

Dual-Language General-Purpose Self-Hosted Visual Language and new Textual Programming Language for Applications

لغة مرئية ذاتية الإستضافة ثنائية اللغة للأغراض العامة
ولغة برمجة نصية جديدة للتطبيقات

Presented by:

Mahmoud Samir Fayed

Supervised by:

Dr. Yousef Ahmed Alohal



Kingdom of Saudi Arabia

King Saud University

Department of Computer Science

College of Computer and

Information Sciences

This relates to the dissertation submitted for the degree
Doctor of Philosophy in Computer Science, May 28, 2025

Thesis Presentation Outline

01

Introduction

02

**Problem Statement & Motivation &
Contributions**

03

Literature Review

04

Materials & Proposed Methods

05

Experimental Results

06

Discussion and Comparisons

07

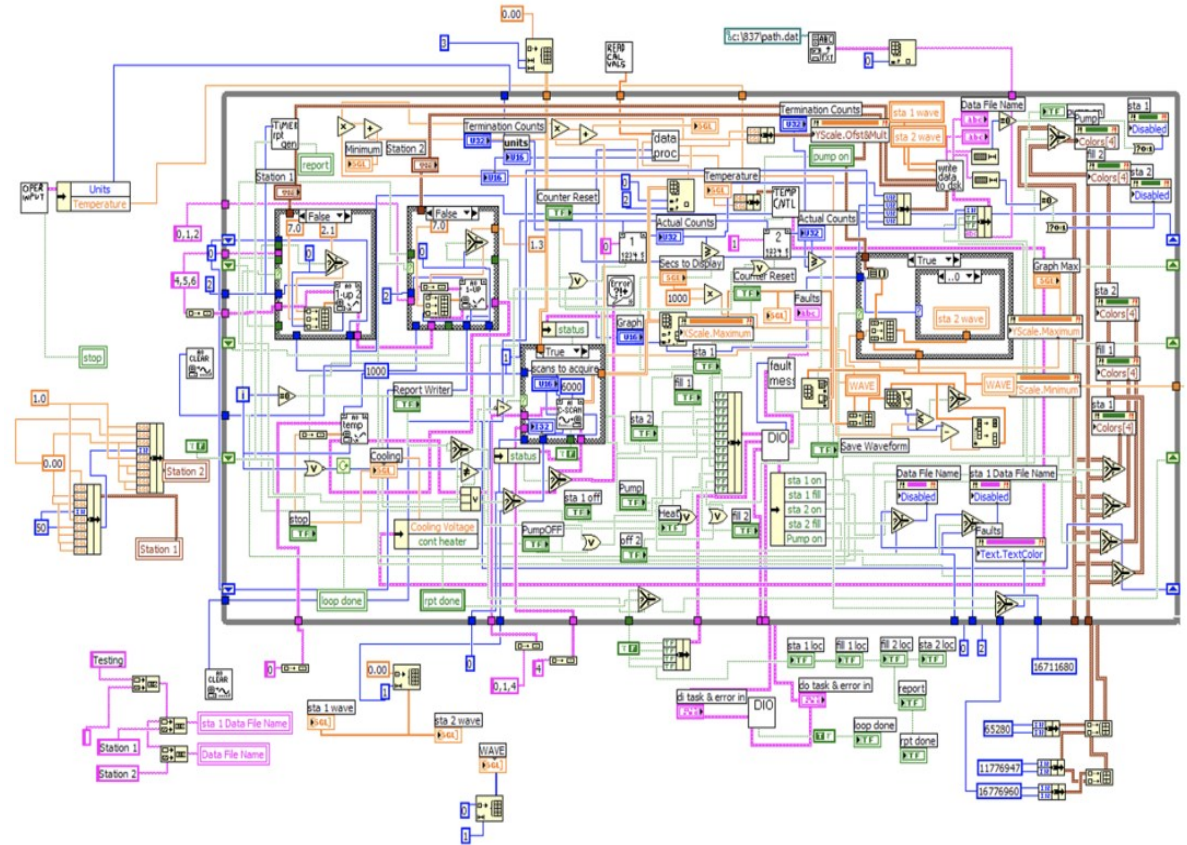
Conclusion

Visual Programming Languages (VPLs):

- Uses more than one dimension.
- Interaction with graphical elements.

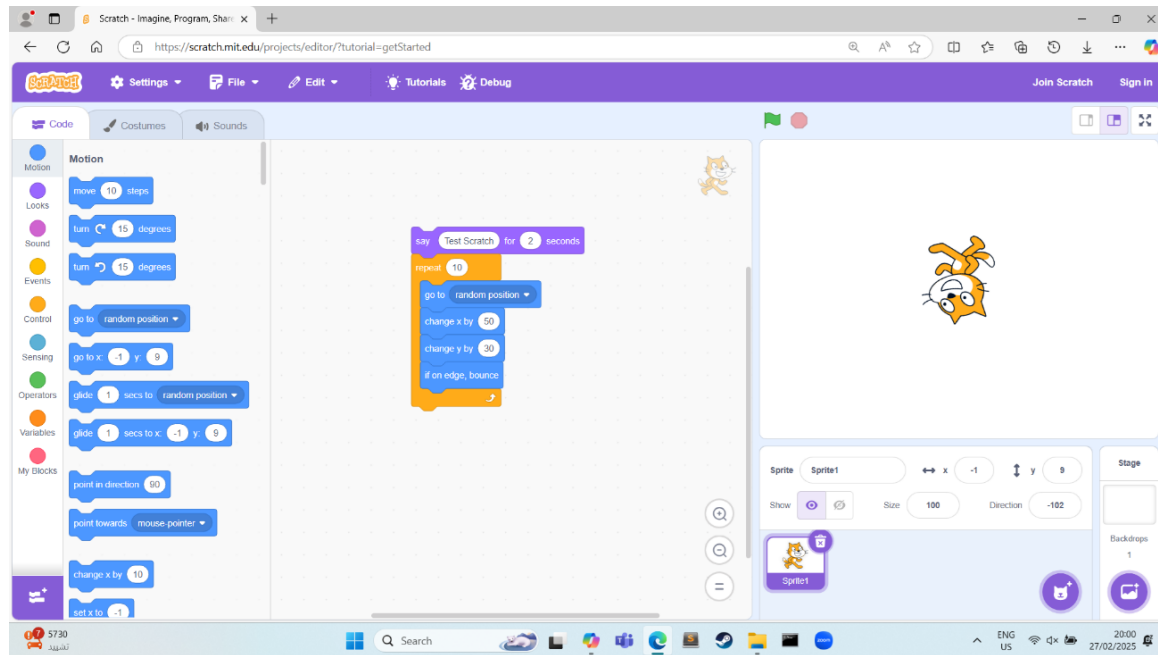
Visual representations:

- Diagrammatic
- Iconic
- Form-based
- Block-based
- Hybrid



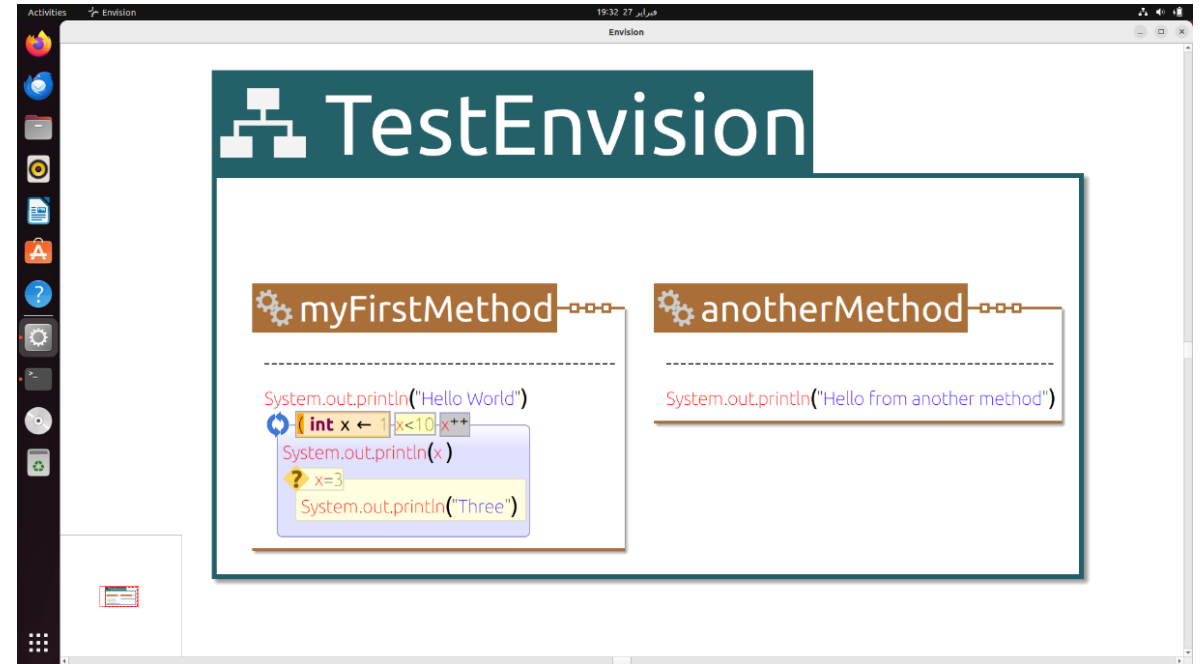
The LabView VPL.

Problems: Larger than the text, Maze of wires, etc.



(A): The Scratch Visual Programming Language.

- Drag-and-drop.
- No fast interactions (keyboard).
- No time dimension.



(B): The Envision Visual Programming System.

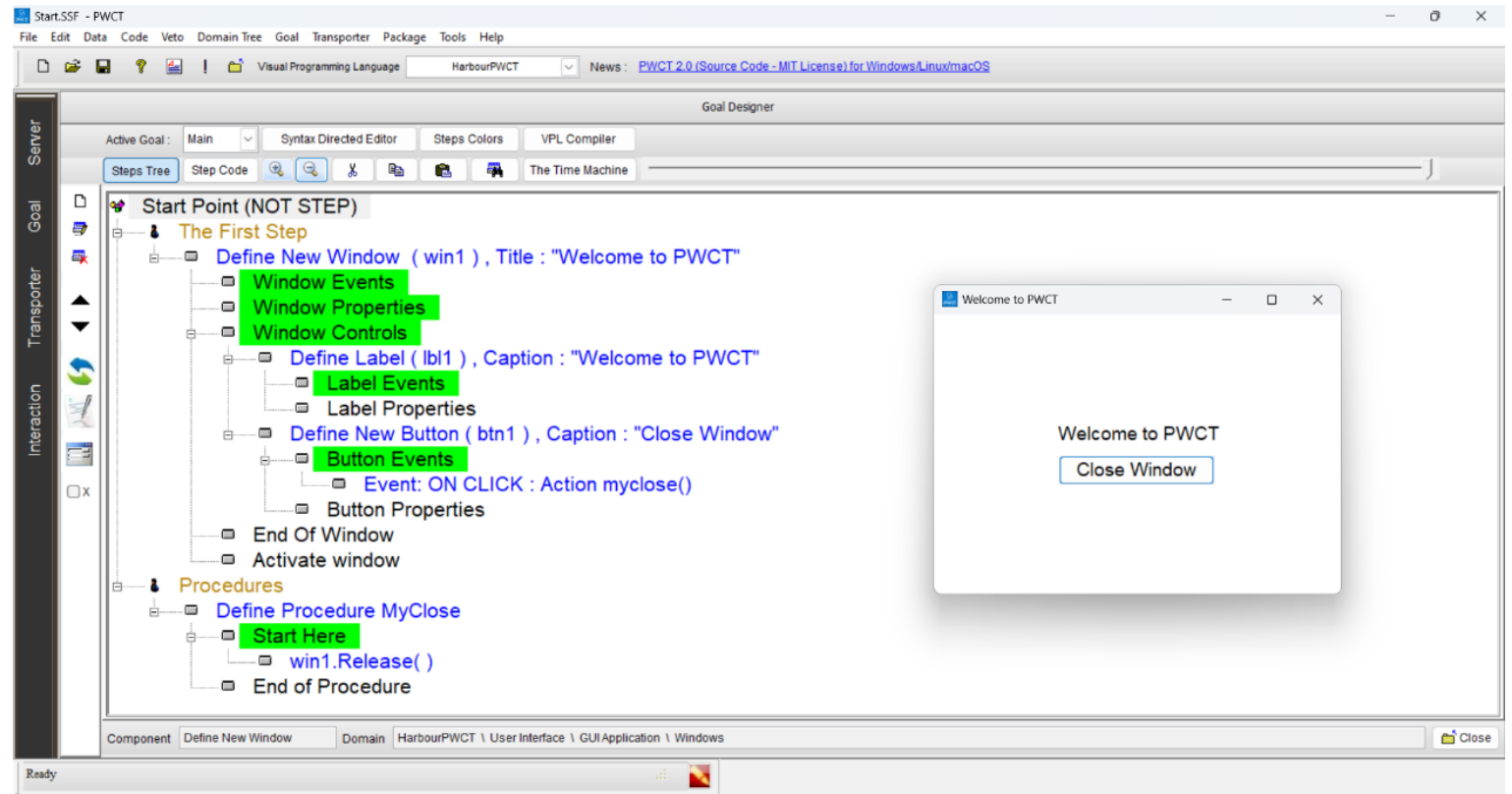
- Fast interactions (Keyboard).
- Not a Self-hosting VPL.
- Developed using C++.

Features:

1. General-Purpose.
2. Developed at KSU.
3. Many programming languages.
4. Graphical Code Replacement method.

Limitations:

- Windows Product.
- No Translations.
- Developed using VFP.



The PWCT visual programming language.

Thesis Presentation Outline

01

Introduction

02

**Problem Statement & Motivation &
Contributions**

03

Literature Review

04

Materials & Proposed Methods

05

Experimental Results

06

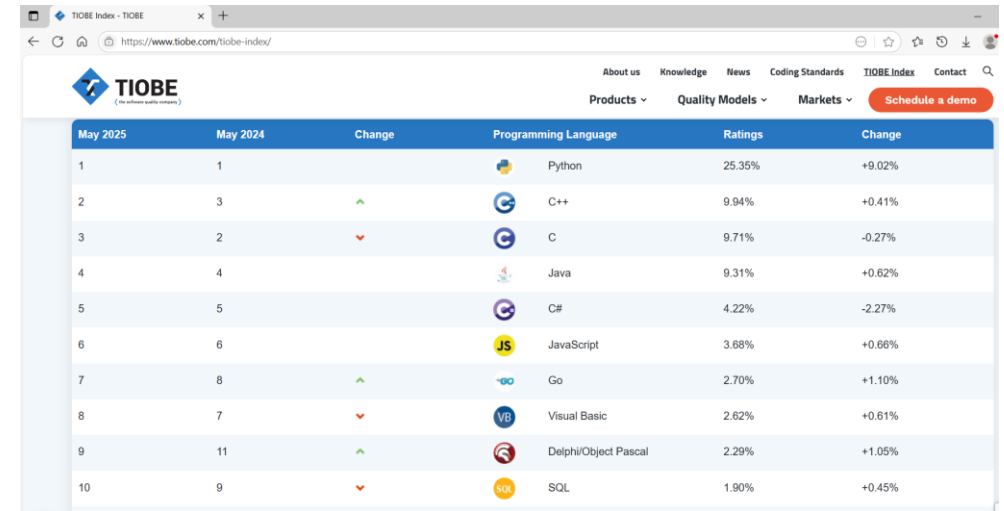
Discussion and Comparisons

07

Conclusion

Existing general-purpose visual programming languages, such as Envision and PWCT, have **limitations** that reduce their effectiveness in application development. Furthermore, there is **no practical evidence** of their use in **developing large or complex low-level systems**, such as the creation of a new textual programming language. Also, improving them requires textual programming.

- Success of many domain-specific VPLs.
- The necessity of being able to import/export textual code.
- Designing a new textual programming language for tools development will help many similar projects in their mission.
- No popular programming language is developed in Middle East.



The screenshot shows the TIOBE Index website. The table lists the top 10 programming languages as of May 2025, compared to May 2024. The languages are ranked by their percentage of the total programming language market.

	May 2025	May 2024	Change	Programming Language	Ratings	Change
1	1			Python	25.35%	+9.02%
2	3		▲	C++	9.94%	+0.41%
3	2		▼	C	9.71%	-0.27%
4	4			Java	9.31%	+0.62%
5	5			C#	4.22%	-2.27%
6	6			JavaScript	3.68%	+0.66%
7	8		▲	Go	2.70%	+1.10%
8	7		▼	Visual Basic	2.62%	+0.81%
9	11		▲	Delphi/Object Pascal	2.29%	+1.05%
10	9		▼	SQL	1.90%	+0.45%

TIOBE Index.

- Using **visual programming** to develop/maintain a Compiler/Virtual Machine for many years.
- Dynamic language (**lightweight implementation, rich features, Desktop/WebAsm/Microcontrollers/etc**).
- **Novel features** that can extend the **OOP** paradigm (Develop DSL **resemble CSS/Supernova**).
- The research prototype **PWCT2** (**lower storage requirements, better performance, etc.**).
- VPL that supports Ring language (**394 visual components**).
- Textual-to-visual code conversion tool (enables a **self-hosting VPL based on Ring**).
- Testing the **feasibility of using the Ring language** in the development of **PWCT2**.
- **Arabic Translation** for the **PWCT2** Environment/Components.

Thesis Presentation Outline

01

Introduction

02

**Problem Statement & Motivation &
Contributions**

03

Literature Review

04

Materials & Proposed Methods

05

Experimental Results

06

Discussion and Comparisons

07

Conclusion

Category	Examples
Lightweight and Embeddable	Lua, Squirrel, Wren, etc.
Comes with Ready-to-Use Libraries	Tcl, Perl, Python, etc.
Support creating Embedded DSLs	Lisp, Ruby, etc.
Comes with Powerful IDEs	Smalltalk, Visual FoxPro, etc.
Supporting Non-English Syntax	Supernova, Citrine, etc.
Domain-specific dynamic languages	R, xBase, etc.
Concurrency-oriented design	Erlang, Elixir, etc.
Comes with a focus on Performance	Julia, Mojo, etc.
Other implementations	MicroPython, mRuby, etc.

(A) Dynamic TPLs.

Category	Examples
Block-based	Scratch, Snap!, etc.
Diagrammatic	Tersus, RAPTOR, etc.
Iconic	Kodu, Limnor, etc.
Form-based and spreadsheet-based	Forms/3, FAR, etc.
Domain-specific	Blueprints, Pure Data, etc.
General-purpose	PWCT, Envision, etc.

(B) VPLs

Research Gap (In Dynamic TPLs)

- Most of the dynamic languages are developed using textual programming.
- Few studies about developing a language with lightweight implementation and rich features.
- Few languages provides support for translation.
- Embedded DSLs doesn't resemble external DSLs like CSS/SQL/Supernova/xBase.

Criteria	Lua	Python	Ruby	VFP	Supernova	Proposed Language (Ring)
Open Source	√	√	√	X	√	√
Portable	√	√	√	*	*	√
Lightweight	√	*	*	X	√	√
Embeddable	√	√	√	X	X	√
Dynamic Typing	√	√	√	√	√	√
Function like Eval()	√	√	√	√	X	√
Classes Concept	*	√	√	√	X	√
Inheritance Concept	*	√	√	√	X	√
Private Attributes	*	*	√	√	X	√
Batteries Included	*	√	√	√	*	√
IDE	*	√	*	√	*	√
Form Designer	*	*	*	√	*	√
Non-English Syntax	*	*	*	*	√	√
Case insensitive	X	X	X	√	√	√
1-based indexing	√	X	X	√	√	√
Change Keywords	X	X	X	X	X	√
Internal DSL	√	√	√	√	X	√
IDSL (Custom Syntax)	X	X	X	X	X	√
Visual Implementation	X	X	X	X	√	√
VI Based on CPWCT	X	X	X	X	X	√
Desktop	√	√	√	√	√	√
Web	√	√	√	√	X	√
WebAssembly	*	√	√	X	X	√
Microcontroller	*	*	*	X	X	√
No-GIL	√	*	*	X	X	√
Register based VM	√	X	X	X	X	X
Off-side rule	X	√	X	X	X	X
xBase (Database DSL)	X	X	X	√	X	X

Research Gap (In General-Purpose VPLs)

- Few studies about using VPLs in large/complex system projects.
- Many GPVPLs are no longer under active development.
- No Self-hosting GPVPL.
- Importing textual code is not common/complete in most VPLs.
- Envision support for interactive visualization is limited. The Time Machine in PWCT doesn't support the Auto-Run feature.

Criteria	Scratch	Forms/3	Envision	Lava	PWCT	Proposed VPL (PWCT2)
Open Source	√	X	√	√	√	√
Portable	√	X	√	√	X	√
Rich Colors	√	X	√	X	X	√
Time Dimension	X	√	X	X	√	√
Auto-Run	√	√	X	X	X	√
Rich-Comments	X	X	√	X	X	√
Generate Ring Code	X	X	X	X	X	√
Interactive Visualization	X	X	√	X	X	√
Implementation using Ring	X	X	X	X	X	√
Import Ring Code	X	X	X	X	X	√
Self-hosting	X	X	*	X	X	√
Form Designer	X	√	X	√	√	√
Steps Tree/Blocks	√	X	X	√	√	√
Steps Tree/Blocks (DAD)	√	X	X	X	X	√
Play programs as movie	X	X	X	X	√	√
Supports OOP	X	X	√	√	√	√
Designed for Children	√	X	X	X	X	X
Just for Research	X	√	√	√	X	X

Thesis Presentation Outline

01

Introduction

02

**Problem Statement & Motivation &
Contributions**

03

Literature Review

04

Materials & Proposed Methods

05

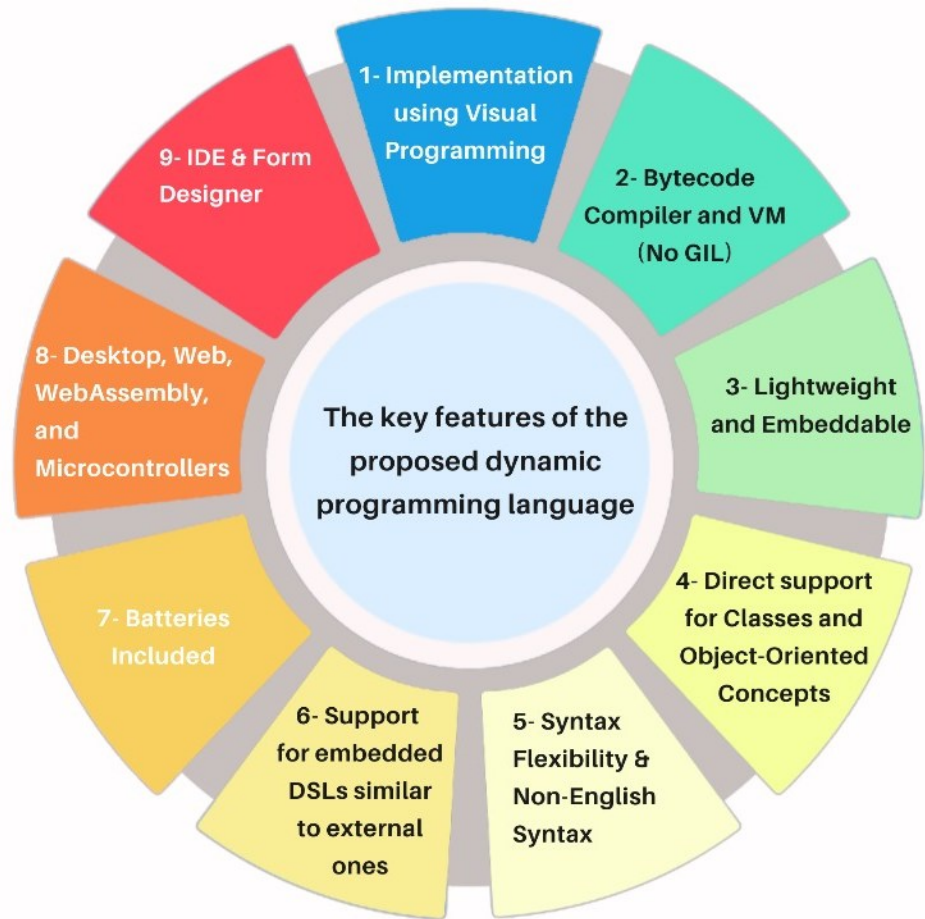
Experimental Results

06

Discussion and Comparisons

07

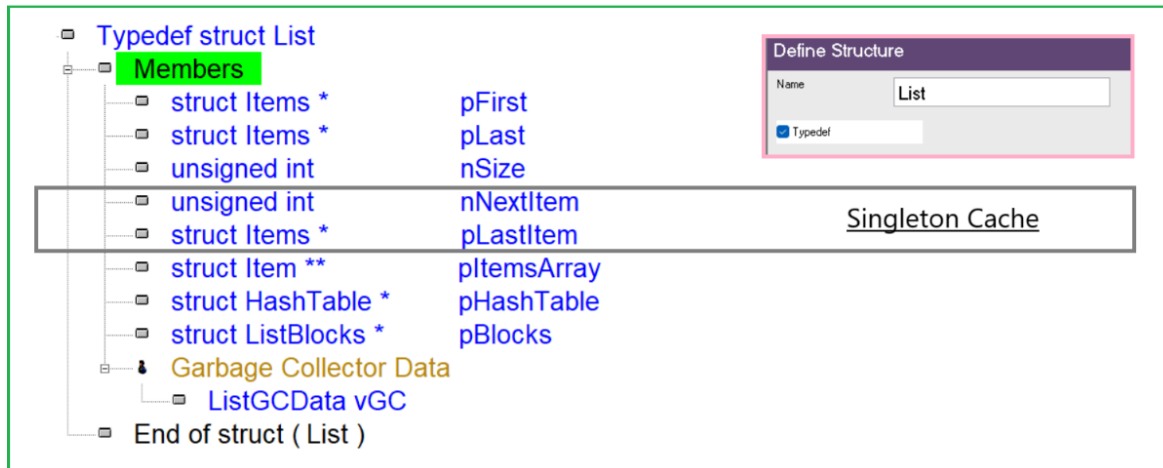
Conclusion



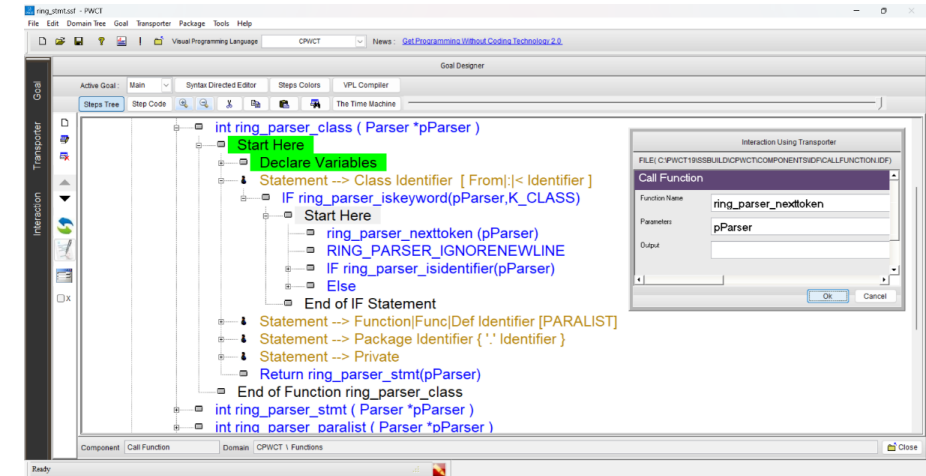
(A) The key features of the proposed dynamic language and environment.



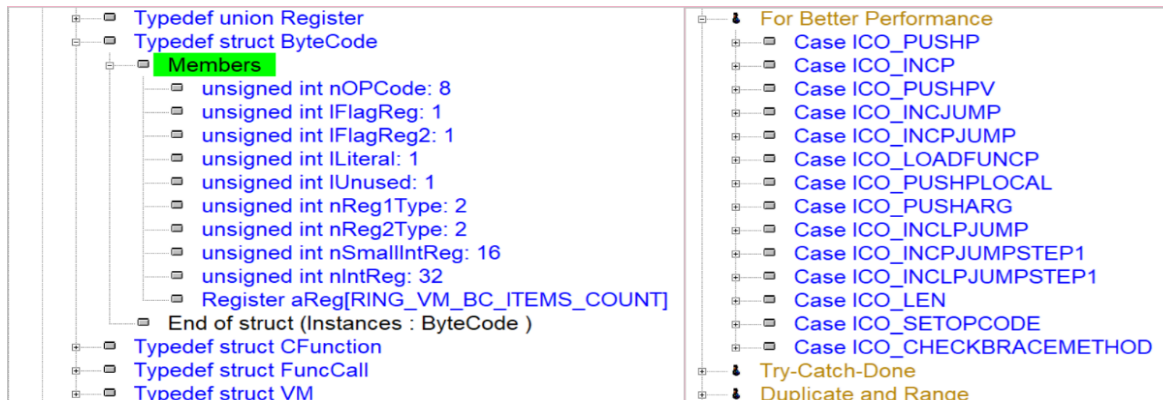
(B) The proposed system architecture.



(A) The List structure (Singleton cache).



(B) The Ring language grammar.



(C) Developing Ring Virtual Machine using PWCT.

- Single-pass compiler and optional modules.
- Optimized Ring Lists (Use C structures for critical features).
- Storing bytecode in a single continuous memory block.
- Writable long-byte code format.
- Avoiding the use of a global interpreter lock (No GIL).

C Compiler	Platform/OS (Target)
Watcom C/C++	MS-DOS
Microsoft Visual C/C++	Microsoft Windows
GNU C/C++	Ubuntu Linux
Clang	macOS
Android-clang	Android
Emscripten	WebAssembly
GNU ARM embedded toolchain	Raspberry Pi Pico

(A) C/C++ Compilers

Domain	C/C++ Libraries/Tools	Count
Terminal User Interface (TUI)	ConsoleColors and RogueUtil	2
Network and Security	LibCurl, Libuv, and OpenSSL	3
Web Servers	HTTPLib and Apache Web Server	2
Database	ODBC, SQLite, MySQL, and PostgreSQL	4
Games & multi-media	Allegro, LibSDL, RayLib and Tilengine	4
Graphics	OpenGL, FreeGLUT and StbImage	3
Graphical User Interface (GUI)	Qt, Libui, and NAppGUI	3
Common Files	MiniZip, PDFGen and CJSON	3
SDK for Specific Platforms	Android SDK and Raspberry Pi Pico SDK	2

(B) C/C++ Libraries

The screenshot shows a web browser window with the URL <https://ring-lang.github.io/web/tryringonline/project.html>. The interface includes a 'Source Code' editor on the left, a 'Run' button, and an 'Output' panel on the right. The 'Source Code' panel contains a list of keywords with their Arabic equivalents: 'ChangeRingKeyword put' (اطيع), 'ChangeRingKeyword get' (ادخل), 'ChangeRingKeyword if' (لو), 'ChangeRingKeyword elseif' (امالو), 'ChangeRingKeyword else' (عداذلك), and 'ChangeRingKeyword endif' (تمام). Below this, a code snippet is shown with a callout box stating: 'This Arabic syntax creates a program that prompts the user for their age and delivers a message based on the input'. The code snippet is:

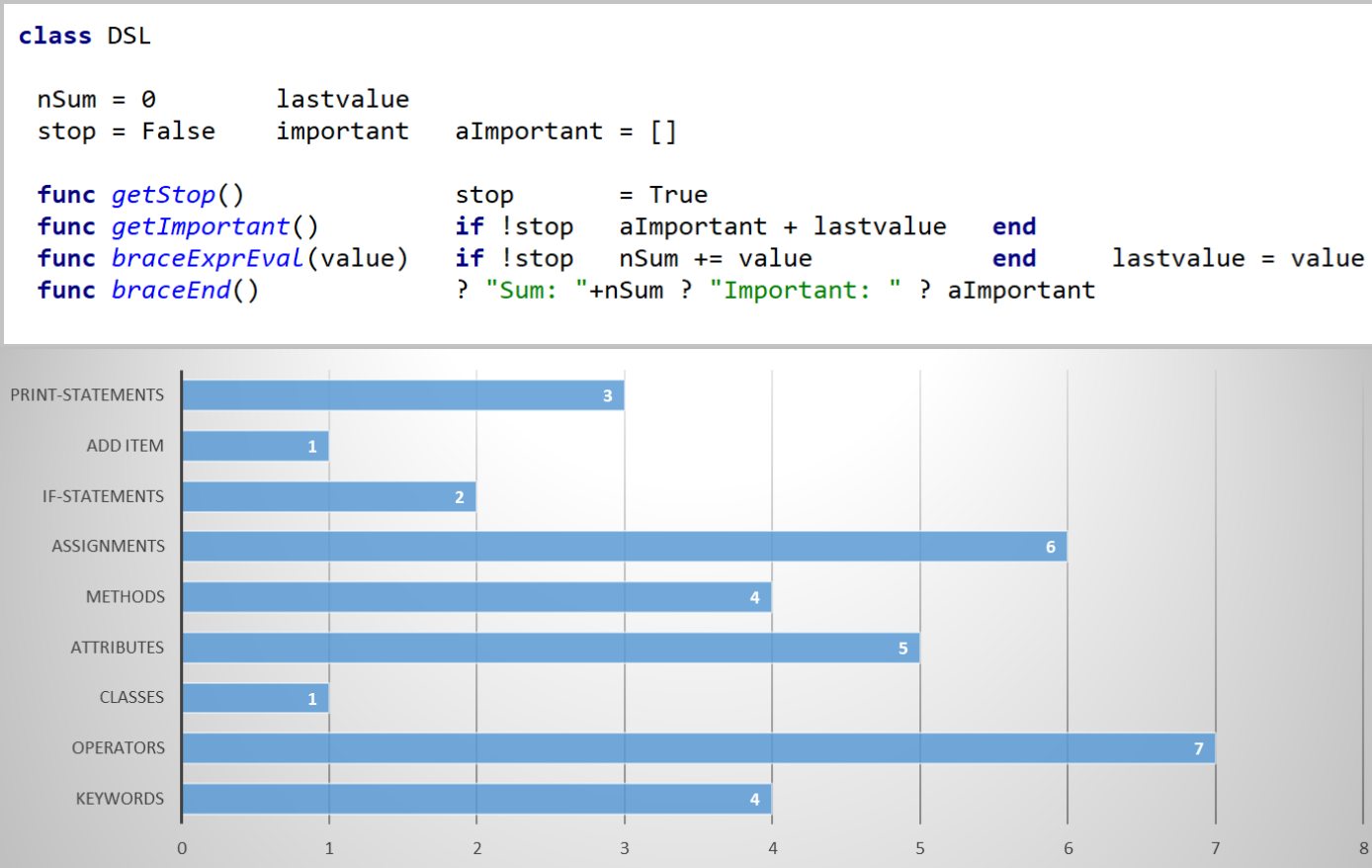
```
nl = سطر جديد
اطيع "أهلا ومرحبا بكم معنا" + سطر جديد
اطيع "من فضلك ... كم عمرك؟" + سطر جديد
ادخل العمر
العمر = العمر + 0
لو العمر > 10
اطيع "عمرك اقل من 10 سنوات" + سطر جديد
امالو العمر > 30
اطيع "عمرك اكبر من او يساوي 10 سنوات واقل من 30 عاما" + سطر جديد
عداذلك
اطيع "العمر اكبر من او يساوي 30 عاما" + سطر جديد
تمام
```

 The 'Output' panel shows the result of running the code: '30' and 'أهلا ومرحبا بكم معنا من فضلك ... كم عمرك؟ العمر اكبر من او يساوي 30 عاما'. Below the output, a 'Translation' box contains the English text: 'Welcome! Please, may I ask how old you are? 30 Age is greater than or equal to 30 years.' At the bottom of the 'Output' panel, there is an 'Input' field and a 'Send' button.

Arabic syntax within a WebAssembly application developed using Ring.

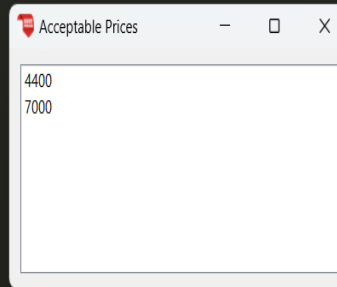
Usage (Ring Code)	Output
new DSL { 200 400 Important 50 600 Important 60 10 20 30 40 50 60 Stop 70 80 90 800 Important }	Sum: 1520 Important: 400 600

(A) Using the DSL class



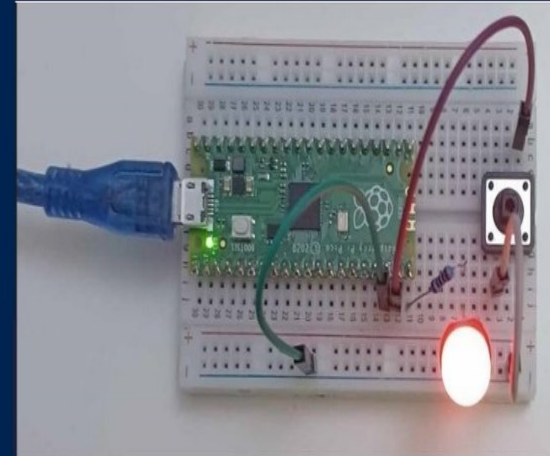
(B) Ring code to implement a simple domain-specific language.

```
1 new PickPrice {
2   Name      Price
3   Computer1 7000      Acceptable
4   Computer2 4400      Acceptable
5   Computer3 9000
6 }
7
8 class PickPrice from DSL
9   Acceptable func getAcceptable() getImportant()
10  func braceEnd
11    load "guilib.ring"
12    import System.GUI
13    app = new App {
14      win = new Window() {
15        setWindowTitle("Acceptable Prices")
16        resize(400,200) setWinIcon(self,"bestprice.png")
17        list = new ListWidget(win) { addList(Sort(this.aImportant)) }
18        setLayout( new VBoxLayout() { addWidget(list) } )
19        show()
20      }
21      exec()
22    }
23  func braceError
24
25 class DSL
```



(A) Extending our DSL using inheritance and the GUI library.

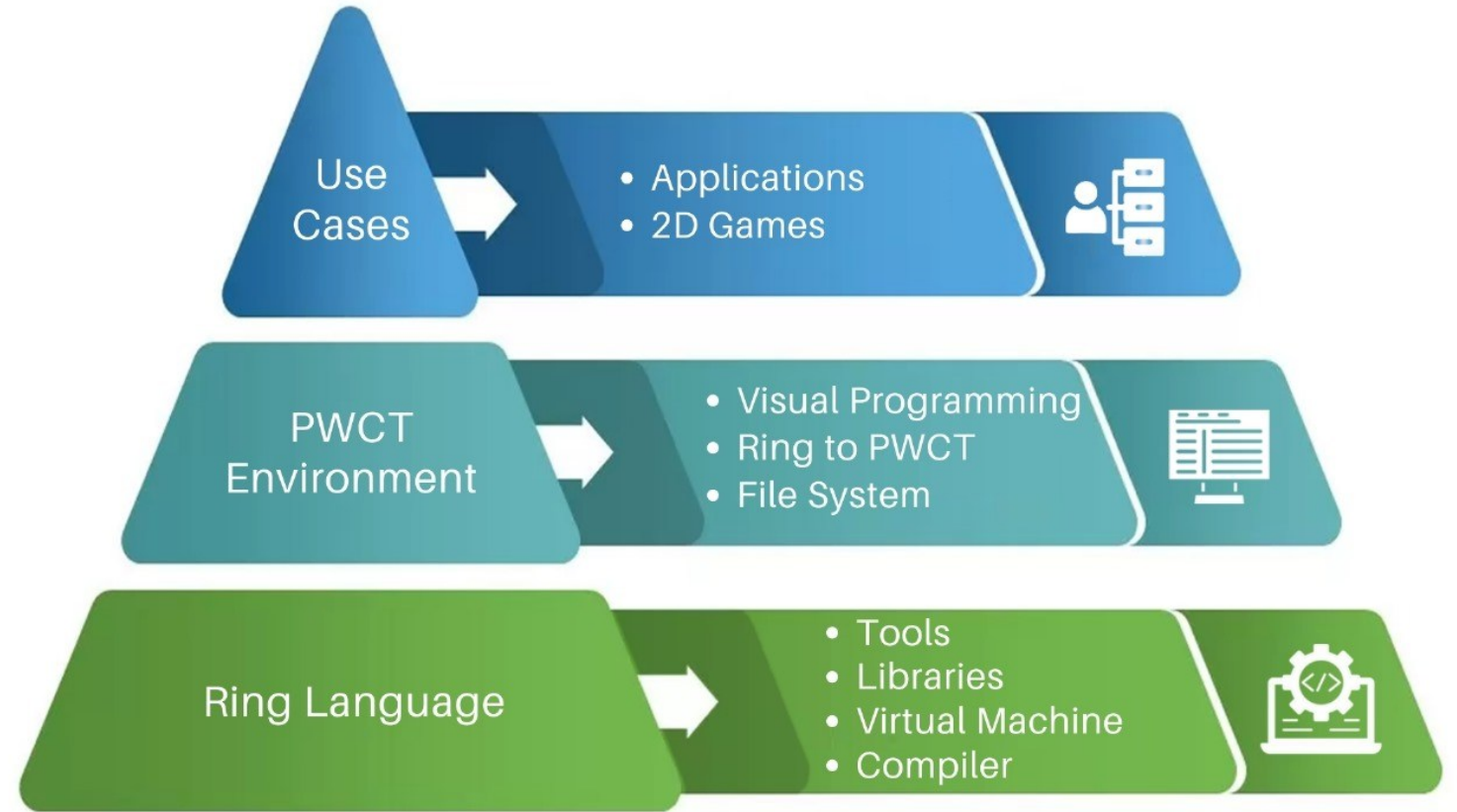
```
1 load "circuit.ring"
2
3 SWITCH_PIN = 14
4 LED_PIN    = 15
5
6 func main
7
8   Circuit {
9     LED {
10      Pin = PICO_DEFAULT_LED_PIN
11      Blink = True
12      Delay = 0.1
13    }
14    LEDSwitch {
15      Pin = SWITCH_PIN
16      LED {
17        Pin = LED_PIN
18        Blink = True
19        Delay = 3
20      }
21    }
22  }
```



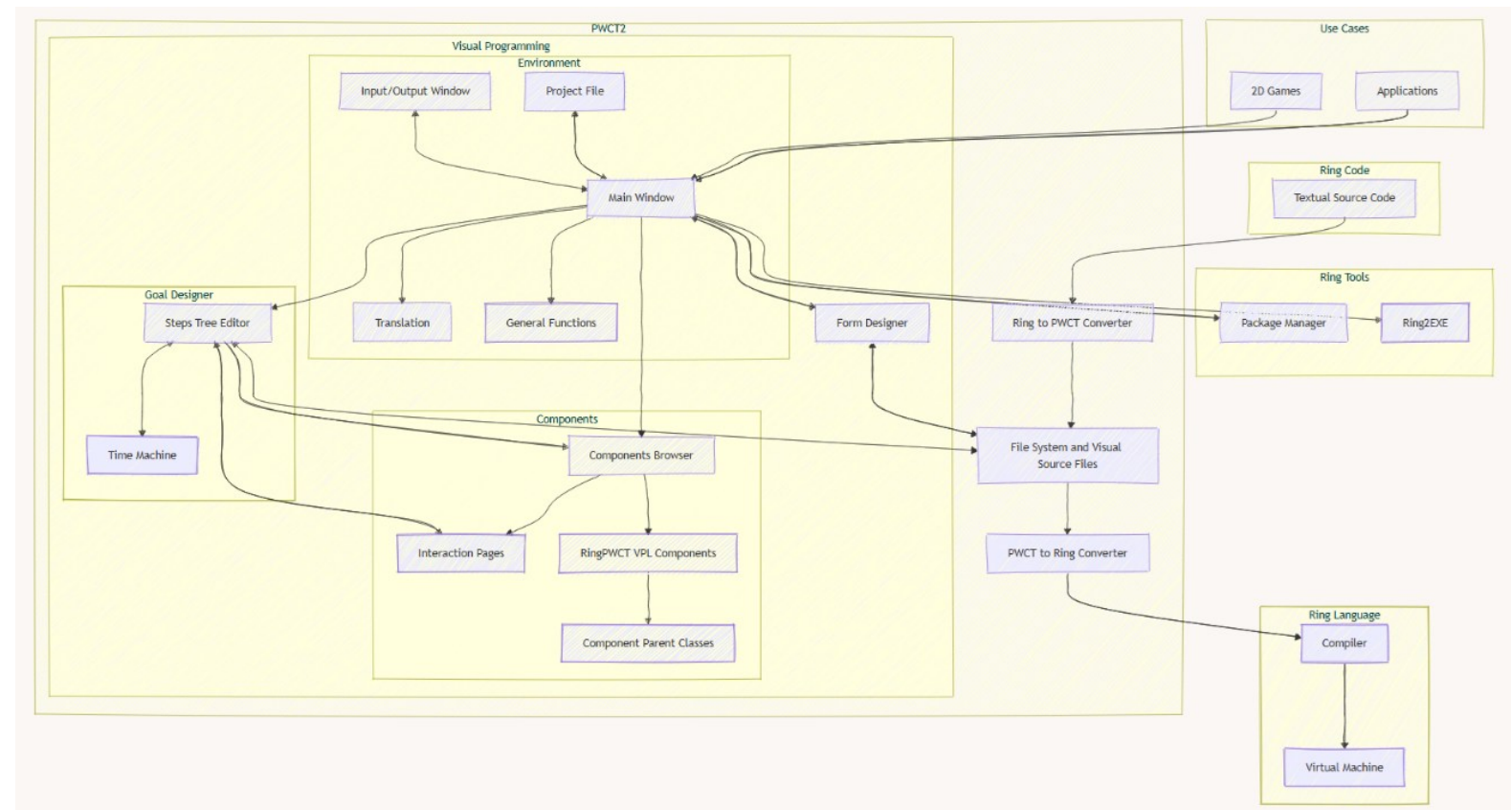
(B) Using Declarative Style in Ring for Raspberry Pi Pico programming.



(A) The key features of the proposed visual programming language.



(B) The proposed self-hosting visual programming language architecture.



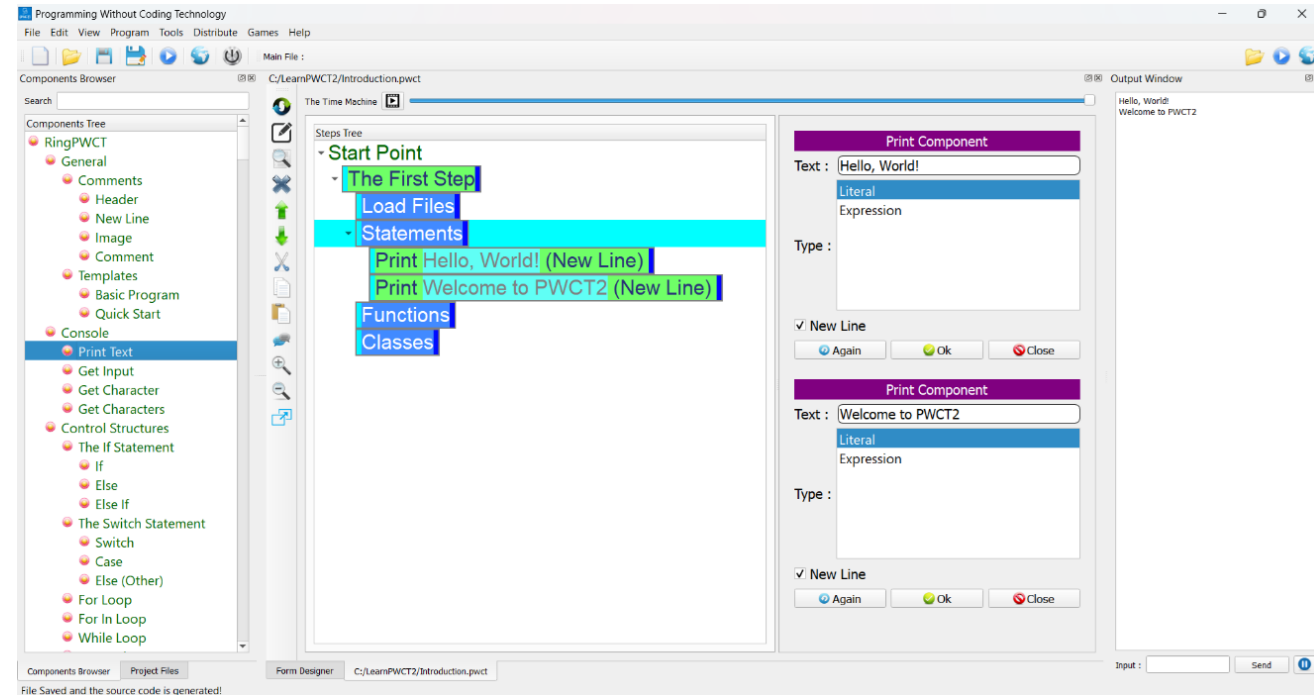
ID	Module	Files	LOC	Comment
1	Environment	5	2649	300
2	General Functions	9	524	122
3	Translation	3	584	20
4	Goal Designer	27	4908	1473
5	Components Browser	5	8876	70
6	VPL Components	1185	57,612	7167
7	Component Parent Classes	3	739	283
8	Form Designer	52	9487	312
9	File System	6	368	415
10	Tools	59	6484	546

(A) The PWCT2 system developed using the Ring programming language.

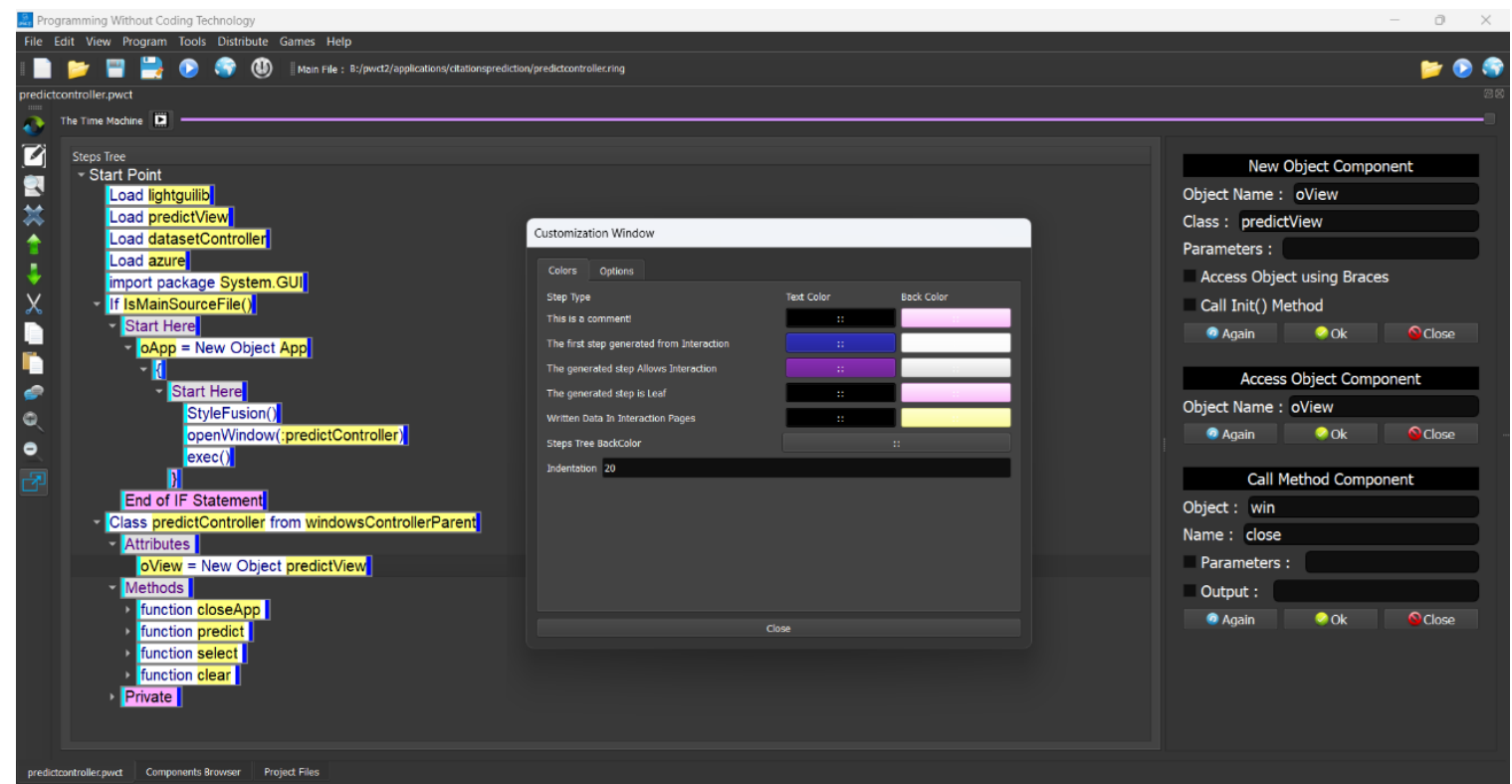
(B) System Modules.

Materials & Proposed Methods

ID	Domain	Components	Example
1	General	6	Quick Start
2	Console	4	Print Text
3	Control Structures	13	For-In Loop
4	Variables and Operators	17	Assignment
5	Functions	3	Define Function
6	Program Structure	2	Load Source File
7	Lists	15	New Empty List
8	Strings	16	Get String Length
9	Date and Time	7	Add Days
10	Check Data Type	3	Check Character
11	Math	1	Math Functions
12	Files	29	Read File to String
13	System	12	Get System Variable
14	Dynamic Code	3	Eval
15	Database	34	ODBC Connect
16	Security and Internet Functions	11	Download
17	Object-Oriented Programming	10	Define Class
18	Functional Programming	3	Anonymous Function
19	Reflection	29	Globals Info
20	Standard Library	71	Stack Class
21	Web Library	12	WL WebPage Class
22	LibCurl Library	5	LibCurl Easy Init
23	GUI	88	Window Class



(B) PWCT2 uses a main window and dock-able windows.



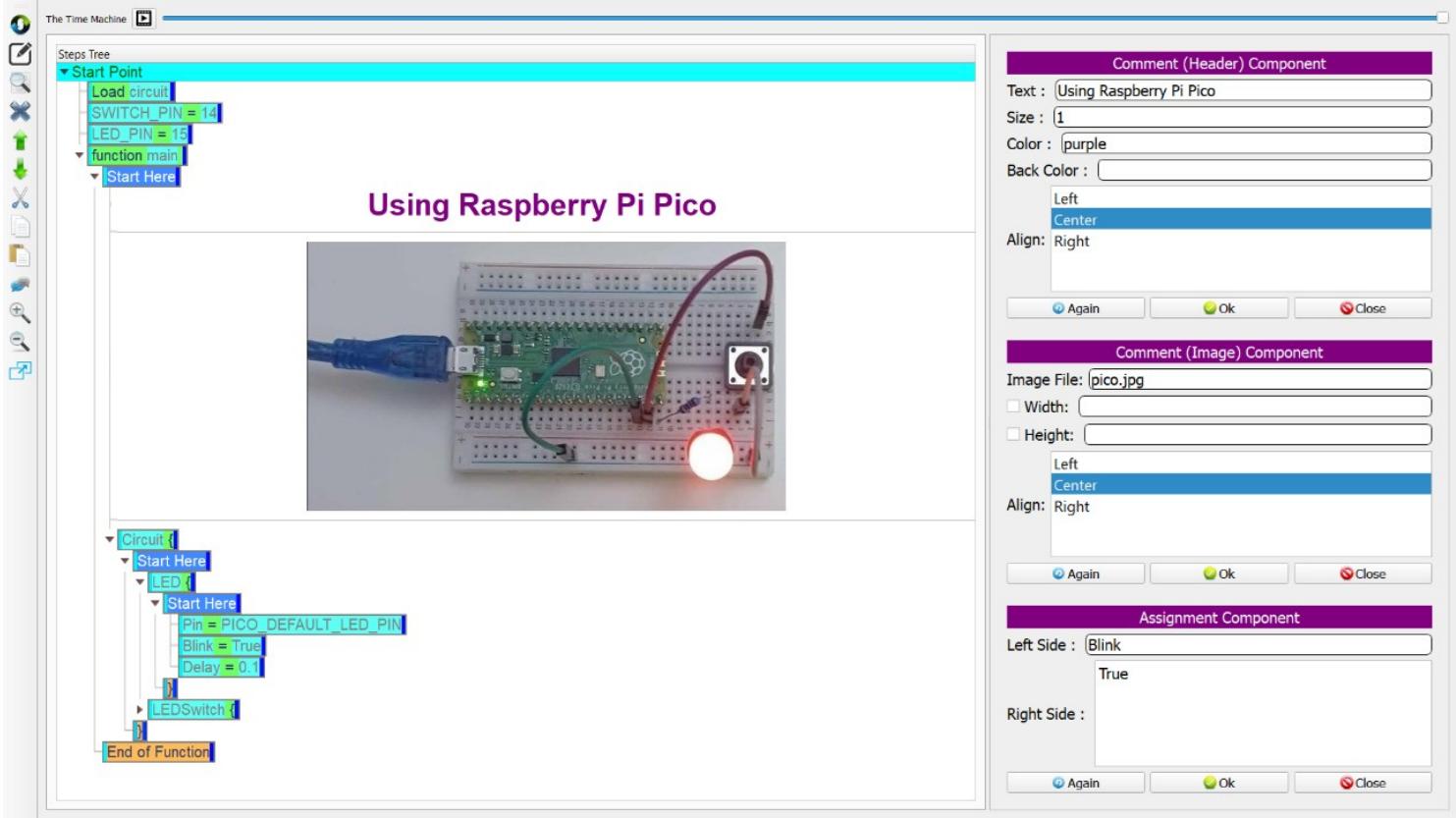
(A) Using the Customization window to select the Steps Tree colors.

ID	Step Type	Description
1	Start Point	The program root (one for each visual source file)
2	Comment	Just a comment and does nothing during runtime
3	First	The first step generated by the component
4	Allows Interaction	The step could include sub steps
5	Leaf	The step cannot include sub steps

(B) Step Type

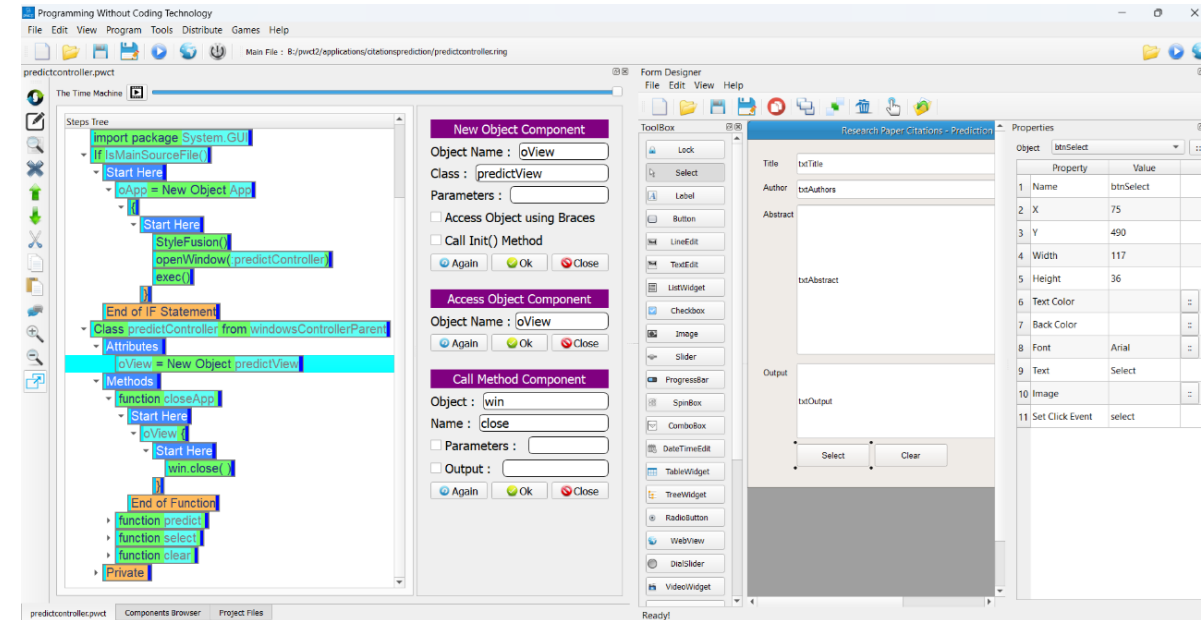
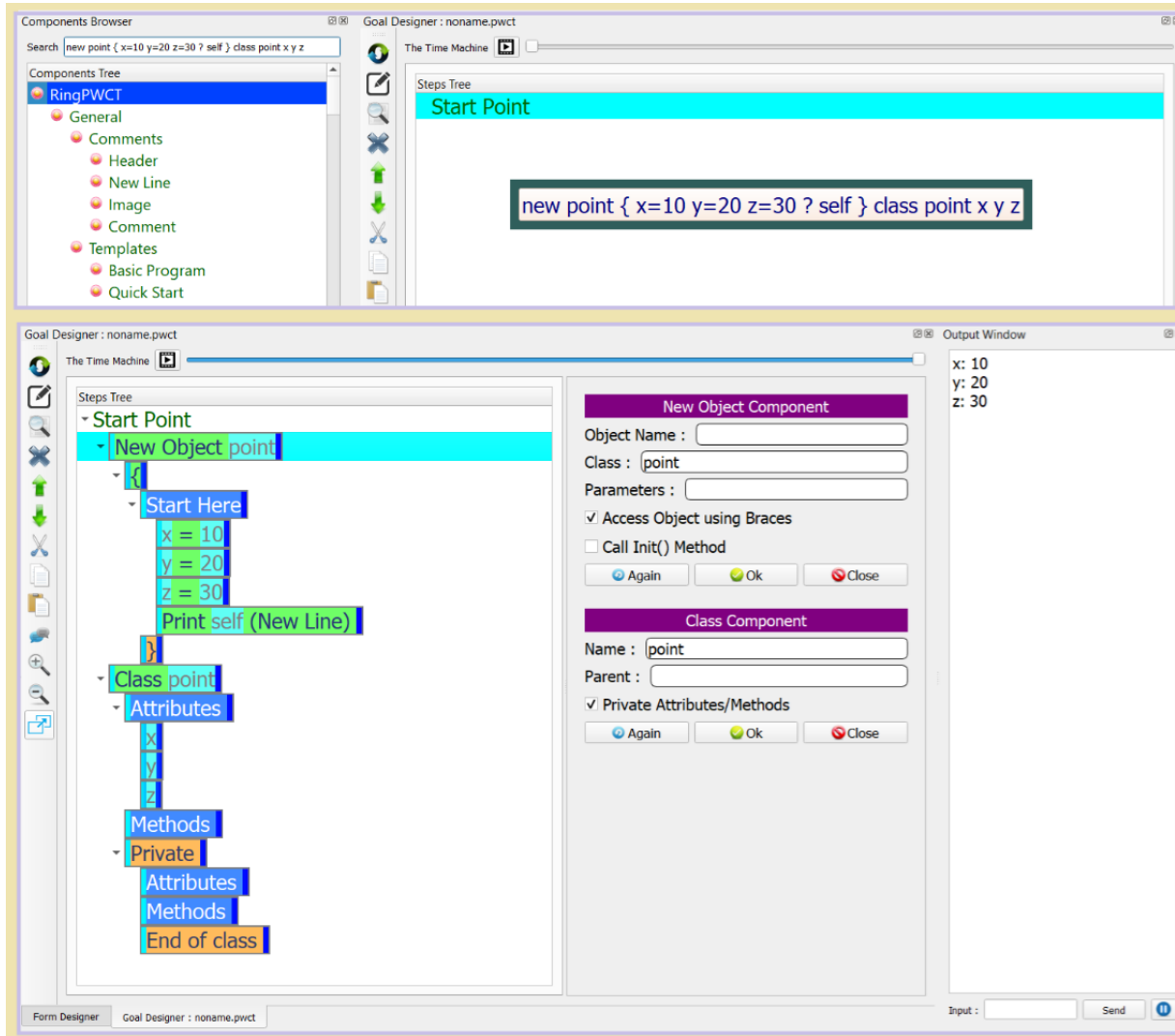


(A) Using the Time Machine and the Auto-Run feature.



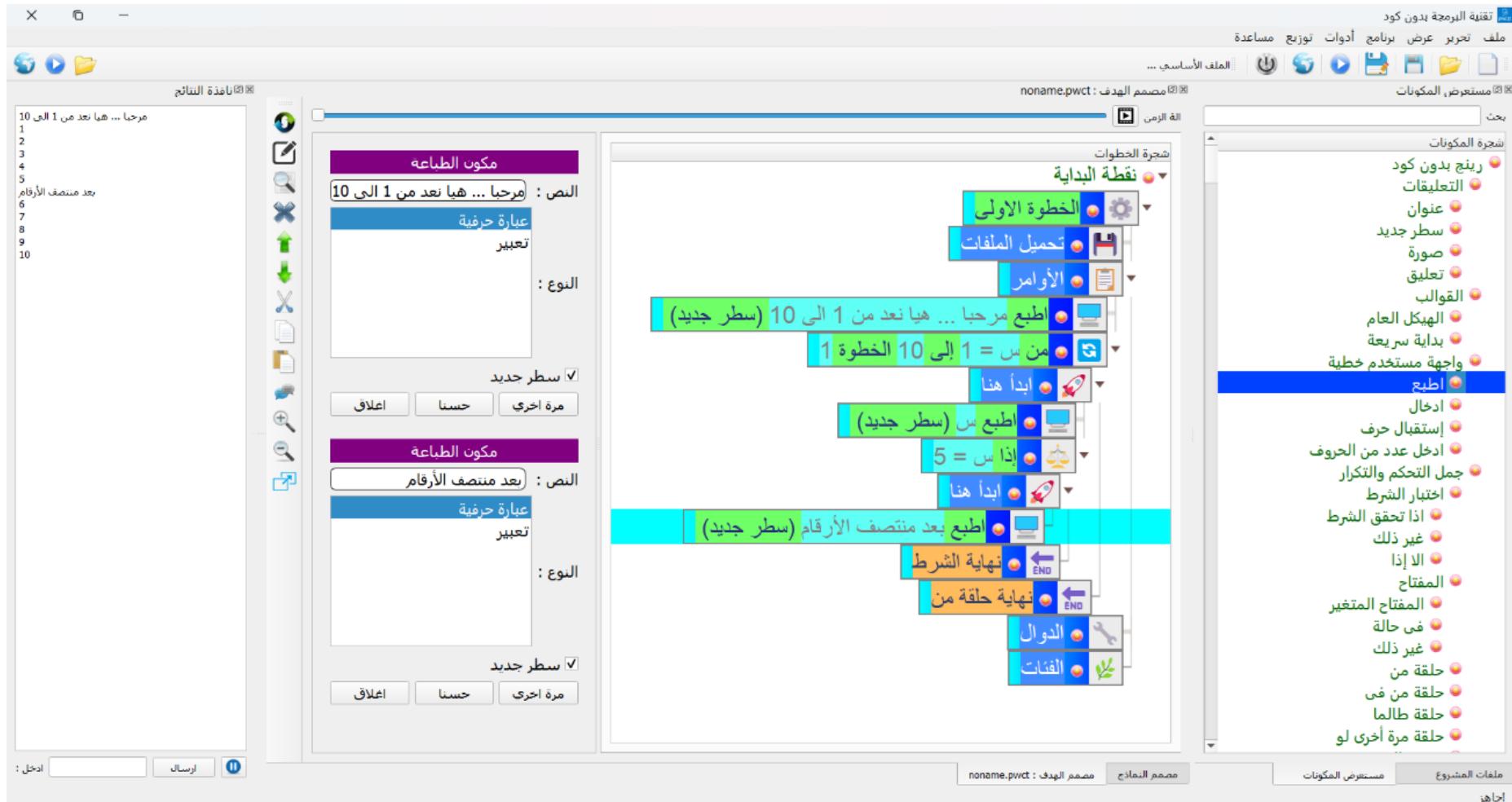
(B) Using rich comments (Lines, Images, and Headers).

Materials & Proposed Methods



(B) Using the PWCT2 Form Designer.

(A) Interactive Textual-to-Visual code conversion (Ring2PWCT).
Materials & Proposed Methods



Arabic translation for the PWCT2 visual programming language.

Thesis Presentation Outline

01

Introduction

02

**Problem Statement & Motivation &
Contributions**

03

Literature Review

04

Materials & Proposed Methods

05

Experimental Results

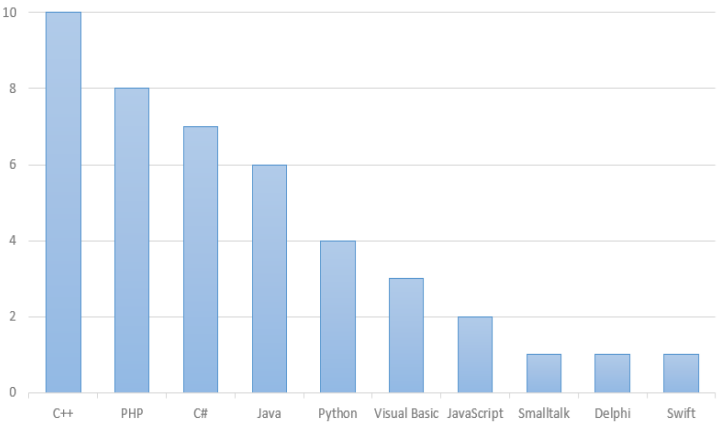
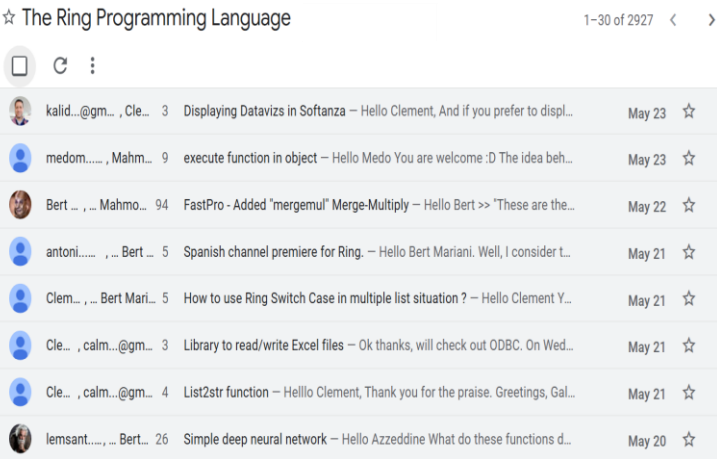
06

Discussion and Comparisons

07

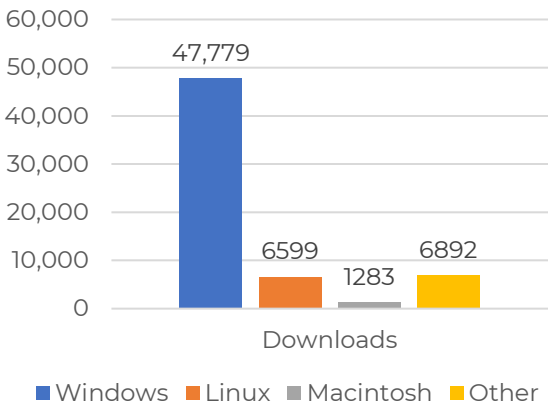
Conclusion

Experimental Results

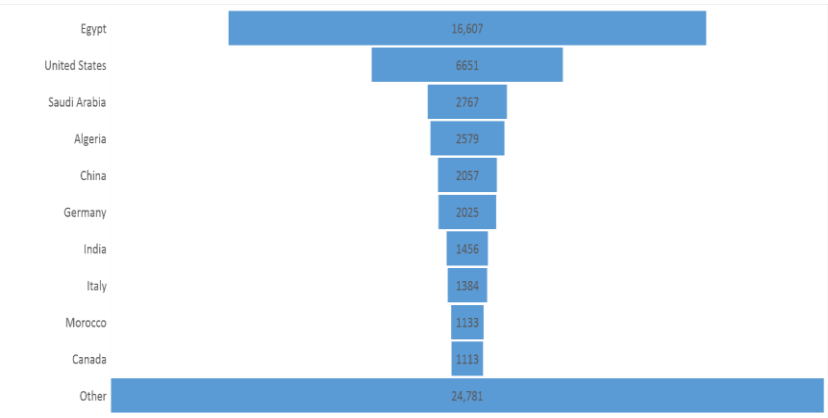


Ref.	Type	Domain
[166]	Research Paper	Front-end apps for ML Models
[167]	Research Paper	Front-end apps for ML Models
[91]	Printed Book (USA)	Games Development
[152]	Steam Game	Games Development
[168]	Research Paper	Text/Data Processing apps
[169]	Printed Book (Egypt)	Text/Data Processing apps
[170]	YouTube Videos	Desktop/Web development
[171]	Research Paper	LLMs Training

(A) Ring Group



(B) Early users and the language used.



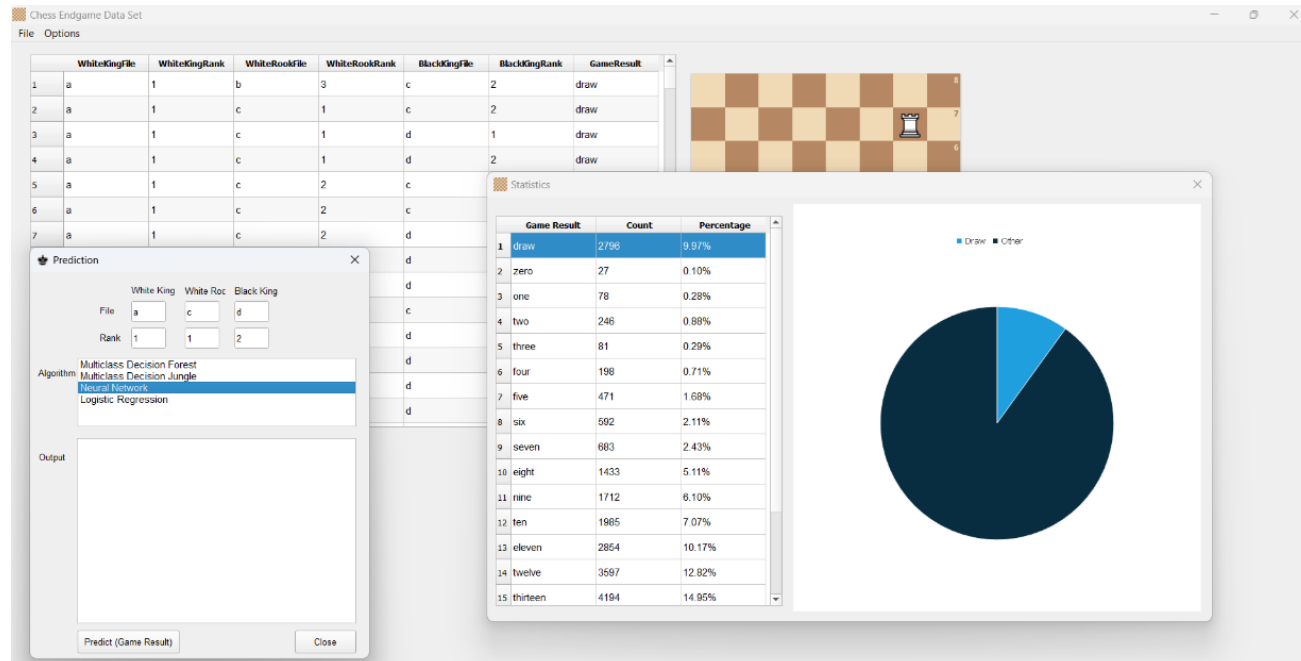
(C) Use-Cases

Variable	Value
Male	70
Female	6
Completed less than two lessons	20
Completed more than one lesson	56
Completed the course	23
Contributors	2

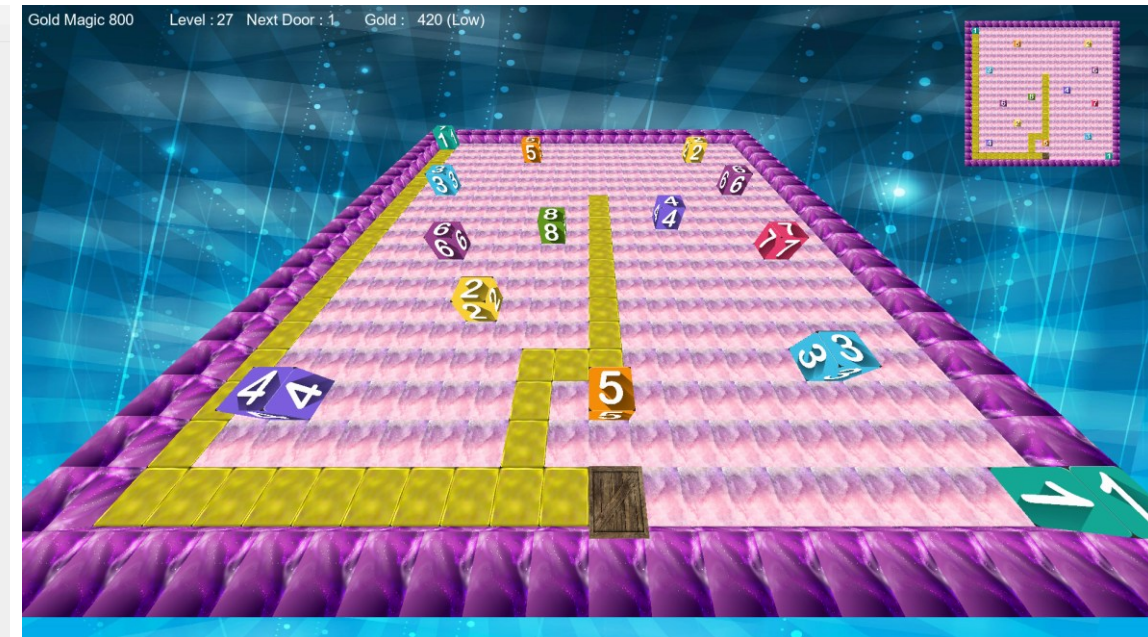
(D) Ring downloads statistics

(E) Ring Arabic Course (18 Lessons)

Experimental Results



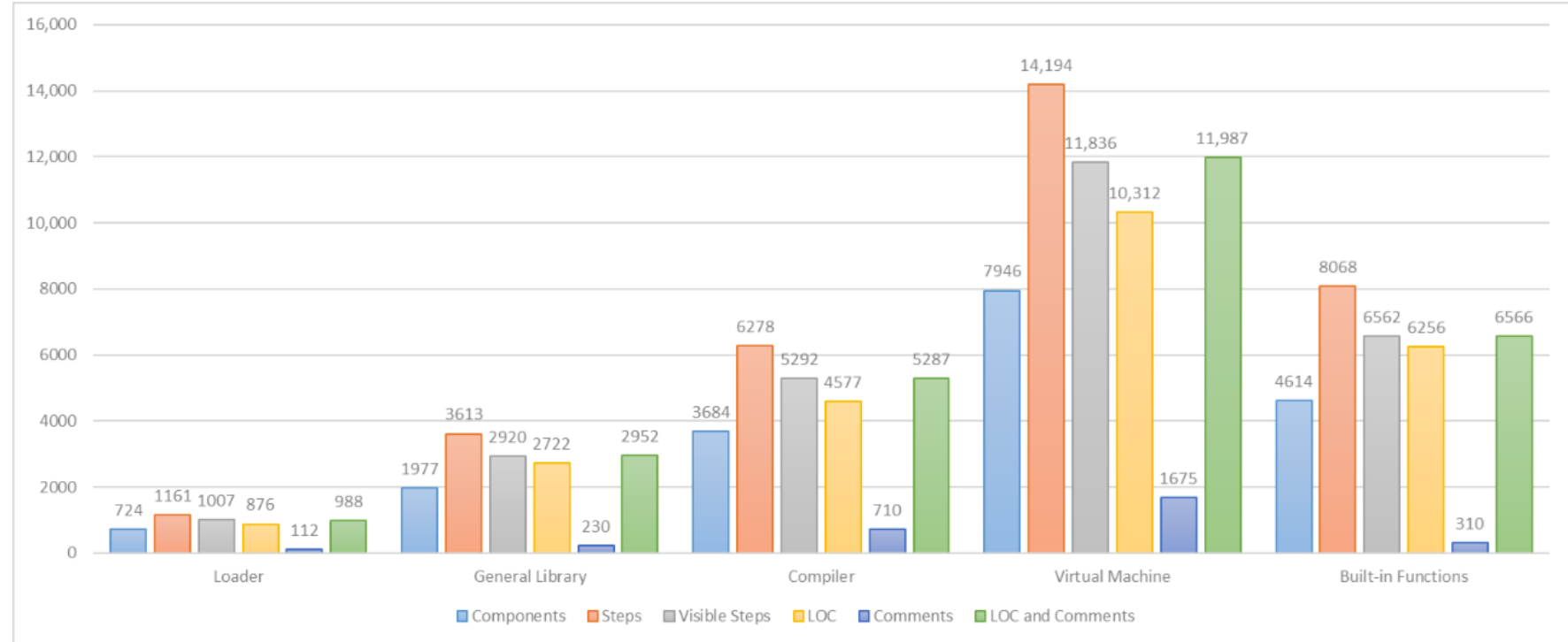
(A) A GUI application developed using the Ring language.



(B) The GoldMagic800 game—A puzzle game developed using RingAllegro.

Criteria	Total
Modules	5
Visual Source Files	43
Storage Size (MB)	278.95
Memory (MB)	1350.6
Visual Components	18,945
Steps	33,314
Steps (Visible)	27,617
Lines of Code (LOC)	24,743
Comments	3037
LOC including comments	27,780

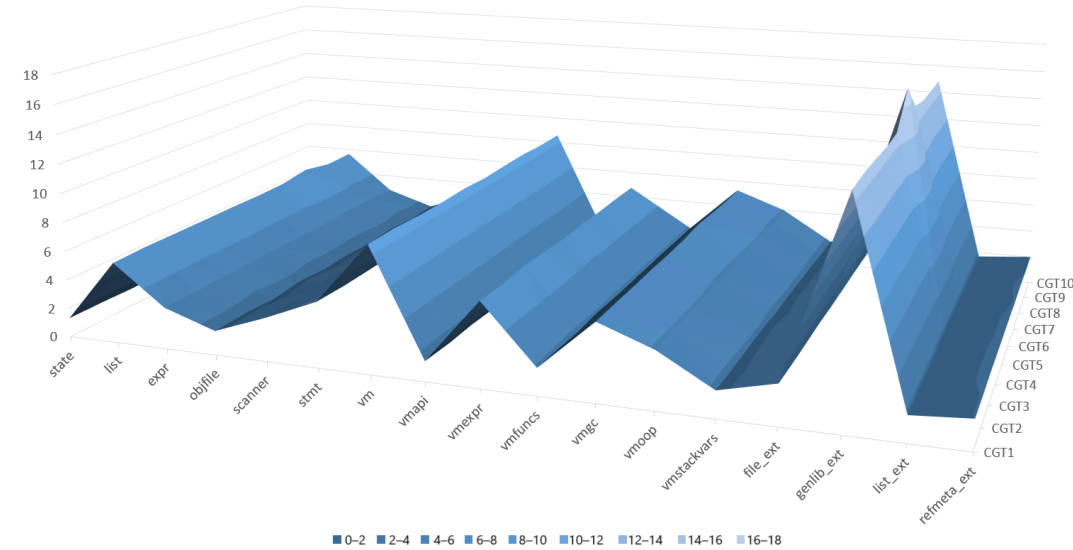
(A) Ring Compiler/VM



(B) Visual implementation size for each module.

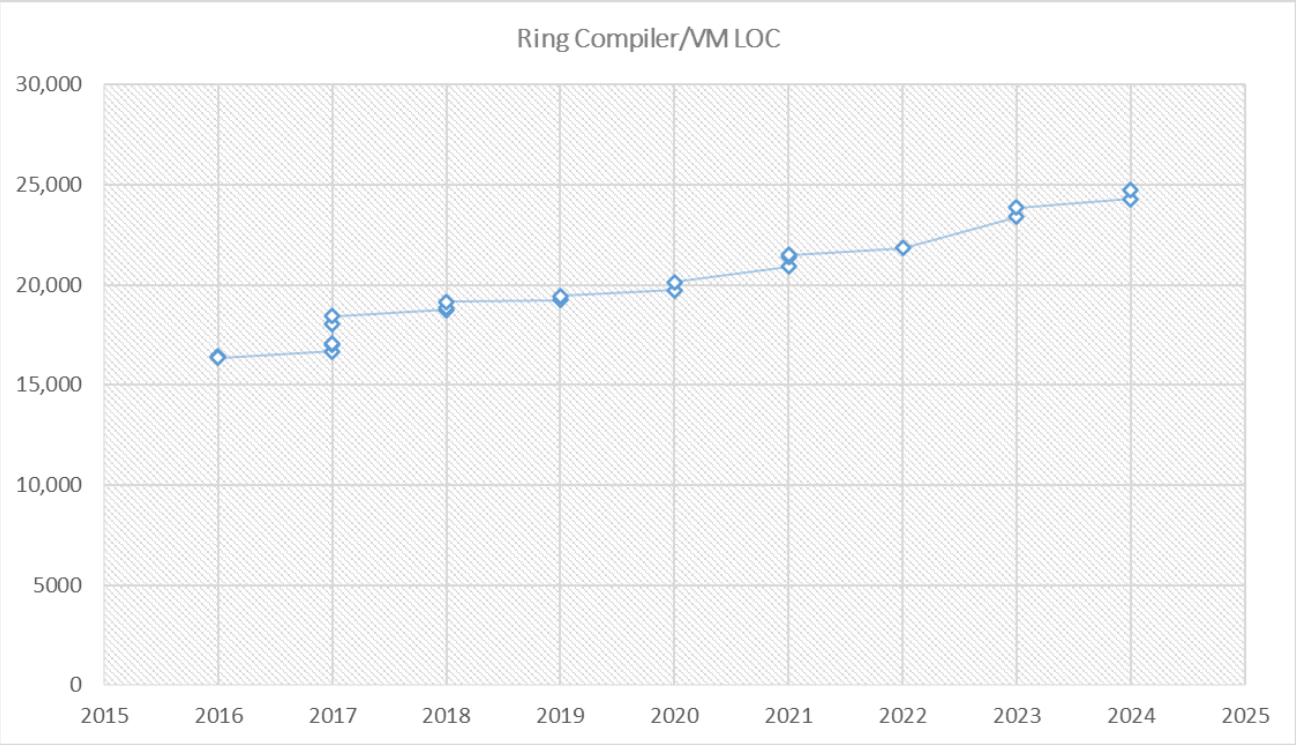
Experimental Results

File Name	LT1	CGT1	LT2	CGT2	LT3	CGT3	LT4	CGT4	LT5	CGT5	LT6	CGT6	LT7	CGT7	LT8	CGT8	LT9	CGT9	LT10	CGT10
ring	0.141	0.208	0.126	0.204	0.126	0.213	0.13	0.201	0.13	0.213	0.119	0.205	0.12	0.206	0.13	0.188	0.11	0.192	0.11	0.207
state	0.518	1.307	0.505	1.288	0.519	1.228	0.5	1.318	0.51	1.321	0.486	1.255	0.49	1.241	0.49	1.286	0.505	1.227	0.534	1.259
general	0.22	0.315	0.202	0.283	0.22	0.299	0.2	0.298	0.21	0.299	0.204	0.313	0.22	0.298	0.22	0.299	0.204	0.289	0.204	0.314
hashtable	0.173	0.211	0.142	0.205	0.157	0.214	0.15	0.205	0.13	0.205	0.141	0.22	0.14	0.203	0.14	0.205	0.125	0.204	0.142	0.204
item	0.236	0.385	0.22	0.346	0.219	0.375	0.22	0.377	0.22	0.377	0.221	0.393	0.22	0.383	0.22	0.365	0.219	0.394	0.204	0.371
items	0.031	0.017	0.037	0.015	0.032	0.016	0.05	0.016	0.05	0.016	0.032	0.016	0.03	0.031	0.05	0.031	0.047	0.006	0.047	0.016
list	1.002	5.564	0.959	5.646	0.958	5.633	0.96	5.627	0.96	5.632	0.989	5.605	0.98	5.641	0.99	5.936	0.982	5.554	0.975	5.612
string	0.252	0.457	0.236	0.472	0.252	0.489	0.25	0.489	0.24	0.471	0.252	0.486	0.24	0.485	0.24	0.488	0.236	0.481	0.236	0.487
hashlib	0.063	0.015	0.062	0.016	0.041	0.015	0.05	0.016	0.05	0.016	0.047	0.016	0.05	0.015	0.05	0.016	0.047	0.016	0.047	0.015
codegen	0.377	0.865	0.346	0.91	0.345	0.834	0.35	0.896	0.35	0.928	0.346	0.871	0.35	0.881	0.35	0.906	0.346	0.853	0.363	0.878
expr	0.999	2.864	1.043	2.818	1.024	2.825	1.04	2.854	1.04	2.84	1.034	2.77	1.04	2.798	1.06	2.854	1.035	2.843	1.039	2.778
objfile	0.441	1.668	0.439	1.675	0.441	1.528	0.44	1.725	0.44	1.555	0.452	1.654	0.44	1.557	0.44	1.718	0.441	1.551	0.455	1.671
parser	0.204	0.394	0.204	0.393	0.189	0.393	0.19	0.401	0.2	0.401	0.188	0.408	0.19	0.393	0.19	0.382	0.189	0.377	0.188	0.377
scanner	0.846	2.975	0.737	3.011	0.726	3.028	0.75	3.026	0.72	2.981	0.731	2.986	0.74	3.006	0.74	2.982	0.74	3.01	0.736	3.025
stmt	0.956	4.468	0.942	4.402	0.95	4.44	0.86	4.5	0.93	4.453	0.943	4.44	0.99	4.505	0.88	4.439	0.897	4.661	0.972	4.437
vm	1.102	8.629	1.082	8.681	1.099	8.573	1.1	8.561	1.13	8.625	1.184	8.525	1.17	8.587	1.15	8.629	1.135	8.547	1.102	8.576
vmapi	0.471	1.377	0.472	1.321	0.471	1.401	0.5	1.34	0.49	1.371	0.47	1.385	0.47	1.358	0.5	1.403	0.47	1.395	0.488	1.338
vmduprange	0.063	0.041	0.047	0.032	0.047	0.032	0.05	0.04	0.06	0.045	0.063	0.039	0.06	0.045	0.06	0.048	0.063	0.047	0.47	0.48
vmerror	0.094	0.16	0.11	0.141	0.11	0.14	0.11	0.14	0.11	0.141	0.11	0.155	0.11	0.142	0.11	0.141	0.11	0.141	0.11	0.141
vmeval	0.157	0.327	0.15	0.362	0.137	0.357	0.16	0.366	0.16	0.363	0.157	0.362	0.16	0.345	0.14	0.347	0.157	0.362	0.157	0.361
vmexit	0.079	0.063	0.078	0.062	0.094	0.048	0.08	0.063	0.08	0.069	0.079	0.063	0.08	0.048	0.08	0.062	0.079	0.063	0.087	0.058
vmexpr	1.098	5.743	1.161	5.725	1.144	5.842	1.2	5.805	1.14	5.7	1.163	5.772	1.18	5.722	1.16	5.754	1.165	5.728	1.161	1.754
vmext	0.106	0.026	0.11	0.016	0.094	0.015	0.09	0.031	0.1	0.015	0.094	0.016	0.1	0.016	0.11	0.016	0.094	0.015	0.091	0.016
vmfuncs	0.565	1.84	0.549	1.818	0.565	1.748	0.55	1.868	0.57	1.811	0.582	1.815	0.57	1.83	0.58	1.85	0.549	1.826	0.55	1.838
vmgc	0.977	5.235	1.084	5.252	1.034	5.243	1.04	5.266	1.05	5.237	1.067	5.269	1.05	5.413	1.06	5.305	1.067	5.213	1.086	5.255
vmjump	0.126	0.112	0.11	0.111	0.11	0.109	0.11	0.103	0.11	0.127	0.11	0.11	0.09	0.11	0.11	0.11	0.11	0.109	0.126	0.122
vmlists	0.442	0.815	0.471	0.784	0.455	0.849	0.44	0.866	0.45	0.879	0.458	0.843	0.44	0.879	0.45	0.882	0.449	0.878	0.465	0.817
vmoop	0.898	3.888	0.925	3.923	0.879	3.937	0.93	3.922	0.97	3.862	0.926	3.873	0.93	3.93	0.86	3.95	0.911	3.899	0.879	3.968
vmperformance	0.162	0.215	0.16	0.22	0.173	0.226	0.16	0.196	0.17	0.22	0.161	0.229	0.17	0.236	0.17	0.206	0.172	0.204	0.173	0.22
vmstackvars	0.551	1.804	0.534	1.777	0.533	1.819	0.55	1.851	0.53	1.772	0.534	1.819	0.55	1.806	0.54	1.8	0.534	1.757	0.549	1.775
vmstate	0.283	0.725	0.282	0.738	0.283	0.66	0.27	0.676	0.28	0.723	0.283	0.733	0.28	0.662	0.28	0.674	0.28	0.724	0.268	0.707
vmstrindex	0.047	0.007	0.047	0.016	0.032	0.015	0.05	0.016	0.05	0.016	0.031	0.016	0.05	0.016	0.05	0.007	0.047	0.015	0.047	0.022
vmthread	0.11	0.127	0.094	0.142	0.095	0.142	0.09	0.141	0.11	0.142	0.094	0.143	0.1	0.149	0.1	0.142	0.094	0.142	0.094	0.142
vmtrycatch	0.032	0.016	0.032	0.016	0.031	0.006	0.03	0.016	0.04	0.01	0.017	0.004	0.03	0.015	0.03	0.015	0.031	0.004	0.016	0.016
vmvars	0.487	0.911	0.503	0.894	0.526	0.863	0.49	0.867	0.5	0.898	0.511	0.92	0.5	0.891	0.49	0.839	0.488	0.879	0.502	0.895
vminfo_ext	0.284	0.371	0.283	0.378	0.283	0.362	0.3	0.36	0.28	0.366	0.277	0.362	0.28	0.363	0.3	0.348	0.282	0.388	0.267	0.376
dll_ext	0.205	0.042	0.188	0.047	0.171	0.048	0.19	0.047	0.19	0.047	0.172	0.047	0.19	0.039	0.19	0.043	0.178	0.047	0.174	0.047
file_ext	0.81	2.704	0.75	2.669	0.802	2.666	0.77	2.713	0.82	2.656	0.747	2.661	0.76	2.675	0.8	2.661	0.8	2.651	0.754	2.732
genlib_ext	2.545	14.66	2.429	14.78	2.431	14.751	2.43	14.66	2.47	14.78	2.426	17.04	2.43	15.113	2.44	14.77	2.339	14.81	2.415	14.84
list_ext	0.439	1.728	0.456	1.759	0.445	1.73	0.44	1.762	0.44	1.778	0.44	1.761	0.44	1.71	0.46	1.717	0.439	1.735	0.456	1.683
math_ext	0.314	0.79	0.283	0.816	0.299	0.829	0.31	0.816	0.29	0.769	0.289	0.756	0.28	0.815	0.3	0.8	0.283	0.786	0.314	0.774
os_ext	0.33	0.642	0.329	0.647	0.33	0.629	0.31	0.598	0.3	0.597	0.314	0.598	0.31	0.597	0.33	0.643	0.298	0.603	0.314	0.646
refmeta_ext	0.611	2.01	0.595	2.015	0.596	2.062	0.62	2.055	0.59	2.038	0.596	2.047	0.6	2.014	0.6	2.043	0.628	1.993	0.698	2.013

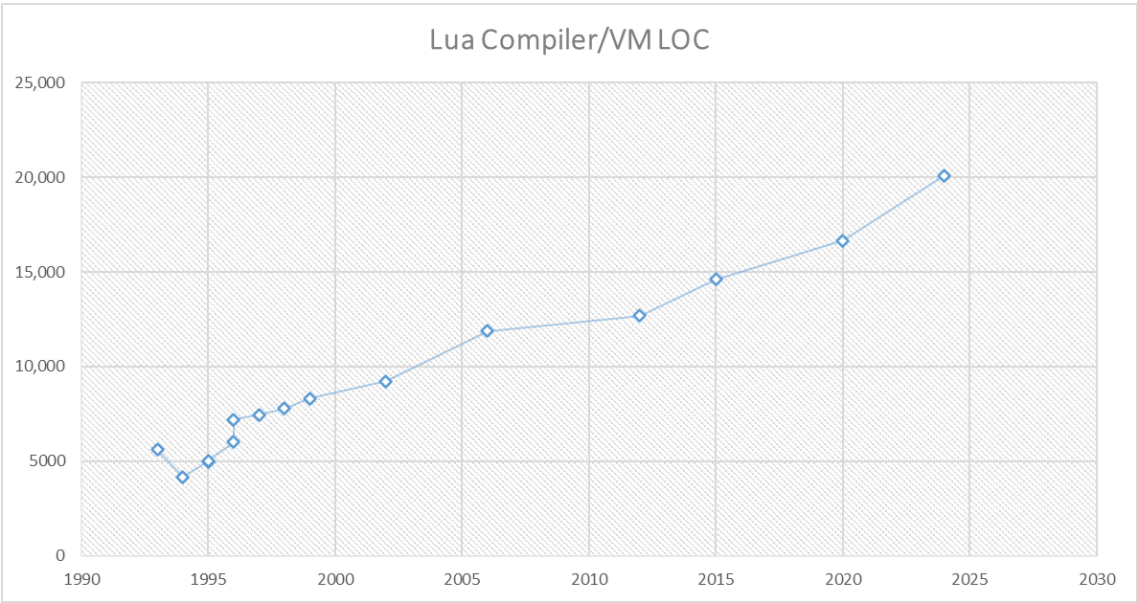


(B) Code generation time (CGT) for large visual source files.

(A) The loading time (LT) and code generation time (CGT).



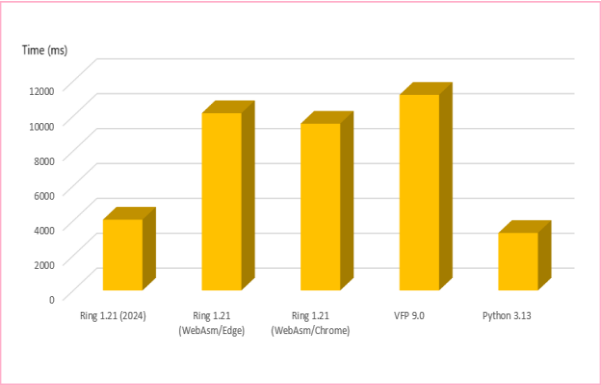
(A) Generated code size for Ring Compiler/VM from 2016 to 2024.



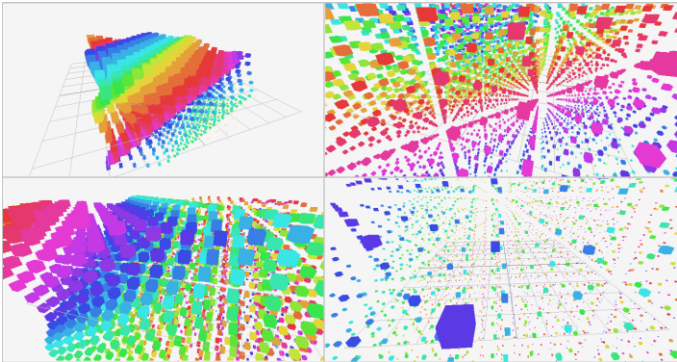
(B) Code size for Lua Compiler/VM from 1993 to 2024.

Language	Period	Impl.	LOC (FR)	LOC (LR)	Growth
Ring	2016–2024	C	16,402	24,743	51%
mRuby	2014–2024	C	18,134	23,742	31%
Squirrel	2004–2022	C++	9311	13,991	50%
Lua	1993–2024	C	5603	20,081	258%

(C) Lightweight TPLs.



(A) Function call (100 M) benchmark



(B) Waving Cubes Sample

Language	FPS (Min)	FPS (Max)
C	470	480
Ring 1.21	161	170
Python 3.13	80	85
Ring 1.20	33	40

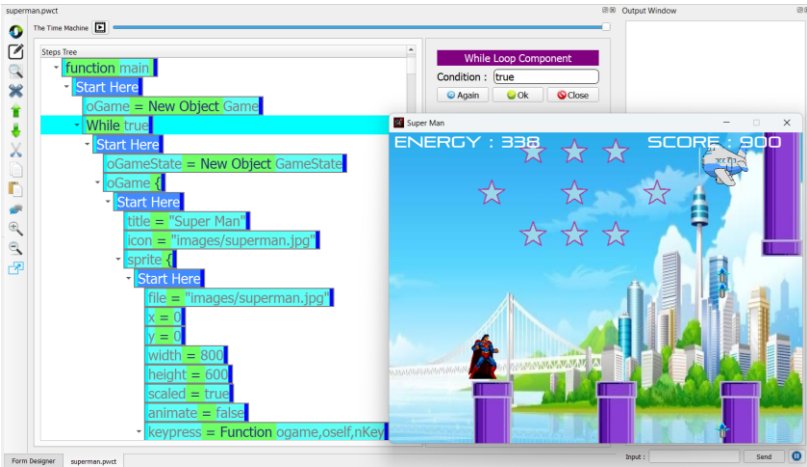
(C) Waving Cubes Performance

Variable	Value
Extension	RingQt
Configuration Files (Input)	439
Input Size	478 KB
Generated Files	197
Generated Lines of Code	211,174
Output Size	6.27 MB
Processing Time	3420 ms

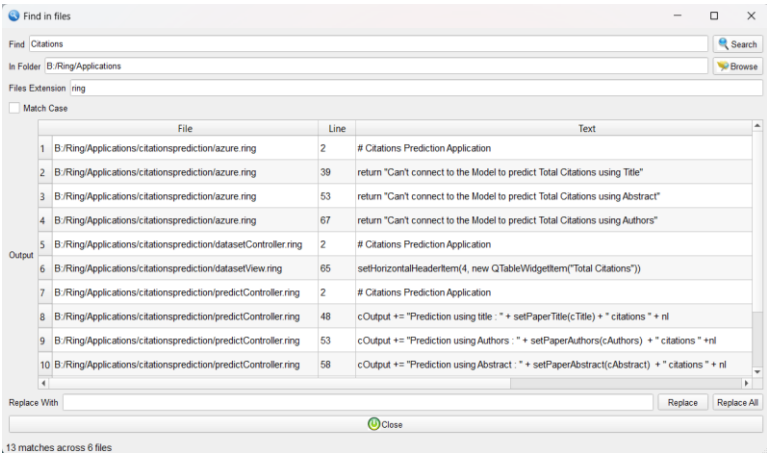
(D) Code Generator

Application/Sample	Size (LOC)	Loading Time (ms)
Analog Clock	256	36
Image Pixel	548	66
Knight Tour	646	67
Othello	780	78
Visualize Sort	817	81
Game Of Life	903	90
Laser	1051	94
Checkers	1354	124
Get Quotes History	3401	117
Discrete Fourier Transform	6417	203

(E) Ring Notepad – Loading Files



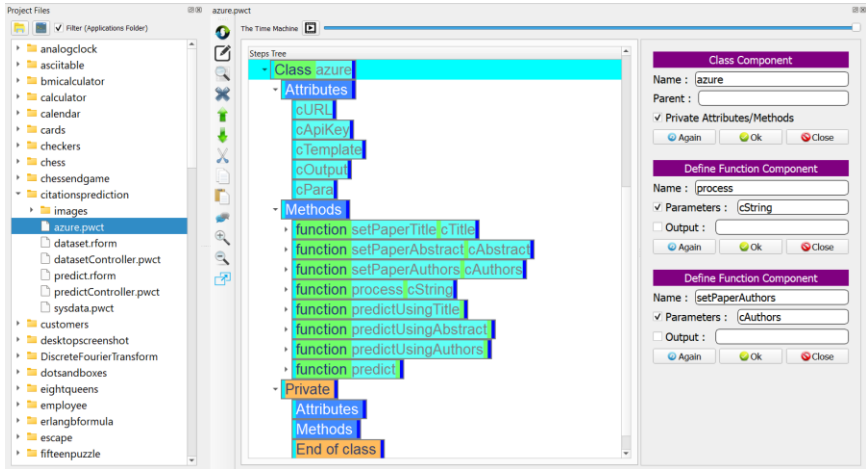
(A) SuperMan Game.



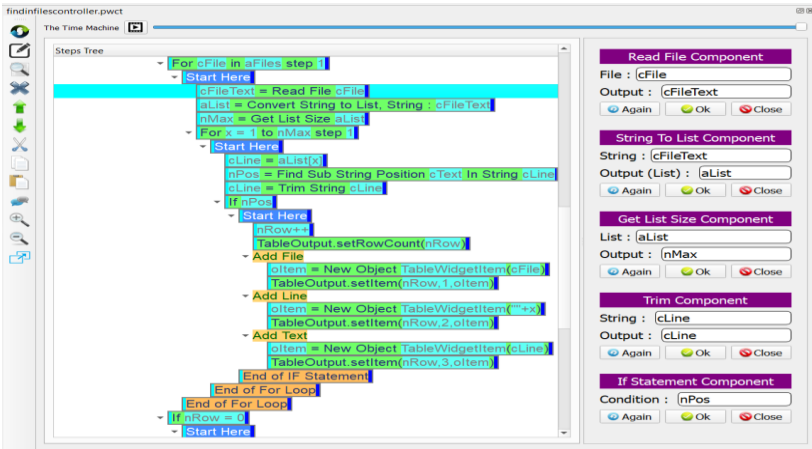
(C) Find in Files Screen Shot.

Attribute	Value
Source code files	1354
Lines of Code	92 KLOC
Dependencies	27 KLOC
Total Lines of Code	119 KLOC

(E) PWCT2 Project Size.



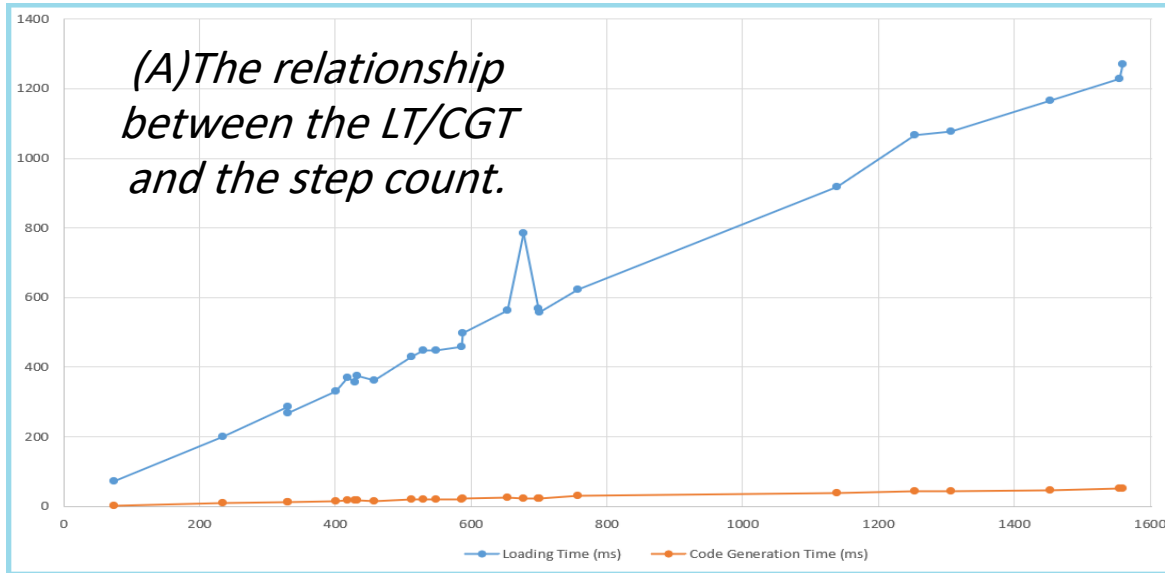
(B) Citations Prediction application.



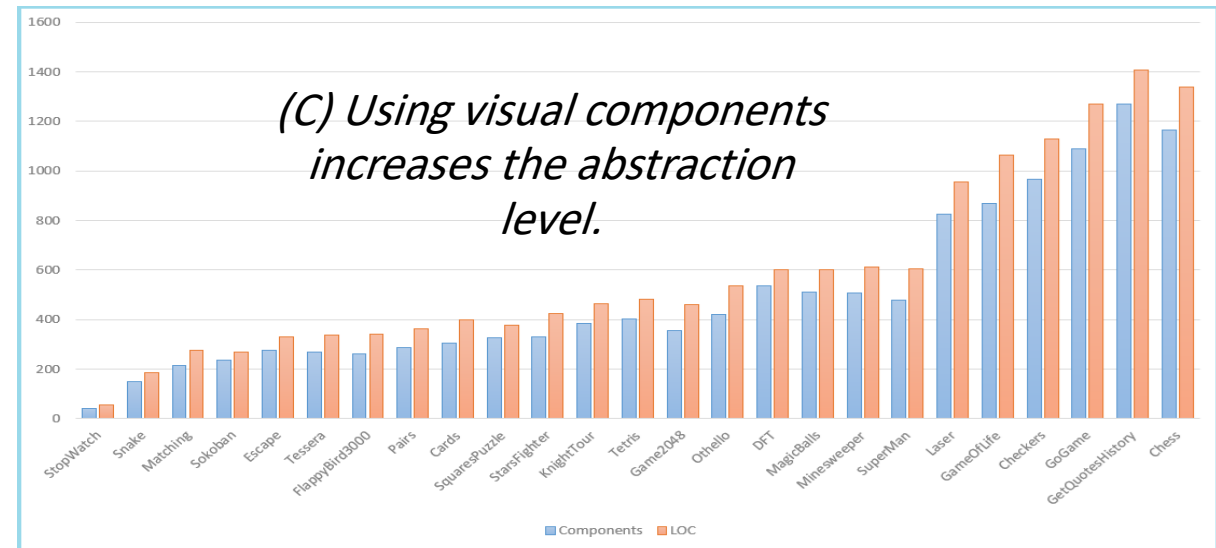
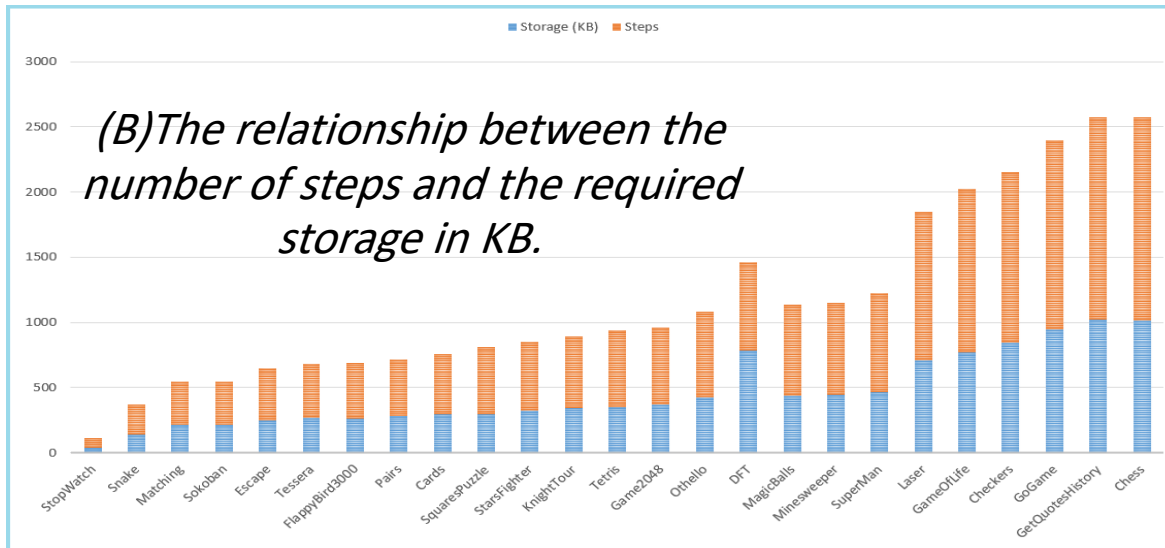
(D) The Find in Files implementation.

Attribute	Ring 1.22 (2024)
Compile-time (ms)	871
Byte-code Instructions	724,382
Ring Object File Size (KB)	18,952
Object File Compressed (KB)	2322

(F) Using Ring to develop PWCT2.



- Storing the Steps Tree in the correct order of control flow.
- Storing the visual source in memory through Ring Lists.
- Using the Ring language instead of Visual FoxPro (VFP).
- Using the Qt framework (RingQt).



Experimental Results

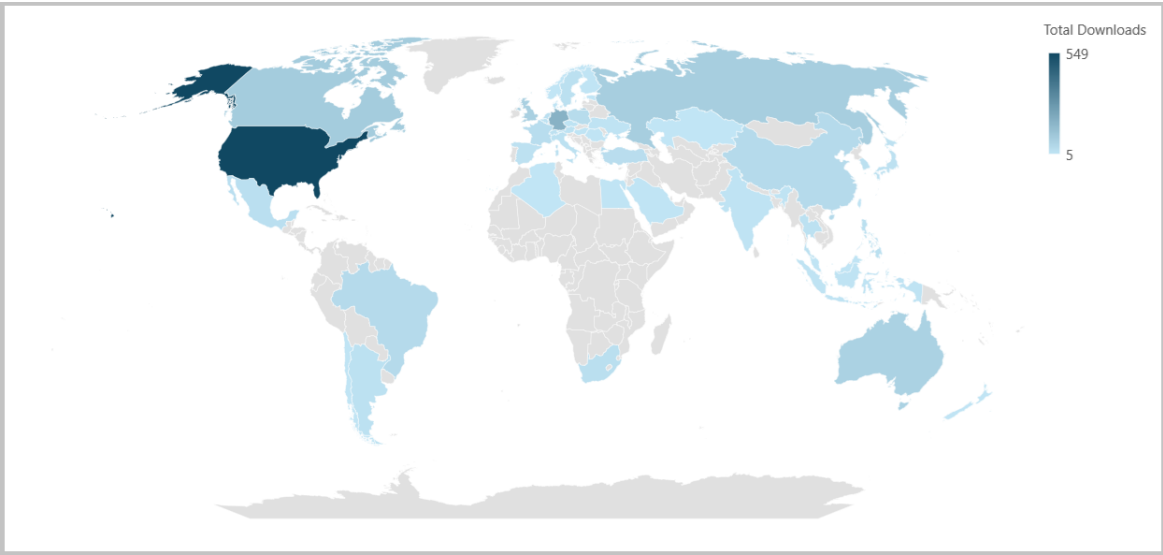
PWCT2 provides approximately **36 times faster code generation** and **20 times lower storage requirements** for visual source files.

Generation	VPL	File	Storage (KB)	Steps	LT (ms)	CGT (ms)
PWCT2	RingPWCT	GameOfLife	773	1253	1068	43
PWCT2	RingPWCT	Checkers	845	1307	1077	43
PWCT2	RingPWCT	GoGame	946	1453	1166	47
PWCT2	RingPWCT	Chess	1012	1560	1270	52
PWCT1	CPWCT	Vmfuncs	7966	1000	549	1748
PWCT1	CPWCT	Refmeta_ext	8243	1075	593	1993
PWCT1	CPWCT	File_ext	9799	1235	747	2651
PWCT1	CPWCT	Vm_loop	11397	1497	862	3862

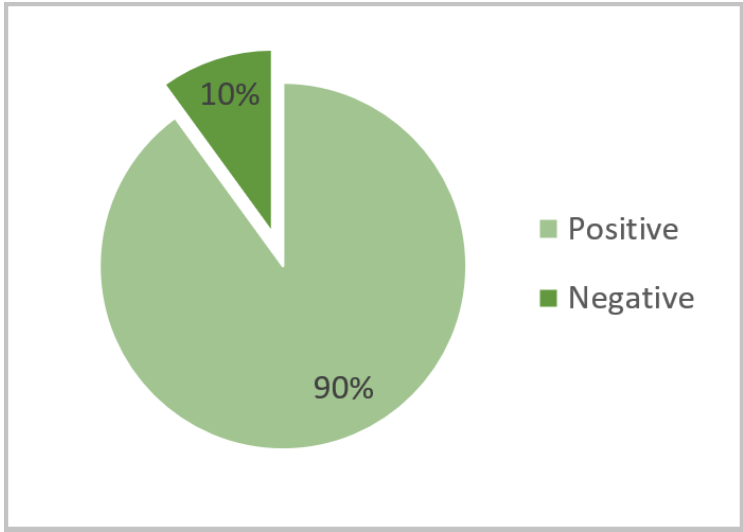
(A) Large Visual Source Files.

Attribute	PWCT1	PWCT2
Visual Programming Language	CPWCT	RingPWCT
Visual Source Files Count (Sample Size)	43	25
Pearson Correlation (Storage vs. Steps)	0.8693	0.9662
Pearson Correlation (CGT vs. Steps)	0.9105	0.9947
Spearman Correlation (Storage vs. Steps)	0.8198	0.9867
Spearman Correlation (CGT vs. Steps)	0.9914	0.9855
Average Storage per Step (KB/step)	13.6751	0.6543
Average CGT per Step (ms/step)	1.2956	0.0353
RMSE for Storage	23.4063	0.1082
RMSE for CGT	1.0539	0.0032

(B) PWCT1 vs. PWCT2.



(A) PWCT2 Software downloads across top countries.



(B) User satisfaction according to steam statistics.

Attribute	Value
Total Videos Count	39
Average Duration (M:S)	8:47
Total Duration (H:M:S)	5:42:27

(C) PWCT2 videos

Attribute	Value
Impressions	1.72 M
Web page visits	159 K
Software owners	20,623
Users launched the software	1772
Average usage time	9 h and 40 min
Total usage time	Over 17,000 h

(D) PWCT2 Users.

Thesis Presentation Outline

01

Introduction

02

**Problem Statement & Motivation &
Contributions**

03

Literature Review

04

Materials & Proposed Methods

05

Experimental Results

06

Discussion and Comparisons

07

Conclusion

- **Large Storage Size:** Visual implementations tend to occupy more storage space.
- **Memory Requirements** for Multiple Instances.
- **Lacks support for drag-and-drop** functionality (Steps Tree Editor).
- **Performance Challenges** with Large Visual Source Files
- **No** support for **importing** textual source code.
- **PWCT** is designed to work only under **Microsoft Windows**.

Suggestions to mitigate these challenges

- Separate the visual source into **many files** with clear names and purposes.
- Keep each visual source file to **fewer than a few thousand steps**.
- Open **related visual source files** according to the current task.
- External **tools** are needed when **searching multiple generated source code files**.

- **PWCT2** supports only the **Ring programming language**
- **PWCT1** provided visual components for **various textual programming languages**
- **PWCT2 is not compatible with PWCT1** (visual component design/visual source file formats).
- **PWCT2** is currently distributed as a **desktop tool rather than a web-based application**.

Thesis Presentation Outline

01

Introduction

02

**Problem Statement & Motivation &
Contributions**

03

Literature Review

04

Materials & Proposed Methods

05

Experimental Results

06

Discussion and Comparisons

07

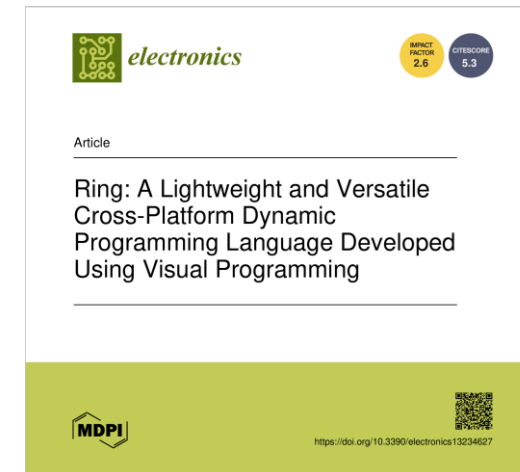
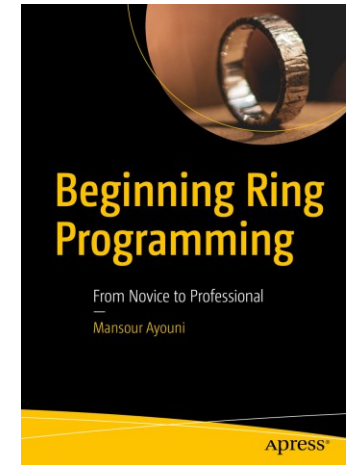
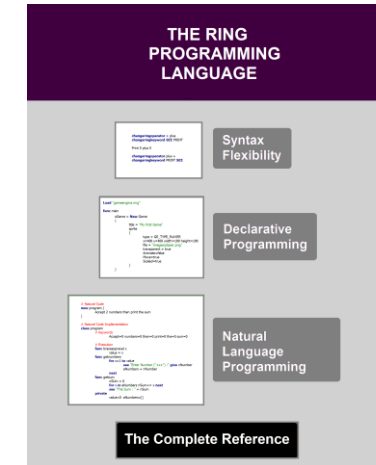
Conclusion

In this thesis we introduced the **Ring textual programming language** and the **PWCT2 visual programming language**.

Ring is based on **visual implementation** (18,945 components that generate 24,743 lines of ANSI C code).

Ring combines a **lightweight implementation**, rich and versatile **standard library** and the same Ring implementation serves a **wide range of environments**.

Customization is a key feature of Ring (Change syntax/Create DSLs).



We have developed PWCT2 (**enhanced features, works on different systems, provides approximately 36 times faster code generation and 20 times lower storage requirements for visual source files**).

PWCT2 is a **self-hosting** VPL developed and maintained for many years using the Ring language (**92,000 lines of Ring code**).

PWCT2 **contains 394 visual components and can convert textual Ring code into visual code**.

PWCT2 has been widely distributed to users via the Steam platform, receiving positive feedback. On Steam, the software has been launched by 1,772 users, with a total recorded usage time exceeding 17,000 hours.



Article

PWCT2: A Self-Hosting Visual Programming Language Based on Ring with Interactive Textual-to-Visual Code Conversion



<https://doi.org/10.3390/app15031521>

We demonstrated the growth of the Ring language over eight years; **while being a lightweight language, we noticed a growth in the implementation size from 16 KLOC in 2016 to 24 KLOC in 2024. This percentage of growth (51%) requires attention.**

We notice that the performance of the Ring programming language has improved over time, and it is now fast enough for many use cases as a scripting language. **However, improving Ring's performance remains a challenge, and we aim to provide optimizations and enhancements with each new release.**

In the future, we plan to build multiple projects on top of the Ring language

- Localization package for many human languages,
- Various domain-specific languages for different fields.
- Modern framework that includes many templates for database applications.

We aim to enhance the PWCT2 visual programming language

- Supporting additional textual programming languages such as C, Java, C# and Python.
- Improve the environment by offering translations in various human languages.
- Add more components that provide better support for Ring libraries.

جامعة
الملك سعود
King Saud University



Thank You