ETSI
The Standards People

IoT Conference 2023

A Survey on IoT Programming Platforms: A Business-Domain Experts Perspective

Presented by: Fatma-Zohra Hannou

MINES
Saint-Étienne

06/07/2023

Introduction

The Internet of Things (IoT) has opened up a broader, deeper and more realistic perception of the surrounding environment by transforming any "thing" into a quasi-continuously available data source or control lever.

- The IoT market expected to double from 300 billions in 2021 to reach 600 billions in 2026.
- While this diversity accelerates the adoption of IoT within multiple domains, such as building automation, healthcare, agriculture, or energy management, it raises interoperability and access challenges. The subsequent complexity might represent a significant barrier to the immediate IoT technologies use for small organizations (or non-technical companies) that cannot afford the cost of hiring IoT expert teams to handle complex architectures and deployment processes.

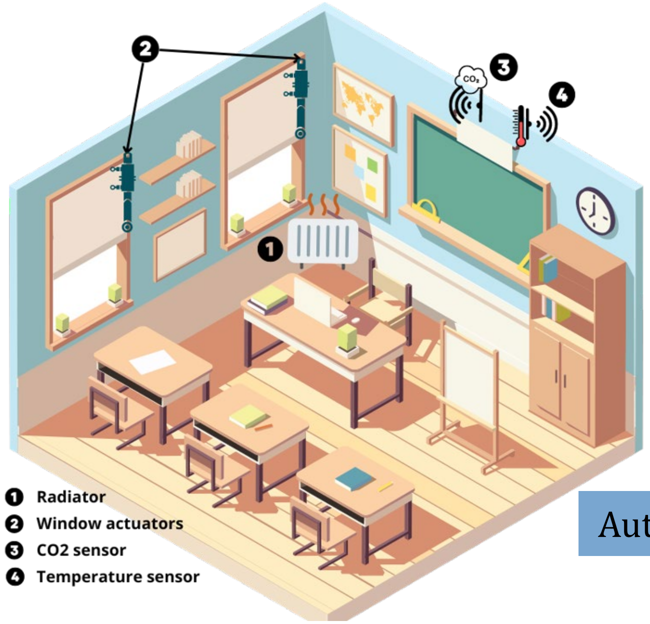
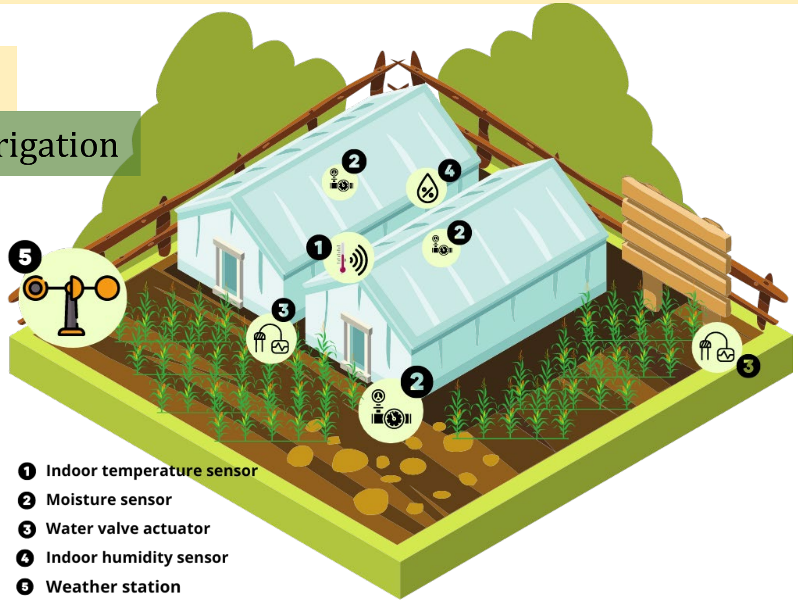


Figure 1

Automatic Windows Openning

Figure 2

Automatic Crop Irrigation



IoT Programming Platforms 620

IoT platforms combine hardware and software technologies to enable the building and deployment of IoT applications through a common user interface, easing access and interoperability within the IoT ecosystem. It provides services and facilities to develop IoT solutions, including device integration, data storage and processing, user communication and development tools.

IoT platform characteristics:

- Business domain
- System architecture
- Interoperability
- Cost and license
- Development support

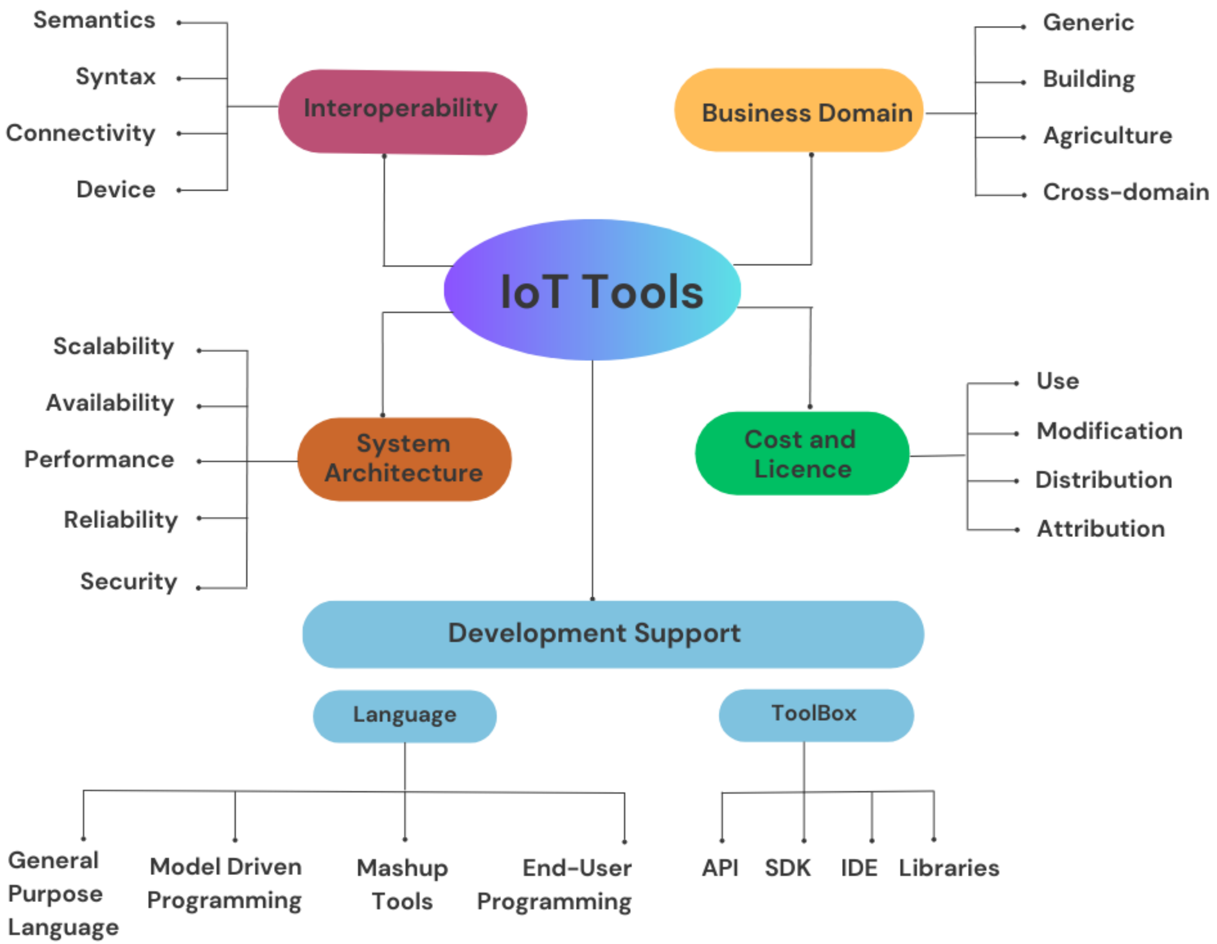


Figure 3: IoT platforms characteristics

Domain Specific Language

> *``a programming language or executable specification language that offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain''* [1]

Internet of Things

Challenges Identify suitable IoT programming platform

Time and staff availability

Business requirements

Affordable costs

Application outcomes

High heteregoneity

Required technical knowledge

Documentation availability

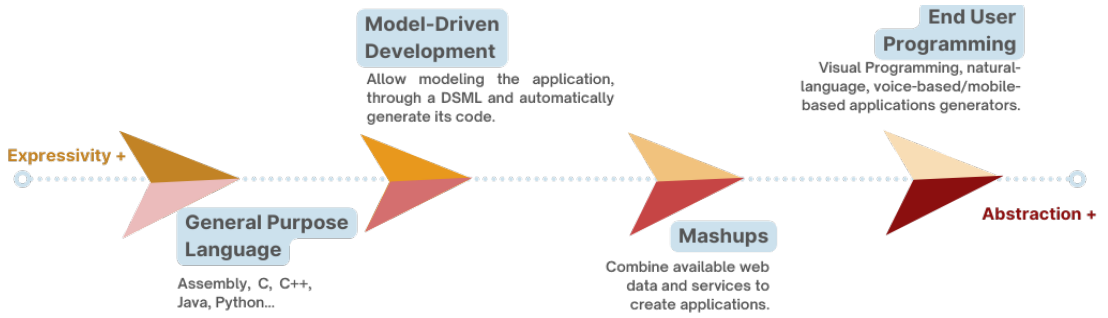
Market fragmentation

Purpose: promote IoT solutions’ adoption by reducing the knowledge gap between domain experts and IoT specialists.

Programming the IoT

Expressivity - Abstraction

Expressive power: denotes the range and complexity of probems that a programming language can effectively represent or solve.



Expressivity +

General Purpose Language

Assembly, C, C++, Java, Python...

Model-Driven Development

Allow modeling the application, through a DSML and automatically generate its code.

Mashups

Combine available web data and services to create applications.

End User Programming

Visual Programming, natural-language, voice-based/mobile-based applications generators.

Abstraction +

Abstraction: denotes extent to which it simplifies and encapsulates low-level technical details providing understandable and usable syntaxes.

General Purpose Programming

- Large community
- Higher interoperability with external systems
- Efficiency (resource-constrained devices)
- Adaptability to hardware specifications
- Integration in cloud-based platforms and scripting APIs

- Multi languages following range of used devices/technologies.
- Steep learning curve

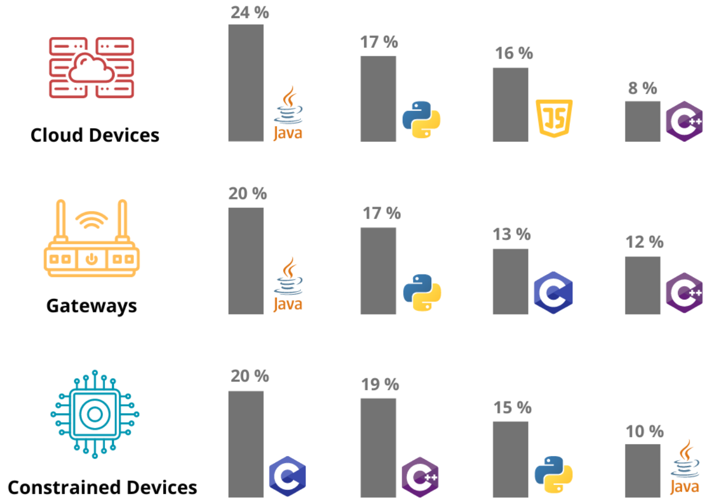


Figure 4: Programming languages usage statistics in IoT programming per tier [2]

Model Driven Programming

A software programming approach concerned with addressing challenges related to application development for complex systems, such as cyber-physical systems or multilayered IoT architectures, using a simplified representation of systems' components through modelling.

An open-source project that provides a Domain-Specific Modeling Language (DSML) along with a suite of tools for cross-platform code generation. It supports various languages including C, C++, Java, and JavaScript, and is designed for use in distributed systems, running on a variety of devices and operating systems. ThingML allows developers to model system components, their behaviors, and interactions, using a textual descriptive syntax.

Other MDP platforms:

- **Midgar**
- **FRASAD** FRAmework for Sensor Application Development
- **UML4Iot**
- **WOX** Web of Topics
- .....

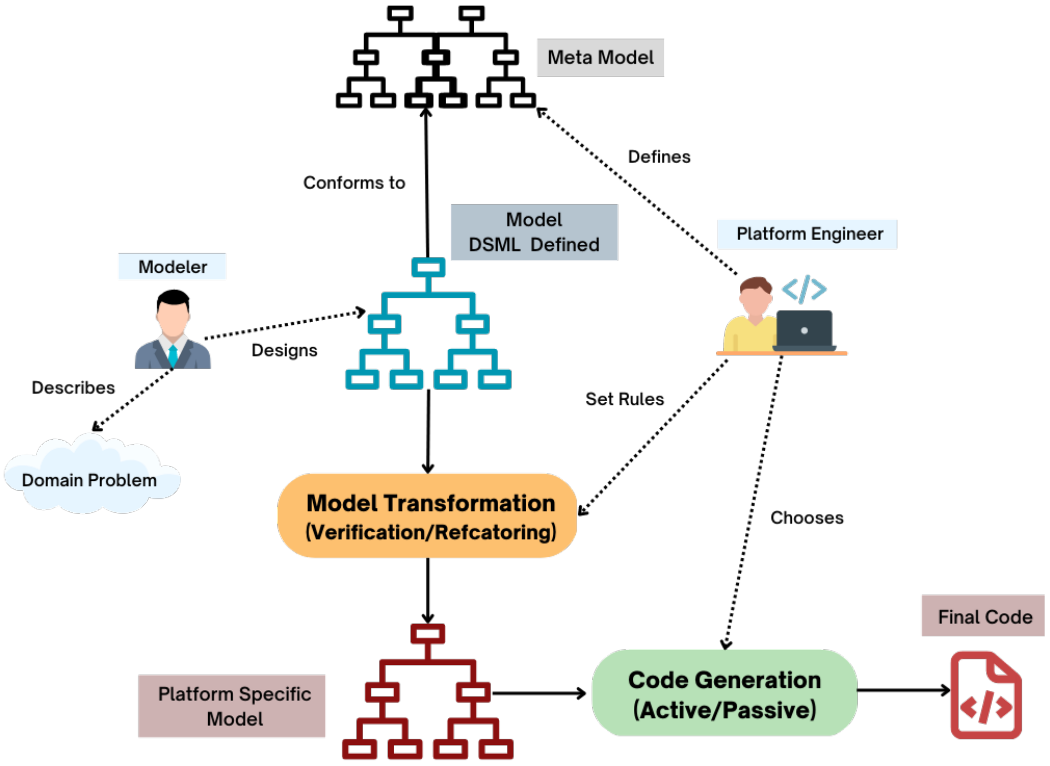


Figure 5: Overview of the model driven programming paradigm

Programming the IoT

Mashups

Applications combining the use of services and data available on the Web to offer a uniform user interface with programming functionalities.

Node-RED

A popular open-source, flow-based programming platform for connecting hardware devices, APIs, and online services. Built on Node.js, providing a browser-based, user-friendly interface for designing and deploying custom IoT applications.
Node-red supports interoperability by offering almost 4400 plugins and integrations, libraries and APIs to integrate additional systems and tools.

Other mashups:

- **WotKit**
- **A-Mage** Atomic Mashup Generator
- **Dynamic Dashboard**
- **CropX**
- …..

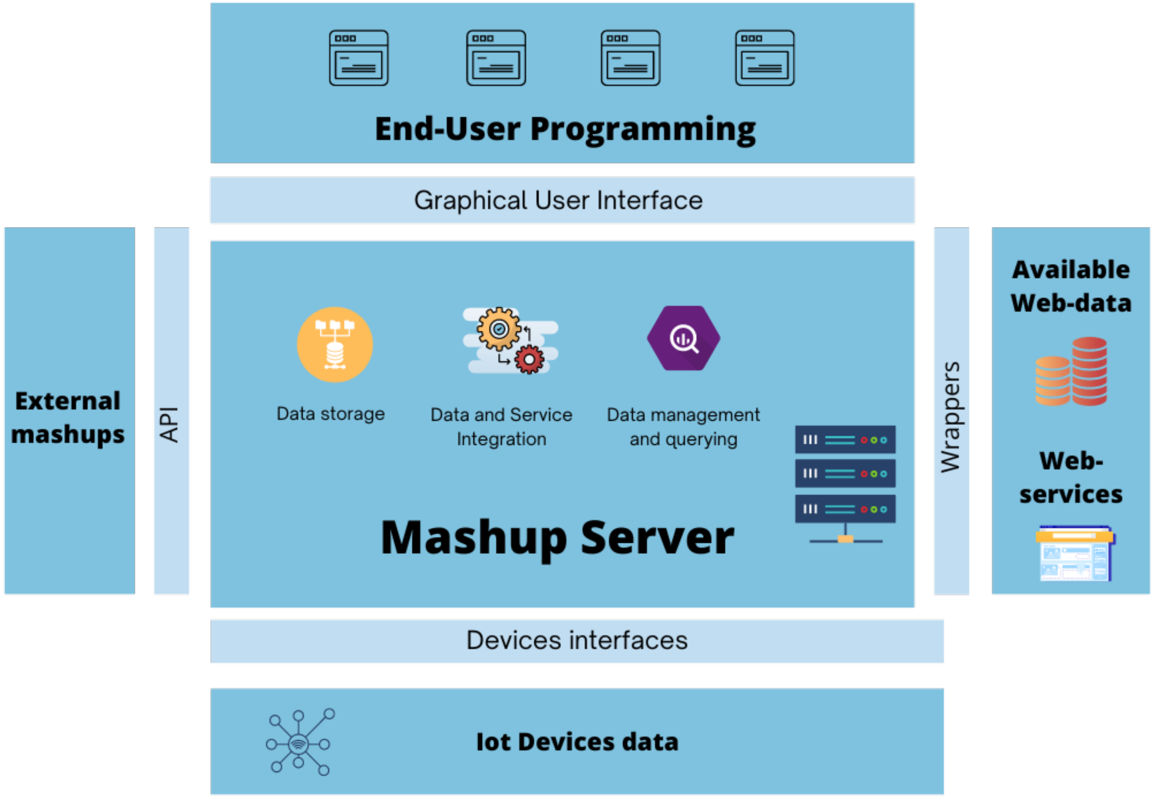
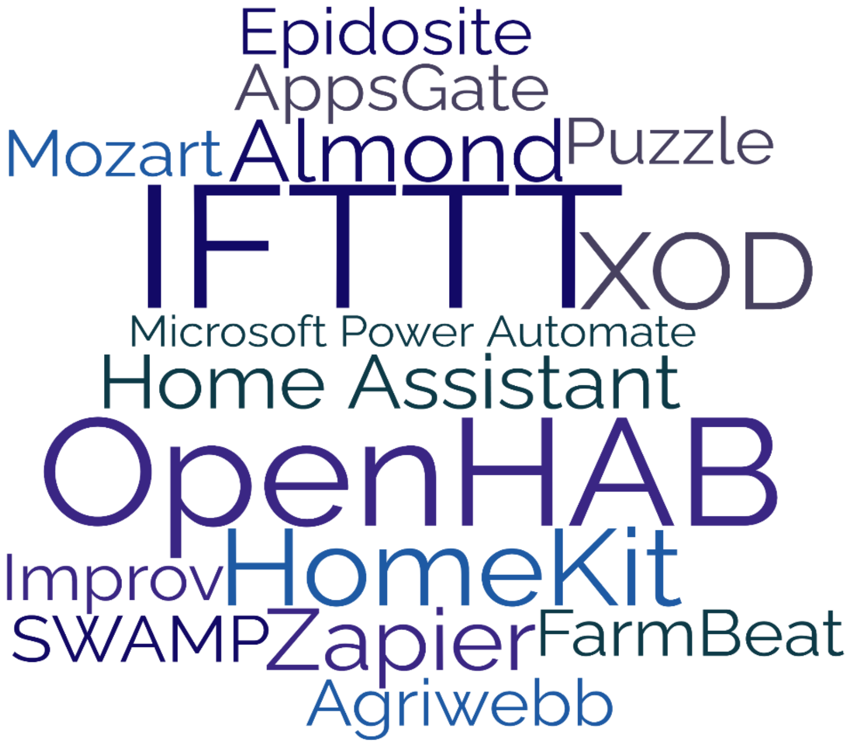


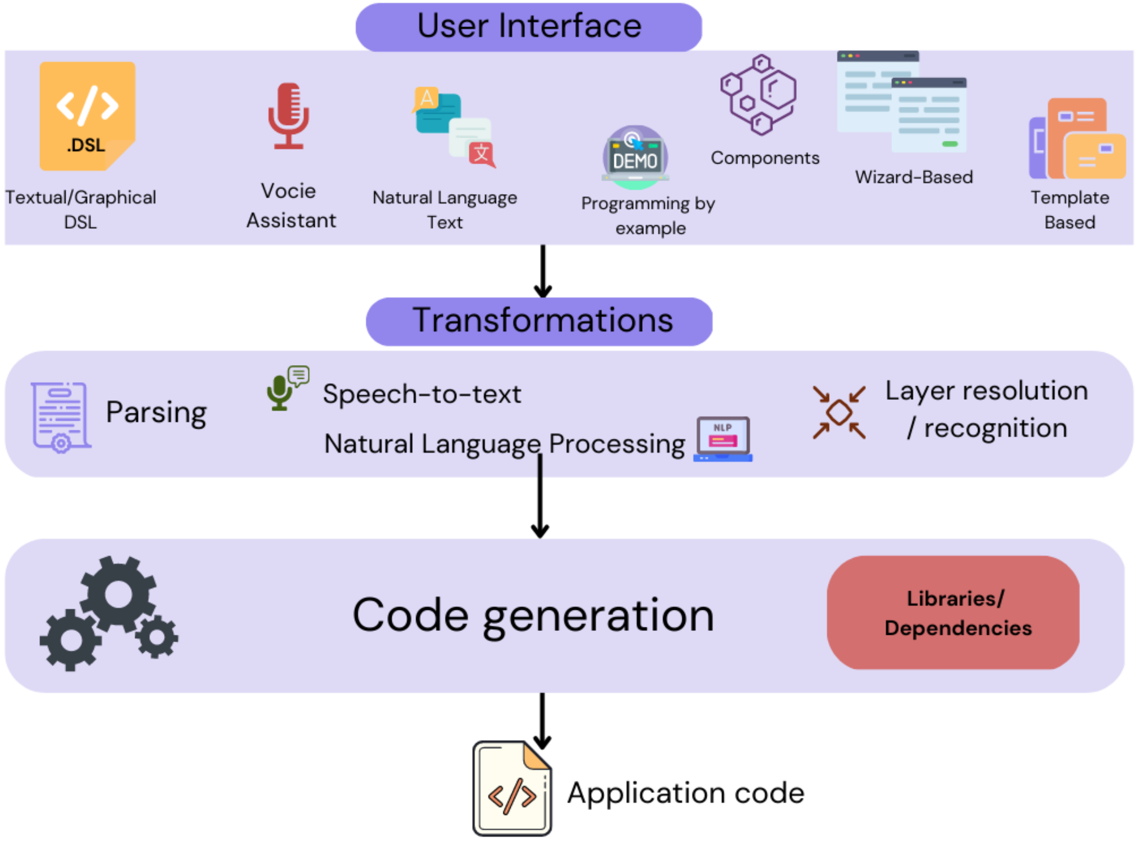
Figure 6: Overview of mashup tools programming style.

Programming the IoT

End-User Programming

A set of techniques that empower end users to write programs by adopting special purpose programming languages.




User Interface

Textual/Graphical DSL

Vocie Assistant

Natural Language Text

Programming by example

Components

Wizard-Based

Template Based

Transformations

Parsing

Speech-to-text

Natural Language Processing

Layer resolution / recognition

Code generation

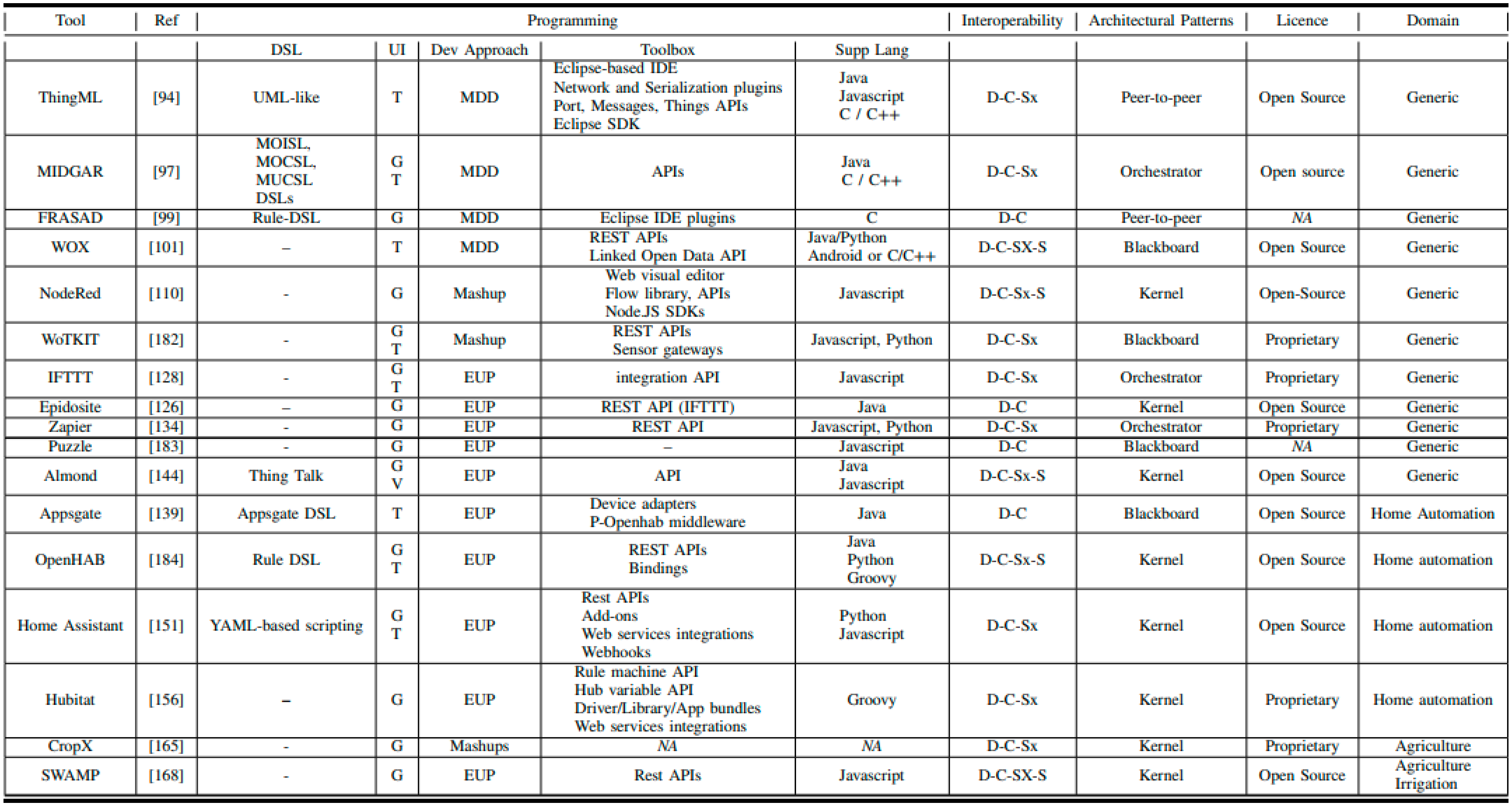
Libraries/ Dependencies

Application code

Figure 7: Overview of the end-user programming techniques

Programming the IoT

Comprehensive Platforms Overview



Tool	Ref	Programming					Interoperability	Architectural Patterns	Licence	Domain
		DSL	UI	Dev Approach	Toolbox	Supp Lang				
ThingML	[94]	UML-like	T	MDD	Eclipse-based IDE Network and Serialization plugins Port, Messages, Things APIs Eclipse SDK	Java Javascript C / C++	D-C-Sx	Peer-to-peer	Open Source	Generic
MIDGAR	[97]	MOISL, MOCSL, MUCSL DSLs	G T	MDD	APIs	Java C / C++	D-C-Sx	Orchestrator	Open source	Generic
FRASAD	[99]	Rule-DSL	G	MDD	Eclipse IDE plugins	C	D-C	Peer-to-peer	*NA*	Generic
WOX	[101]	–	T	MDD	REST APIs Linked Open Data API	Java/Python Android or C/C++	D-C-SX-S	Blackboard	Open Source	Generic
NodeRed	[110]	-	G	Mashup	Web visual editor Flow library, APIs Node.JS SDKs	Javascript	D-C-Sx-S	Kernel	Open-Source	Generic
WoTKIT	[182]	-	G T	Mashup	REST APIs Sensor gateways	Javascript, Python	D-C-Sx	Blackboard	Proprietary	Generic
IFTTT	[128]	-	G T	EUP	integration API	Javascript	D-C-Sx	Orchestrator	Proprietary	Generic
Epidosite	[126]	–	G	EUP	REST API (IFTTT)	Java	D-C	Kernel	Open Source	Generic
Zapier	[134]	-	G	EUP	REST API	Javascript, Python	D-C-Sx	Orchestrator	Proprietary	Generic
Puzzle	[183]	-	G	EUP	–	Javascript	D-C	Blackboard	*NA*	Generic
Almond	[144]	Thing Talk	G V	EUP	API	Java Javascript	D-C-Sx-S	Kernel	Open Source	Generic
Appsgate	[139]	Appsgate DSL	T	EUP	Device adapters P-Openhab middleware	Java	D-C	Blackboard	Open Source	Home Automation
OpenHAB	[184]	Rule DSL	G T	EUP	REST APIs Bindings	Java Python Groovy	D-C-Sx-S	Kernel	Open Source	Home automation
Home Assistant	[151]	YAML-based scripting	G T	EUP	Rest APIs Add-ons Web services integrations Webhooks	Python Javascript	D-C-Sx	Kernel	Open Source	Home automation
Hubitat	[156]	–	G	EUP	Rule machine API Hub variable API Driver/Library/App bundles Web services integrations	Groovy	D-C-Sx	Kernel	Proprietary	Home automation
CropX	[165]	-	G	Mashups	*NA*	*NA*	D-C-Sx	Kernel	Proprietary	Agriculture
SWAMP	[168]	-	G	EUP	Rest APIs	Javascript	D-C-SX-S	Kernel	Open Source	Agriculture Irrigation

User Interface: **T**(Textual), **G**(Graphical), **V**(Voice-Based), **Development approach**: MDD, Mashups, EUP, Supported language: C,C++, Python.., **Interoperability**: **D**evice, **C**onnectivity, **S**ynta**X**, **S**emantics, **Architectural Pattern**: kernel, Orchestrator, Blackboard, Peer-to-peer

Decision Insights

Features-Expressivity

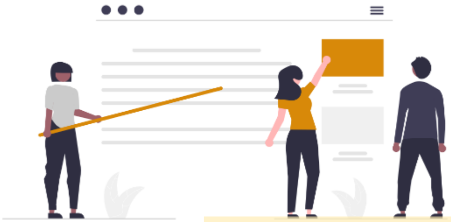
Platform features adaptation to use case requirements, and organization technical settings.

Learning Curve

Abstraction level, toolbox, system architectures, frequency of updates, documentation and tutorials

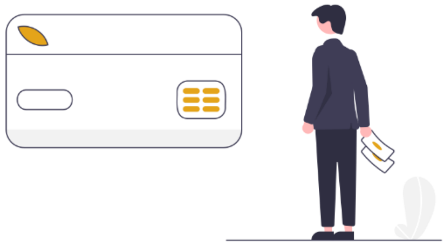
Interoperability

Plugins, integrations, adapters, gateways APIs..



Community Strength

Metric examples: Active user base statistics, mobile application downloads, Github indicators (stars, forks)



Cost-License

Upfront costs: software solution acquisition, platforms-compatible hardware.
Ongoing expenses: subscription plans, support/ maintenance costs, scalability fees

Thank you !