Розробка синтаксичної та семантичної моделі мови визначення і опису даних предметної області

Ігор Невлюдов[1], Владислав Євсєєв[1], Анастасія Демська[2]

1. Кафедра КІТАМ, Харківський національний університет радіоелектроніки, УКРАЇНА, Харків, пр. Науки. 14, email: vladyslav.yevsieiv@nure.ua
2. Кафедра МСТ, Харківський національний університет радіоелектроніки, УКРАЇНА, Харків, пр.Науки 14, email: anastasiia.demska@nure.ua

Анотація: У роботі запропонована синтаксична та семантична моделі нової формальної мови програмного продукту для КІС ТПВ «Jump». Визначені та описані термінології, в основу яких покладено запропоновані поняття моделі «Jump» і відносини між метаданими та даними предметної області і способів їх перетворення, приведена синтаксична діаграма мовної моделі, що розробляється.

Ключові слова: програмний продукт, предметна область, мовна модель, специфікація, синтаксична діаграма.

I. Вступ

Основою розвитку сучасного приладобудування у світі є комп'ютеризація та інтеграція всіх виробничих процесів та управління виробництвом від початку розробки до поставки готової продукції споживачеві, відповідно концепції міжнародних стандартів ISO серії 9000.

Якщо спиратися на ідеологію, відповідну зазначеним міжнародним стандартам [1], то слід в першу чергу говорити про інтеграцію з метою вдосконалення діяльності щодо забезпечення всіх етапів життєвого циклу (ЖЦВ) програмного продукту [2]. Існуючі аспекти та рекомендації, які представлені в [1] не дозволяють використовувати їх в рамках комплексного підходу для вирішення завдання розробки єдиної методології в області автоматизації проектування програмних продуктів (ПП) та програмних модулів (ПМ) для комп'ютерно-інтегрованих систем технологічної підготовки виробництва (КІС ТПВ). Всі існуючі стандарти мають часткові пропозиції і рекомендації щодо систематизації розробки програмних продуктів загального призначення.

Для розробки автоматизованої системи проектування ПП та ПМ КІС ТПВ необхідна розробка формальної мови опису структури і параметрів розроблюваного ПП і ПМ, що буде максимально зрозумілою і простою в описі предметної області (ПрО) і в застосуванні. З одного боку було виявлено що розробники ПП і ПМ для КІС ТПВ використовують прив'язку до елементів інтерфейсу користувача і мінімізують використання програмного коду, який в більшості випадків служить оброблювачем тієї чи іншої події, яка ініціалізується користувачем шляхом взаємодії з необхідним елементом для вирішення того чи іншого завдання. З іншого боку – користувачі, що є фахівцями ПрО, можуть сформулювати семантично правильну структуру і внутрішню ієрархію всіх елементів інтерфейсу користувача з необхідними значеннями параметрів і подій, які необхідно реалізувати для вирішення поставлених завдань в тій чи іншій КІС ТПВ, що розробляється.

Основною метою даного дослідження є розробка синтаксичної та семантичної моделі нової формальної мови, близької до деякої підмножині природної мови, яка б об'єднала в собі конструкцію побудови зрозумілої і простої як для фахівців ІТ, так і для звичайного користувача, ґрунтуючись на вимогах, що висуваються до моделі ЖЦ проектування ПП і ПМ для КІС ТПВ «Jump», її основних концепцій і математичній формалізації.

II. РОЗРОБКА СИНТАКСИЧНОЇ ТА СЕМАНТИЧНОЇ МОВНОЇ МОДЕЛІ

Модель предметної області – це модель домену GDM (Generative Domain Model), яка може включати в себе декілька моделей для окремих підгалузей ПрО, відповідних окремим членам сімейства програм. На їх перетині формуються загальні поняття, характеристики та обмеження в моделі характеристик ПрО. Опис кожної моделі здійснюється у відповідній предметно-орієнтованій DSL - мові. Даний опис за спеціальними правилами трансформується безпосередньо до відповідної мови програмування (МП) реалізації цієї моделі або в іншу DSL-мову, яка потім трансформується в необхідну МП [3].

Міжмовний інтерфейс – це зв'язок різних МП за типами, що містяться в них даних, методам їх організації і способам відображення цих засобів відповідними системами програмування.

Основними функціями міжмовного інтерфейсу є вирішення наступних чотирьох завдань [3]: забезпечення переходу від середовища функціонування однієї МП до середовища функціонування іншої; забезпечення передачі управління між різномовними модулями; забезпечення доступу до загальних даних; реалізація механізму передачі даних через параметри виклику.

Дамо визначення поняттю мова моделі (ММ) – це декларативна (непроцедурна) мова, призначення якої – визначення та опис термінологій, в основу яких покладено запропоновані поняття моделі «Jump» і відносини між метаданими та даними ПрО і способів їх перетворення.

В даному дослідженні пропонується наступна специфікація ММ даних:

- **дозволені буквено-цифрові символи** які підтримуються середовищами розробок для мов високого рівня програмування і відповідають таблиці ASCII-кодів: + - \ . , ! “ < > = ( ) $ % & ~ * _ & @ пробіл ; { };

- **ключові слова**: *parameter*, *value*, *name*, *event*, *cod*, *EvenForm*, *ParameterForm*, *ParametrElement*, *ElementForm*, *EventElement*, *ValueElement*, *Form*, *LingusticVariable*, *ContainerSolution*, $Form^{master}$, $Form^{slave}$;

- **ідентифікатори** використовуються для позначення таких ознак:

- ознака приналежності параметрів, подій до доменного або не доменного типу: *domen*, *not_domen*. Домени відповідних характеристик (значень) належать до не перераховуваного (облікового) типу, який має можливість вибору із заздалегідь сформованого списку. Прикладом може виступати деякий параметр *ParameterForm*, відображення *Form*, який може приймати значення *true* чи *false*, параметр *Align* який притаманний $dom(parameter^{1}_{k\_спис})$ та $dom(parameter^{1}_{j\_спис})$, який може приймати значення, яке фіксуються середовищем розробки.

- тип даних значень (*value*), який визначає характеристику параметрів *ParameterForm* і *ParametrElement* (текстове, булеве, цілочисельне, цілочисельне негативне, текстове слово поєднання).

- лінгвістичний опис ознаки посилання *LingusticVariable* (*name*) на *ContainerSolution*, який містить необхідний *cod*.

- базові поняття моделі ЖЦ «Jump», які дають можливість зв'язати події *ElementForm* і *EventElement*, які містять набір певних *event*, належних певному візуальному графічного елементу з *ContainerSolution* (*cod*) через *LingusticVariable* (*name*).

На відміну від ключових слів, запропоновані ідентифікатори, теоретично можна перевизначити, але це збільшує можливість виникнення помилок, внаслідок чого перераховані вище ідентифікатори входять в фіксований словник ключових слів.

- **літерали** – певний набір значень, які не представлено ідентифікатором.

Рядкові літерали представляється у вигляді послідовності дозволених символів з різним типом пропису (прописні та строчні) букв. Наприклад, *name_form*, назва форми яка використовуються в параметрах *Caption*, *Name* і.т.д, а також привласнення унікального імені (*name*) для кожного *LingusticVariable*, яке містить певний фрагмент програмного коду. Приклад «зберегти в БД», «розрахувати результат», і.т.п, яке задається кінцевим користувачем з метою зручності методології, що розробляється.

Алгебраїчні літери являють собою опис простих логічних операцій типу *True, False*, які дозволяють задати значення (*value*) тому чи іншому параметру (*parameter*), що належить *ParametrElement*, *ParameterForm*, що є необхідним і достатнім для опису властивостей візуального елемента інтерфейсу користувача проектованого ПП або ПМ для КІС ТПВ.

Зарезервовані літери представляють слово, словосполучення або скорочення, яке дає можливість вибрати ту чи іншу властивість параметру, необхідну для досягнення умов технічного завдання (ТЗ). Прикладом може виступати властивість форми *WindowsState* в середовищі розробки RadStudioXE16, для якого можна вибрати наступні зарезервовані скорочення: *wsNormal*, *wsMinimized*, *wsMaximized*, які розробник може задати при першому запуску, для визначення виду відображення форми. *wsNormal* – відображення за замовчуванням, подається в тому вигляді, в якому вона створювалася на етапі проектування, *wsMinimized* – форма відображається в мінімізованому вигляді на панелі завдань, *wsMaximized* – при запуску форма розгортається на весь розмір робочого столу.

Зарезервовані літери можуть бути загальними для *ParameterForm* і *ParametrElement*, а також спеціалізованими, тобто належними певній візуальній формі, яка визначає специфіку того чи іншого елемента. Але слід зауважити, що зарезервовані літери для визначення значень того чи іншого параметра для візуальних компонентів, маючи однакове призначення,

можуть виконувати різні задані функції в одному середовищі розробки

Цілочисельний тип даних (*integer*) – дозволяє привласнити параметру *ParameterForm* або / чи *ParametrElement*, певне і необхідне цифрове значення розмірності або координат розміщення візуального елемента відносно *Form*. В основному для опису візуальних графічних елементів використовується найменший логічний елемент двовимірного цифрового зображення в растровій графіці (*pixel*). Довжина рядка залежить від дозволу екрану і вимог ТЗ.

Цілочисельний негативний тип даних – дозволяє привласнити параметру певне значення, яке входить в діапазон $(-1,1,2,3,...,n)$, що належить виключно *ParametrElement* і описує нумерацію в даному контексті:

-1 – нумерація відсутня, параметр не задіяний;

$1,2,3,...,n$ – нумерація графічного зображення (іконки), котре належить певному параметру (*parameter*) для *ElementForm*.

Текстові / лінгвістичні (char) – дозволяють привласнити параметру логічно впорядковані значення символів, які містять необхідні для користувача пояснення, або назву графічних елементів, що необхідні для зручності роботи з проектованим ПП або ПМ для КІС ТПВ. Також даний тип подання значень використовується для завдання визначеного імені *LingusticVariable*, яке присвоюється події *EvenForm*, *EventElement*.

Логічний (булевий) – може приймати тільки два значення *true*, або *false*, які виконують роль перемикача використання того чи іншого параметра в *ParameterForm* і *ParametrElement*.

Текстове словосполучення (перераховуємий тип) – тип даних, заданий списком у вигляді домена [2], що дозволяє задати список зарезервованих слів або скорочень в середовищі розробки, що приймаються для того чи іншого *parameter* в *ParameterForm* і *ParametrElement*.

Роздільники – символьне позначення виділення основних елементів синтаксичної конструкції ММ, що розробляється.

<*Form*> (кутові дужки *Form*) – використовується для вказівки ключового слова, яке є початком метаопису тієї чи іншої *Form* в конструкції ММ.

</*Form*> (слеш кутові дужки *Form*) – використовується для вказівки ключового слова, яке показує завершенням метаопису тієї чи іншої *Form* в конструкції ММ.

Для пропонованої конструкції ключового слова початок і завершення метаопису *Form* накладені такі обмеження: у назві *Form* може міститись нумерація у вигляді *Form*1, або буквене визначення, наприклад *Form_master*, або *Form_add_operat*, при цьому обов'язково ключове слово початку метаопису має збігатися з ключовим словом завершення метаопису тієї чи іншої *Form* в конструкції ММ. При невиконанні даної вимоги до конструкції, інтерпретатор ММ не зможе сприйняти вміст метаопису всіх необхідних параметрів і подій властивих даній *Form*.

{(фігурна дужка, що відкривається) – обов'язковий символ початку рядка метаопису *Form* і *ElementForm*.

} (фігурна дужка, що закривається) – обов'язковий символ завершення рядка метаопису *Form* і *ElementForm*.

(решітка) – після цього символу конструкція інтерпретатора ММ сприймає як початок опису графічних візуальних елементів інтерфейсу користувача (*ElementForm*).

/ # (слеш решітка) – після даної комбінацій символів інтерпретатор ММ вважає, що опис графічних візуальних елементів інтерфейсу користувача (*ElementForm*) закінчений.

/ (слеш) – використовується для завдання ієрархій метаописів візуальних графічних елементів (*ElementForm*), у відповідності з деревом побудови ПП та ПМ для КІС ТПВ, і застосовується всередині # / # метаопису *Form*. *ElementForm*1/*ElementForm*2 – розуміється як *ElementForm*2, що знаходиться всередині *ElementForm*1 і є її невід'ємною частиною.

[] (квадратні дужки) – використовується для завдання метаописів, необхідних параметрів і подій *ParameterForm*, *EvenForm*, *ParametrElement*, *EventElement*.

; (крапка з комою) – обов'язковий символ конструкцій ММ, який показує, що для даного *parameter* або *event* привласнення *value* і *name* відповідно завершено, дозволяється застосовувати всередині [3,4].

,…, (перерахування через кому) – використовується для перерахування назв *parameter* для *ParameterForm*, *ParametrElement*, а також *event* для *EvenForm*, *EventElement*, при умові, що з набору декількох *parameter* або *event* значення *value* і *name* відповідно однакове і застосовується всередині [4,5].

= (знак рівності) – привласнює *parameter* певне значення типу даних *value* і застосовується для вказівки події (*event*) певне *name* з *LingusticVariable*, яке містить посилання на *cod*, або його фрагмент в *ContainerSolution*. Варто врахувати, що в

залежності від контексту (логіки та змісту виконуваних дій), даний знак можна трактувати і як інструкцію присвоєння, згідно з якою для зазначеного базового параметра визначається йому належне значення.

Коментарі – всі символи і рядки, які записані всередині даної конструкції інтерпретатора ММ, що ігноруються і сприймаються як коментар. Допустимі до використання буквено-цифрові символи національних алфавітів, що підтримуються операційною системою і середовищем розробки, обмеженням для коментарів служить те, що їх послідовність не повинна перевищувати 255 символів.

? ** – показує, що після заданих символів, слідує коментар який ігнорується інтерпретатором ММ.

**? – показує, що після заданих символів закінчується коментар і далі текст не ігнорується інтерпретатором ММ.

Для адаптації даного синтаксису опису ММ, нами запропоновано використовувати форму Бекуса-Наура
[4-5], яка дає можливість розробити інтуїтивно просту і адаптивну формальну мову подання та опису необхідних даних для розробки ПП і ПМ для КІС ТПВ.

Синтаксична діаграма ММ, що розробляється представлена на рисунку 1.

Синтаксична діаграма запропонованих в даному дослідженні типів представлення значень, які можуть належати ідентифікатору, представлена на рисунку 2.

Як можна бачити з рис. 2 ідентифікатор *Parameter*, *Event* відносяться до *domen*-ого (облікового) типу і представляється у вигляді текстового слова або скорочення, які відносяться до *ParameterForm _ name* і *EvenForm _ name*, а також *ParametrElement* і *EventElement* відповідно з рис. 1.

Список параметрів (*parameter*) і подій (*event*), які належать *ParametrElement*, *EventElement*, в межах одного середовища розробки, є постійними і не змінними. Для списку параметрів (*parameter*) і подій (*event*), які належать *ParametrElement*, *EventElement* відповідно, є однаковим з обмеженням, що це візуальні графічні елементи, які мають однакове призначення в рамках середовища розробки. Варто звернути увагу, що до даного типу відносять *value* для *ParametrElement* та *EvenForm _ name*, який містить зарезервоване середовищем розробки текстове слово чи скорочення.

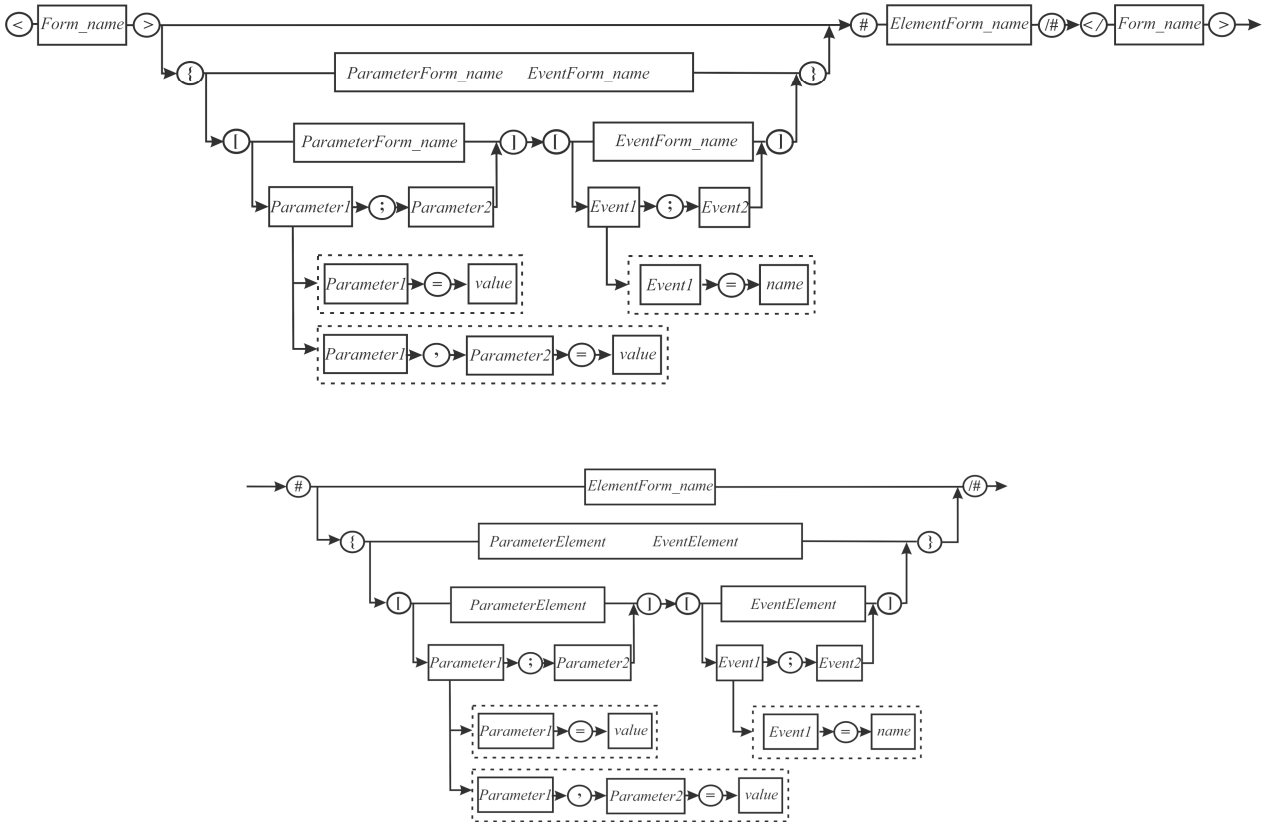

Рис.1. Синтаксична діаграма ММ, що розробляється

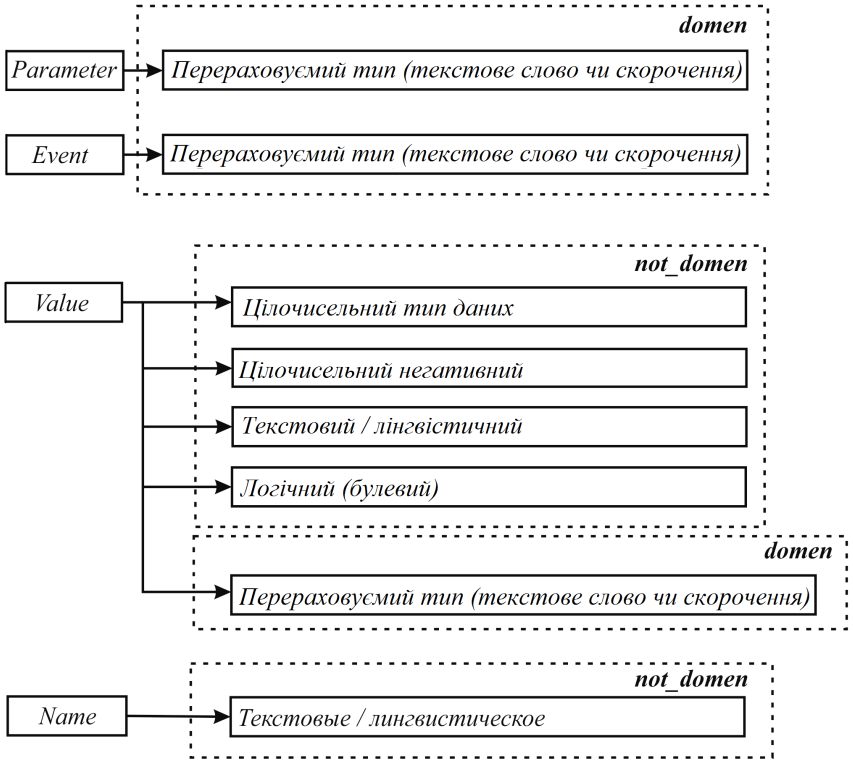

Рис. 2. Синтаксична діаграма типів представлення значень ідентифікаторів

Ідентифікатори *value* та *name* відносять до *not_domen* -го (не списочного) типу. Це обумовлено тим, що значення *value* може задаватись розробниками в залежності від вимог ТЗ до ПП чи ПМ КІС ТПВ.

Для ідентифікатора *name* , що входить до *LinguisticVariable* , ім'я, яке посилається на *ContainerSolution* містить необхідний фрагмент або частину програмного коду (*cod*) та задається користувачем, з урахуванням його логічних вподобань та зручності використання.

Для зручності читання та подання розробленої декларативної мови (рис. 1 – 2) необхідно, що б вона була зручна для розуміння і читання. Це можна досягти, використовуючи хоча б три принципи подання мови [.....]:

- бути максимально лінійною;
- бути короткою;
- бути самодокументованою.

При необхідності реалізації ієрархії (дерева побудови) приналежності візуальних графічних елементів *ElementForm*1/*ElementForm*2 , пропонується наступний фрагмент структури метаопису:

\# "ім'я елемента в середовищі розробки"?** відкриття блоку опис візуальних графічних елементів *Form_master* **?

{ ?** блок опису *Element*1*Form _ master* **?

[*parameter*1 = *value*; *parameter*2, *parameter*3 = *value*]

[*event*1 = *name*; *event*2 = *name*]

}?** закриття блоку опису *Element*1*Form _ master* **?

/ "ім'я елемента в середовищі розробки"

{ ?** блок опису *Element*2*Form _ master* **?

[*parameter*1 = *value*; *parameter*2, *parameter*3 = *value*]

[*event*1 = *name*; *event*2 = *name*]

}?** закриття блоку опису *Element*2*Form _ master* **?

/# ?** закриття блоку опису візуальних графічних елементів *Form_master* **?

Використання "/" (слеш) дозволить інтерпретатору ММ визначити ступінь приналежності візуального елемента до іншого, тобто в середовищі розробки реалізувати дерево побудови (Structure) проектування ПП або ПМ для КІС ТПВ.

III. Висновки

Ґрунтуючись на запропонованих припущеннях і рекомендаціях для розроблюваної декларативної мови проектування ПП і ПМ для КІС ТПВ, пропонується розроблений тип стилю запису мови моделі, яка дає можливість спростити та стандартизувати код. Також можна задати будь-яку глибину вкладення графічних елементів інтерфейсу користувача, що дає можливість реалізувати за допомогою запропонованої синтаксичної діаграми ММ (рис. 1) структуру ПП

або ПМ для КІС ТПВ будь-якого ступеня складності, і спростити їх розробку на початковому етапі проектування.

ПЕРЕЛІК ПОСИЛАНЬ

[1] ISO/IEC 12207: 1995–0801: Informational Technology – Software life cycle processes. // ГОСТ Р ИСО/МЭК 12207–99. Информационная технология. Процессы жизненного цикла программных средств.

[2] Невлюдов І.Ш., Євсєєв В.В., Демська А.І. Розробка моделі життєвого циклу розробки програмного продукту та програмних модулів для КІС ТПВ // Технология приборостроения . - 2017, №1.- с. 12-16.

[3] Лаврищева Е.М., Грищенко В.Н. Сборочное программирование. Основы индустрии программных продуктов: монографія. –Киев: Наук. думка, 2009.– 372с.

[4] Кургаев А. Ф., Григорьев С. Н. Нормальные формы знаний // Доповіді Національної академії наук України, 2015, № 11 – с. 36–43.

[5] Черников Ю.Г. Системный анализ и исследование операций. монография.- Москва, МДГУ, 2006.-370 с.

Механізація виготовлення деталей приладобудування методом холодної листової штамповки

Вячеслав Роменський, Олексій Макаренко

Кафедра КІТАМ, Харківський національний університет радіоелектроніки, УКРАЇНА,
Харків, пр. Науки. 14, e-mail: vjacheslav.romenskiy@nure.ua

Анотація: В даному матеріалі наведено приклад механізації виготовлення деталей приладобудування методом холодної листової штамповки в одиничному дрібносерійному та серійному виробництві із стального прокату різної товщини.

***Ключові слова*:** приладобудування, холодна листова штамповка, механізм автоматичної подачі полоси, транспортер, касета, механізм підйому смуги, промисловий робот, маніпулятор

I. ВСТУП

Холодна листова штамповка знайшла широке призначення в приладобудуванні і машинобудуванні при виготовленні плоских деталей із стального прокату різної форми, розміру і товщини.

Проаналізував основні виробничо-технологічні процеси штампування виявлено, що деталі товщиною до 0,5 мм, особливо у масовому і крупносерійному виробництві виготовляють із рулонного листа на спеціально розроблених автоматичних лініях. Деталі товщиною більше 0,5 мм штампують із карток або смуг, які виготовляють у заготівельних цехах підприємства. Штампування із карток виконує один оператор, а із смуг (полоси) – два: один подає полосу в зону штампування (штами), другий – штампує. Робота штамповщика є монотонною, трудоємкою та шкідливою (шум преса).

До кінця робочої зміни робітник втомляється, а це впливає на продуктивність і якість продукції.

У зв'язку з цим є доцільним на підприємствах, де виконується холодне штампування з полос у ручну, техпроцес замінити на механізований.

II. ПОСТАНОВКА ПРОБЛЕМИ

В якості прикладу розглянемо структурну схему автоматизованої лінії холодної листової штамповки для карток з двома промисловими роботами (рис.1) [1].

Роботизована лінія двоопераційної штамповки з застосуванням промислових роботів складається з подаючого пристрою (1), захоплення картки (2), преса (3), пульта керування пресом (4), штампа (5), датчика контролю захвата картки (6), передавального пристрою (7), промислового робота (8), системи управління ПР (9), тари готової продукції (10).

В даній роботі поставлено мету розробки механізму автоматичної подачі смуг (полоси) в порожнину штампа.

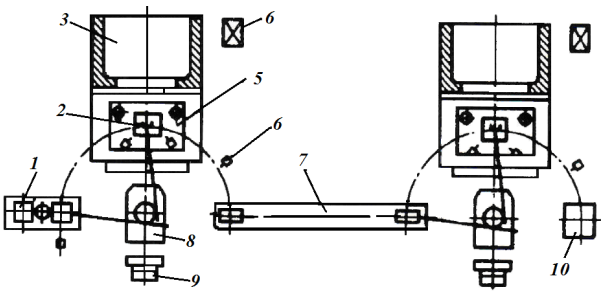

Рис. 1. Структура роботизовано лінії з двома ПР