УДК 004.4’236:004.93

# ВІЗУАЛЬНЕ ПРОГРАМУВАННЯ ЯК ХУДОЖНІЙ МЕТОД РОБОТИ З ЦИФРОВИМИ МЕДІА

Березова А. С., Солодов В.Д.
e-mail: arianna.berezova@nure.ua
Харківський національний університет радіоелектроніки, каф. МІРЕС,
м. Харків, Україна

The publication is devoted to the analysis of visual programming as an artistic approach to working with digital media. The main principles of visual programming languages and their advantages compared to text languages in the context of artistic and design disciplines are considered. The node-based logic of building systems is described in detail: the concepts of a node, data flow, modularity and scalability. The specifics of algorithmic thinking in art are analyzed – building logic instead of a linear timeline, working with parameters and randomness.

Візуальне програмування – це підхід до побудови програм, у якому замість рядків текстового коду використовуються графічні елементи: блоки, вузли, значки та з’єднання між ними. Мови візуального програмування (Visual Programming Languages, VPL) надають середовище, де користувач конструює логіку роботи системи, маніпулюючи об’єктами у просторі, а не набираючи синтаксичні конструкції [1]. Загальна мета VPL – зробити програмування доступнішим і усунути синтаксичні помилки, які неминуче виникають при роботі з текстовим кодом.

Порівняно з традиційними текстовими мовами – Python, C++, Java – візуальне програмування має принципову відмінність: логіка програми стає видимою. У текстовому коді зв’язки між компонентами системи приховані всередині абстрактних рядків; у візуальній мові кожне з’єднання між вузлами є буквально намальованою лінією на екрані.

Для візуальних дисциплін – медіамистецтва, дизайну, сценографії – ця властивість є критично важливою. Монтажер мислить категоріями форм, потоків і перетворень, а не алгоритмічних рядків. Візуальне програмування дозволяє перенести цю просторову логіку безпосередньо в середовище розробки: будувати, переміщувати, з’єднувати і роз’єднувати компоненти системи з такою ж інтуїтивністю, з якою фізичний художник розставляє об’єкти на столі. TouchDesigner, розроблений компанією Derivative, є одним із найбільш потужних прикладів такої мови: він поєднує node-based підхід з повноцінним середовищем для 2D/3D-композитингу, обробки аудіо та управління зовнішніми пристроями в реальному часі [2].

Центральним поняттям у візуальному програмуванні є вузол (node, або у термінології TD – оператор). Вузол – це автономний функціональний блок, який отримує певний тип вхідних даних, перетворює їх за вбудованою або налаштованою логікою і передає результат далі. В TouchDesigner

оператори об’єднані у кольорові сімейства: TOP (зображення та відео), CHOP (канали даних та аудіо), SOP (3D-геометрія), DAT (текстові дані та скрипти) та інші [2]. Такий поділ дає монтажеру швидкий візуальний орієнтир: колір вузла одразу повідомляє, з яким типом даних він працює.

З’єднання між вузлами формують потік даних – ключову концепцію node-based архітектури. На відміну від лінійного сценарію або таймлайну, де дії відбуваються послідовно, у data flow дані постійно «течуть» мережею операторів, оновлюючись у реальному часі при будь-якій зміні параметрів. Це означає, що система є живою та процедурною: змінивши один параметр у вершині ланцюга, монтажер миттєво бачить, як зміна поширюється через усю мережу до кінцевого результату [3]. Компанія Derivative описує цю властивість як «always-alive system» – система продовжує генерувати себе навіть у процесі редагування, що принципово відрізняє її від традиційних середовищ, де потрібна компіляція.

Модульність і масштабованість є природними наслідками node-based підходу. Кожен вузол являється незалежним модулем, що може бути повторно використаний, замінений або перекомпонований без руйнування решти системи. У TouchDesigner групи операторів можна упакувати в компонент (COMP), перетворивши складну підсистему на один вузол з виразним ім’ям. Це робить навіть великі проєкти керованими та зрозумілими, адже користувач завжди може побачити, що всередині будь-якого рівня системи [2]. Дисципліна організації мережі стає настільки ж важливою творчою навичкою, як і генерація самого контенту.

Традиційна анімація і монтаж базуються на лінійному таймлайні: художник розставляє події і ключові кадри у часовій послідовності, фіксуючи кожний момент розвитку твору. Алгоритмічний підхід замінює цю логіку на побудову правил і зв’язків між компонентами. Монтажер формулює, як система має реагувати за будь-яких умов, задаючи правила її народження через параметри [4].

Параметри і змінні є основними виражальними засобами в node-based середовищі. Кожен оператор має набір параметрів, що визначають його поведінку: розмір, швидкість, колір, амплітуду тощо. Більше того, параметри можна зв’язати між собою: швидкість руху об’єкта може залежати від гучності звуку, колір – від даних датчика.

В TouchDesigner це реалізується через оператори Noise, Random та їхні варіації: Perlin-шум, наприклад, забезпечує «контрольований хаос», що залишається органічно зв’язною у просторі та часі [3]. Ця здатність будувати мережі залежностей між параметрами є ключовою рисою алгоритмічного художнього мислення, коли мистецтво стає системою зв’язків, а не набором зафіксованих станів.

Швидке прототипування є однією з найбільш значущих переваг, які node-based підхід привносить у творчий процес. У традиційному програмуванні між ідеєю і її перевіркою стоїть написання коду, компіляція та на-

лагодження – цикл, що займає від хвилин до годин. У TouchDesigner з'єднання двох вузлів і зміна параметра займає секунди, а результат помітно одразу [2].

Ітеративний підхід, природний для node-based систем, перетворює процес створення на безперервний діалог між користувачем та системою. Монтажер будує початкову мережу, спостерігає поведінку системи, коригує параметри, додає або вилучає вузли, рухаючись ближче до задуманого візуалу [3].

Коли система є живою і процедурною, вона перестає бути пасивним носієм художнього задуму і стає активним учасником творчого процесу. Кожна зміна параметра породжує відгук, кожен відгук стимулює нову зміну. Користувач у цьому процесі є не тільки автором системи, але й її спостерігачем і інтерпретатором. Цей діалог між людиною і алгоритмом є сутністю того, що відрізняє сучасне медіамистецтво від будь-якої попередньої художньої практики.

Таким чином, візуальне програмування і node-based підхід є не просто зручним інтерфейсом для технічних операцій, а принципово новим способом художнього мислення. Він дозволяє монтажерам будувати складні динамічні системи, мислити зв'язками і потоками замість фіксованих станів, та вступати в діалог із системою, яку вони самі створюють. TouchDesigner на сьогодні є одним із найбільш повних втілень цієї парадигми, що пояснює його широке поширення в сучасному медіамистецтві.

Node-based середовища підтримують побудову наочних моделей обчислювальних процесів, у яких взаємозв'язки між елементами системи представлені явно. Це полегшує аналіз логіки роботи програми, виявлення помилок і розуміння причинно-наслідкових залежностей між операціями. На відміну від текстових мов, де структура програми часто є розподіленою між різними рівнями абстракції, візуальне програмування концентрує ключові елементи управління в єдиному просторовому полі. Такий підхід є ефективним у міждисциплінарних контекстах, поєднуючи технічні та художні методи роботи.

Список використаних джерел:

1. Chang S.-K., Ichikawa T., Ligomenides P. (Eds.) Visual Languages. Springer, 1990. – 352 p.

2. Derivative. TouchDesigner User Guide. URL: https://derivative.ca/UserGuide/TouchDesigner (дата звернення: 26.02.2026).

3. Foro3D. TouchDesigner: Building Visual Systems with Nodes 2026. URL: https://foro3d.com/en/2026/january/touchdesigner-building-visual-systems-with-nodes.html (дата звернення: 01.03.2026).

4. Pearson M. Generative Art: A Practical Guide Using Processing / Matt Pearson. – Manning Publications, 2011. – 240 с.