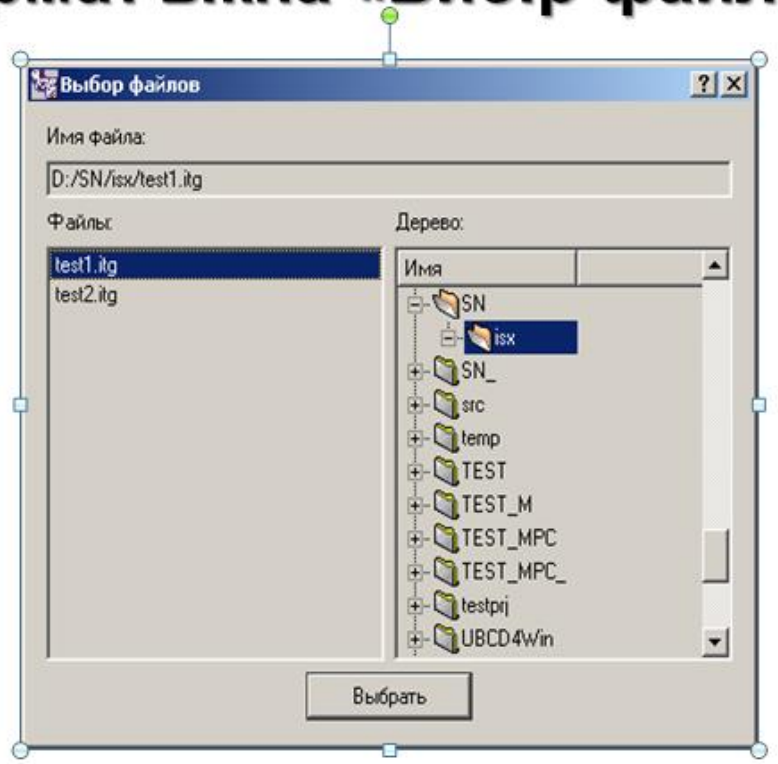
10

Формат вікна «Налаштування комплекту»



11

Формат вікна «Вибір файлів»



12

ПРОГРАМА PACKCREATOR

Програма PackCreator представляє собою набір взаємозв'язаних модулів, перелік яких наведено в поданій таблиці.

Ім'я файлу	Ім'я підпрограми	Виконувані функції
main.cpp	main()	Головна підпрограма
SupplyCreator.cpp	TC_SupplyCreator()	Підпрограма формування комплектів поставки для послідовності комплексів програм, певної у настроювальному файлі PackList.xml
	TC_WorkWithPrj()	Підпрограма формування в комплекті поставки даних про окремий проект групового проекту, що входить до складу комплексу програм
ArchiveCreator.cpp	TC_ArchiveCreator()	Підпрограма формування комплектів архіву для послідовності комплексів програм, певної у настроювальному файлі PackList.xml

13

ПРОГРАМА PACKINSTALLER

Перелік основних модулів класу PackInstaller наведено в поданій таблиці.

Ім'я файлу	Ім'я підпрограми	Виконувані функції
PackInstaller.cpp	run()	Підпрограма підготовки даних для основного діалогового вікна програми інсталяції комплектів архіву
	runInstallMode()	Підпрограма формування даних, необхідних для виконання інсталяції комплекту архіву
	runVerifyMode()	Підпрограма формування даних, необхідних для виконання перевірки цілісності встановленого комплекту архіву
	beginInstallProcess()	Підпрограма, яка виконує інсталяцію комплекту архіву
PackInstallerHandlers.cpp	OnCheckPackNum()	Підпрограма перевірки цілісності окремого комплекту архіву
	OnCheckPackNums()	Підпрограма перевірки цілісності встановлених комплектів архіву
	OnInstall()	Підпрограма, що запускає на виконання процес інсталяції комплекту архіву.

14

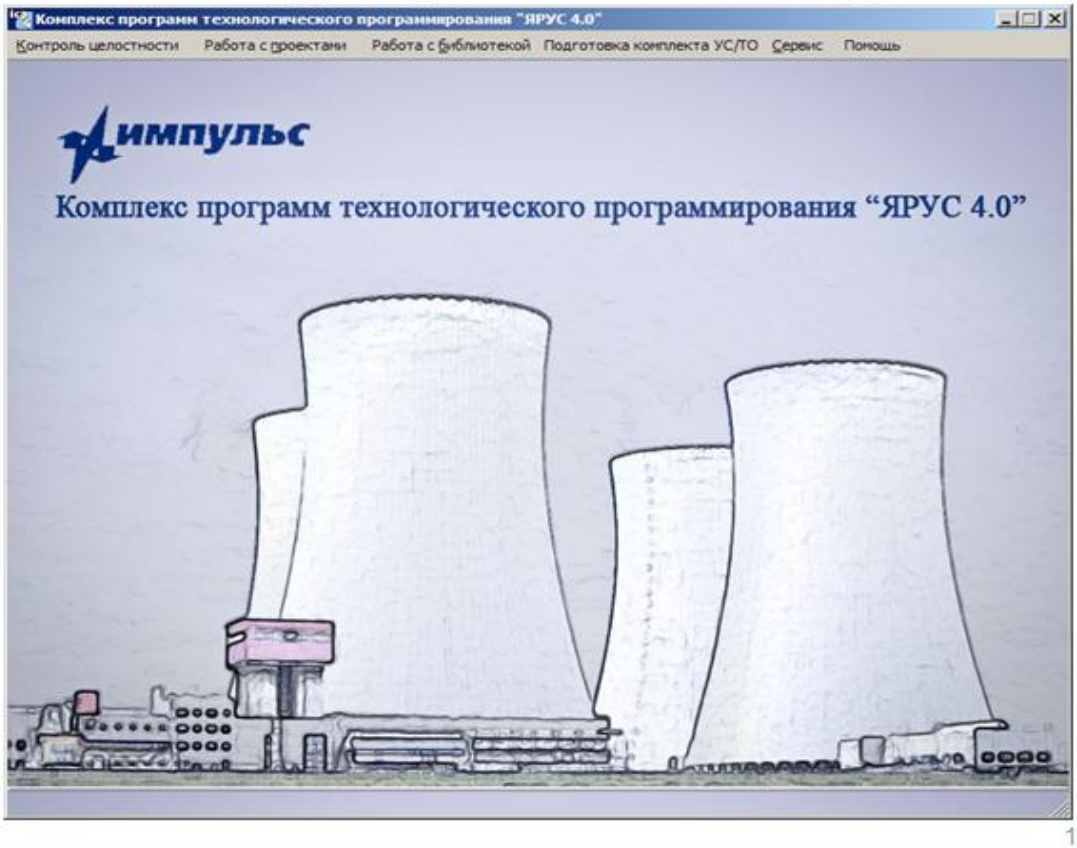
ПРОГРАМА PACKCOMPORATOR

Програма представляє собою набір взаємозв'язаних модулів, перелік яких наведено в поданій таблиці.

Ім'я файлу	Ім'я підпрограми	Виконувані функції
main.cpp	main()	Головна підпрограма
packcomporator helper.cpp	tc_comparepackages()	Підпрограма визначення переліку файлів комплекту ПЗ, створеного програмою підготовки, змінених у порівнянні з відповідним проінстальованим (еталонним) комплектом ПЗ

15

Основний кадр інструментальних засобів підготовки



16

ВИСНОВКИ

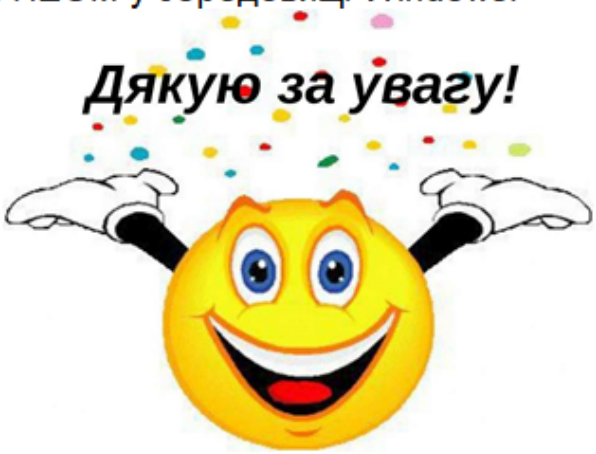
У представленій роботі виконано:

✓ дослідження вибраного об'єкту, розкриття актуальності завдання, сформовані мета й задачі дослідження, проаналізовано програмні засоби, використані при вирішенні поставленої задачі;

✓ розглянута, досліджена та детально описана логічна структура інструментальних засобів технологічного програмування;

✓ представлені алгоритм і комплекс програм, що забезпечують формування компонентів ПТК, розроблені на мові технологічного програмування, в ПЕОМ у середовищі Windows.

Дякую за увагу!



17