Architectural Views: The State of Practice in Open-Source Software Projects

Sofia Migliorini
University of Florence

Roberto Verdecchia
University of Florence

Ivano Malavolta
Vrije Universiteit Amsterdam

Patricia Lago
Vrije Universiteit Amsterdam

Enrico Vicario
University of Florence

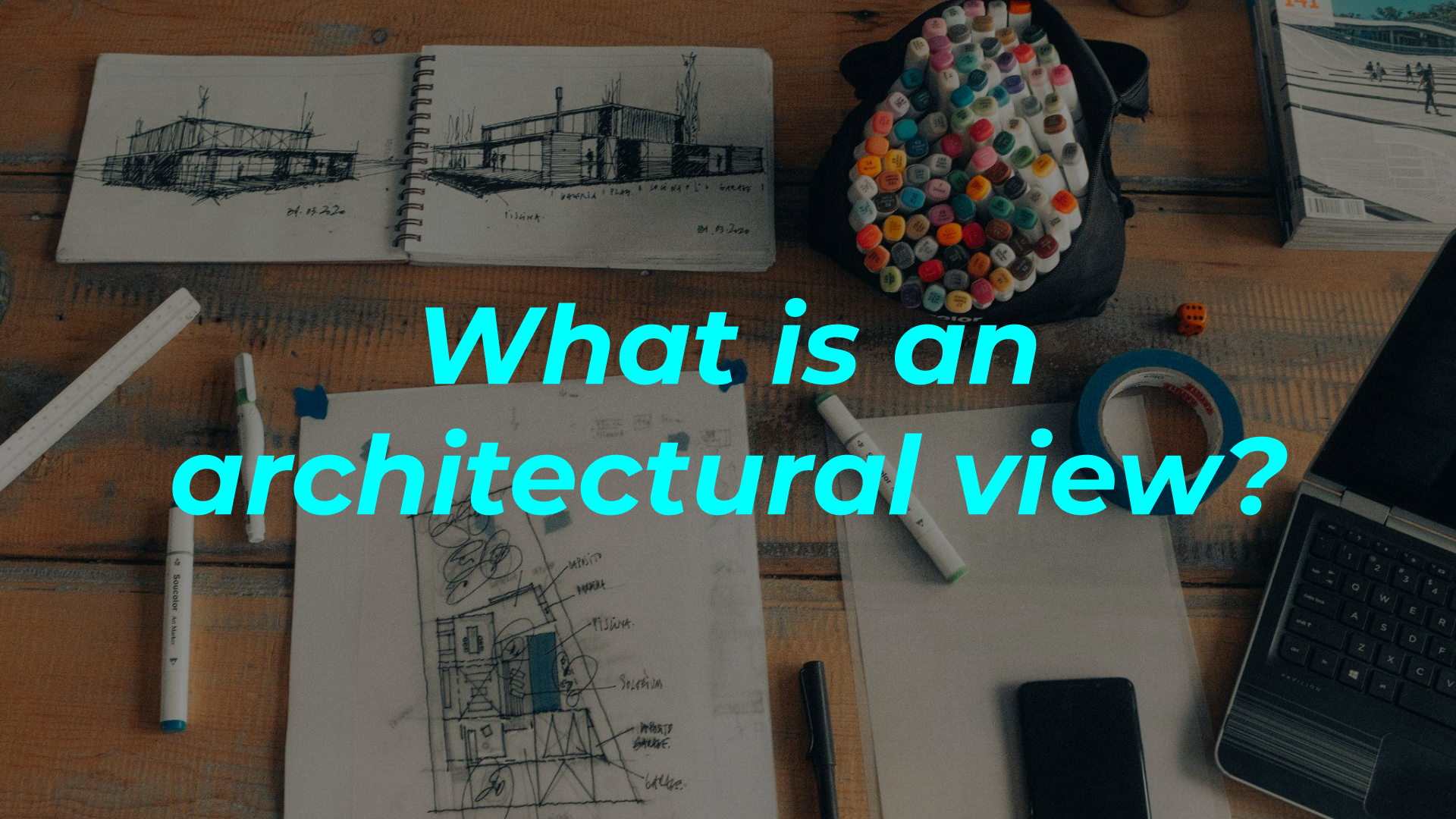
{ST LAB} Software Technologies Laboratory

UNIVERSITÀ DEGLI STUDI FIRENZE

DINFO
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

S2 SOFTWARE AND SUSTAINABILITY
S2 RESEARCH GROUP

VU VRIJE UNIVERSITEIT AMSTERDAM


What is an architectural view?

What is an architectural view?

*Representation of a set of **system elements and relations** associated with them*

Clements, P., Garlan, D., Little, R., Nord, R. and Stafford, J., 2003, May. Documenting Software Architectures: Views and Beyond. In 25th International Conference on Software Engineering, 2003. Proceedings. (pp. 740-741). IEEE.

What is an architectural view?

*One of the **primary methodologies** to design and communicate software architectures*

Clements, P., Garlan, D., Little, R., Nord, R. and Stafford, J., 2003, May. Documenting Software Architectures: Views and Beyond. In 25th International Conference on Software Engineering, 2003. Proceedings. (pp. 740-741). IEEE.

What is an architectural view?

*Effectively **documenting** **an architecture** is **as** **important as crafting it***

Clements, P., Garlan, D., Little, R., Nord, R. and Stafford, J., 2003, May. Documenting Software Architectures: Views and Beyond. In 25th International Conference on Software Engineering, 2003. Proceedings. (pp. 740-741). IEEE.

There must be a lot of studies on how architectural views are documented

Goal

Goal

Understand ***how and for what architectural views are used*** in open-source practice

Goal

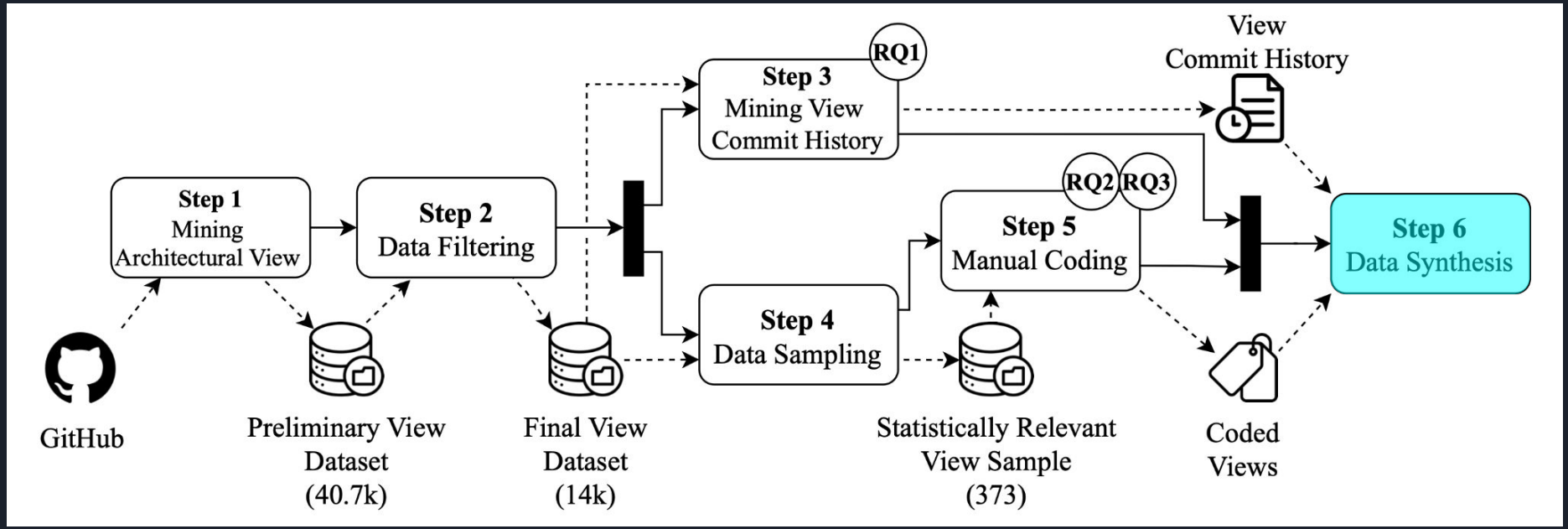
RQ_1: What is the view **history?**

RQ_2: What is the view **syntax?**

RQ_3: What is the view **content?**

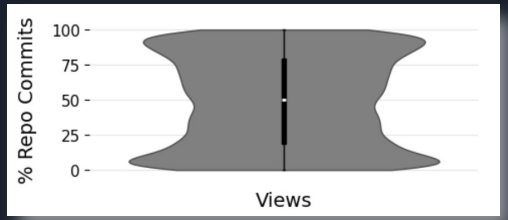

Research process
overview

Research process overview

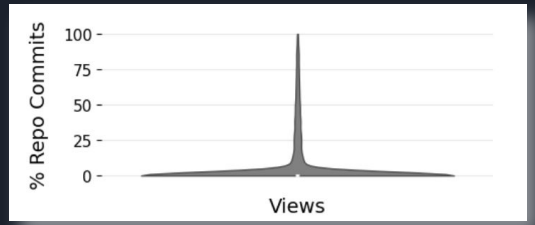


Results

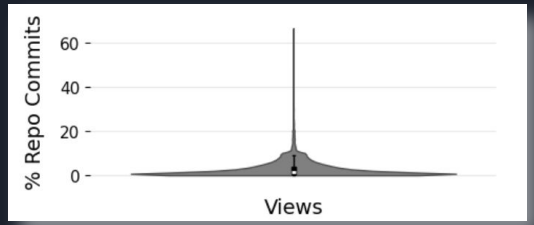
Results RQ_1: History of Architectural Views



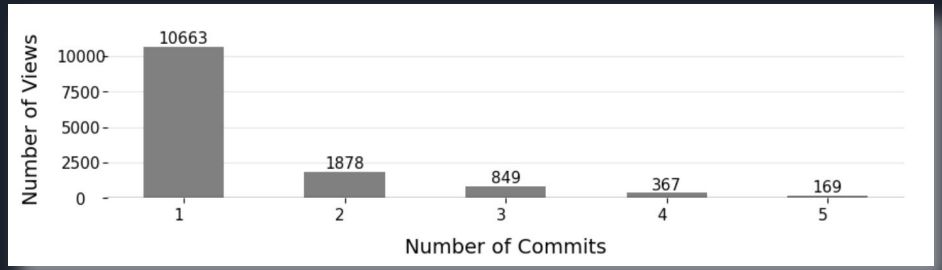
Views are most **introduced** either at the **start or end of projects**



75% of views are **never updated. Edits closely follow each other**

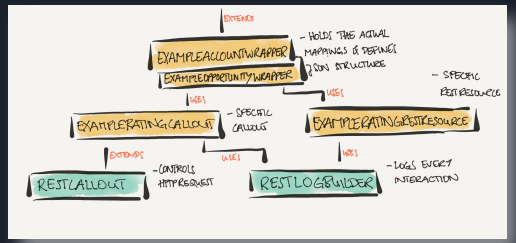


Only **very few of commits focus on views**

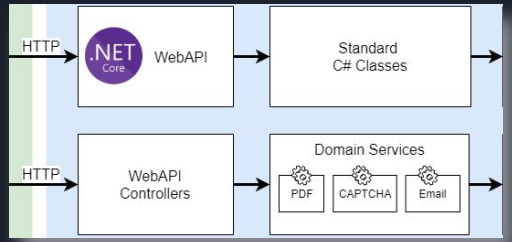


Creating and **editing views** is the **responsibility** of a **single person**

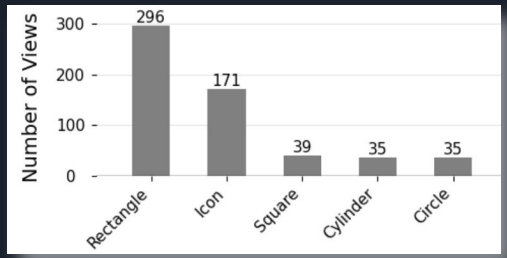
Results RQ_2: View syntax



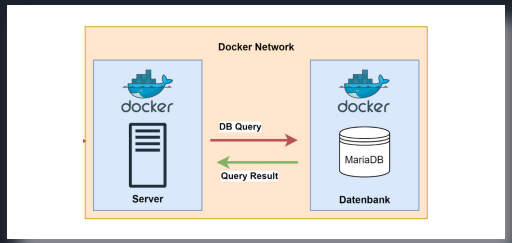
96% of views uses an informal notation (only 4% UML or similar)



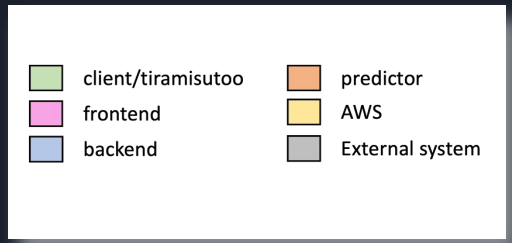
Views use explicit connectors (92%), mostly unidirectional (74%)



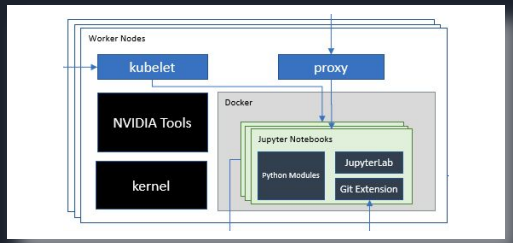
Recurrent rectangle (79%) and icon shapes (46%)



81% of views use colors

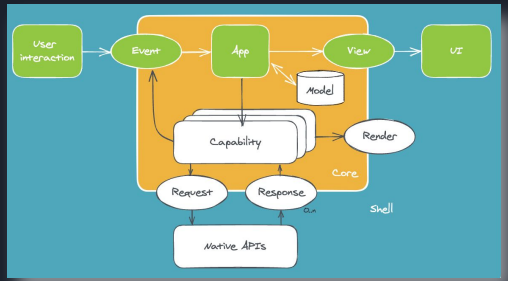


Legends are seldom used (8%)

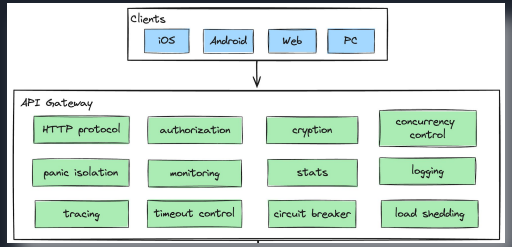


Nested components are common (56%)

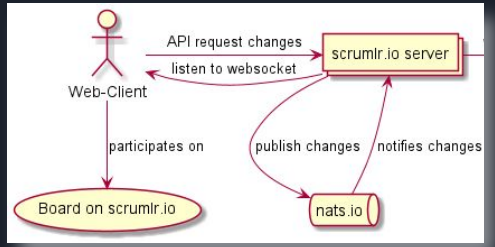
Results RQ_3: Scope, style, and concerns



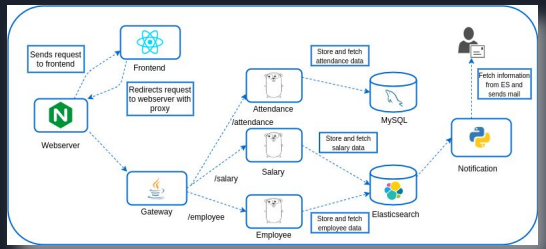
Most views consider the **entire architecture (53%)**



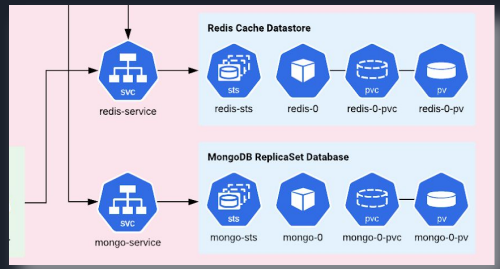
Recurrent **styles: Client-server (20%), Layered (20%), Service-oriented (15%)**



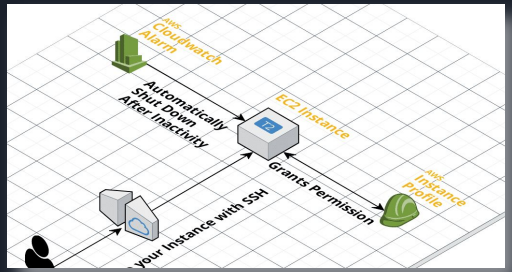
Granularity often **high (53%)** or **medium (37%)**



Either **static (49%)** or **dynamic (42%)** aspects

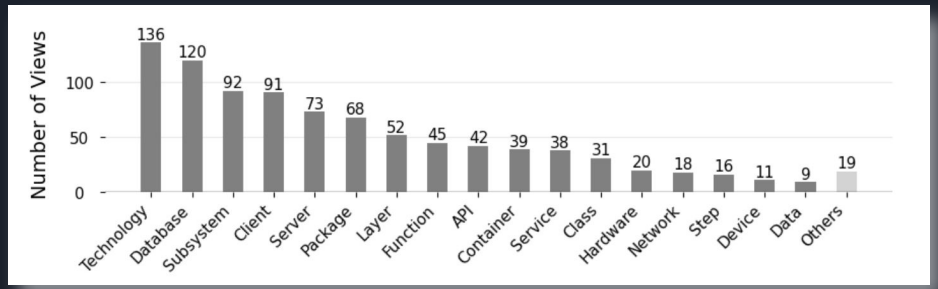


Concerns are often **documentation (30%), deployment (30%)** and **control flow (29%)**

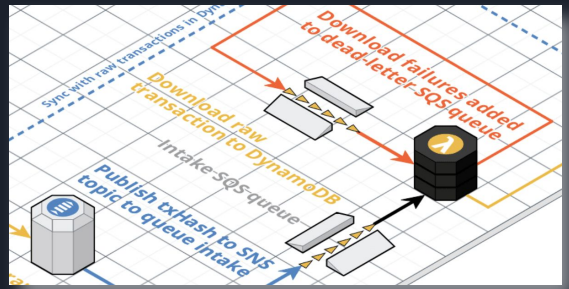


QAs: Maintainability (68%), function suitability (35%), performance (32%), security (9%)

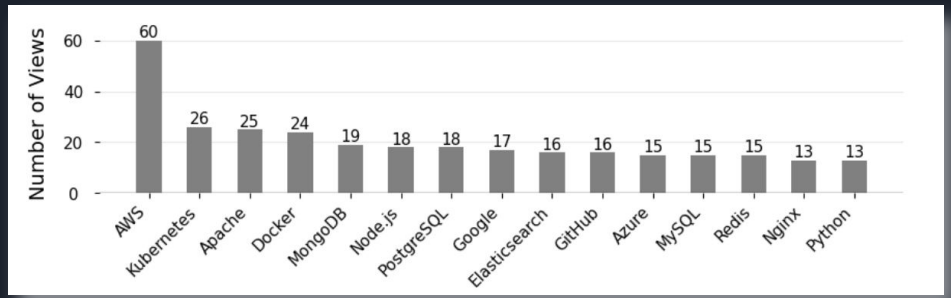
Results RQ_3: Technologies, connectors, and overlays



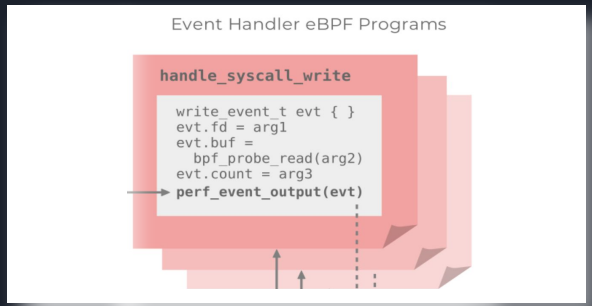
Components are mostly **technologies (36%)**, **databases (24%)**, and **subsystems (24%)**



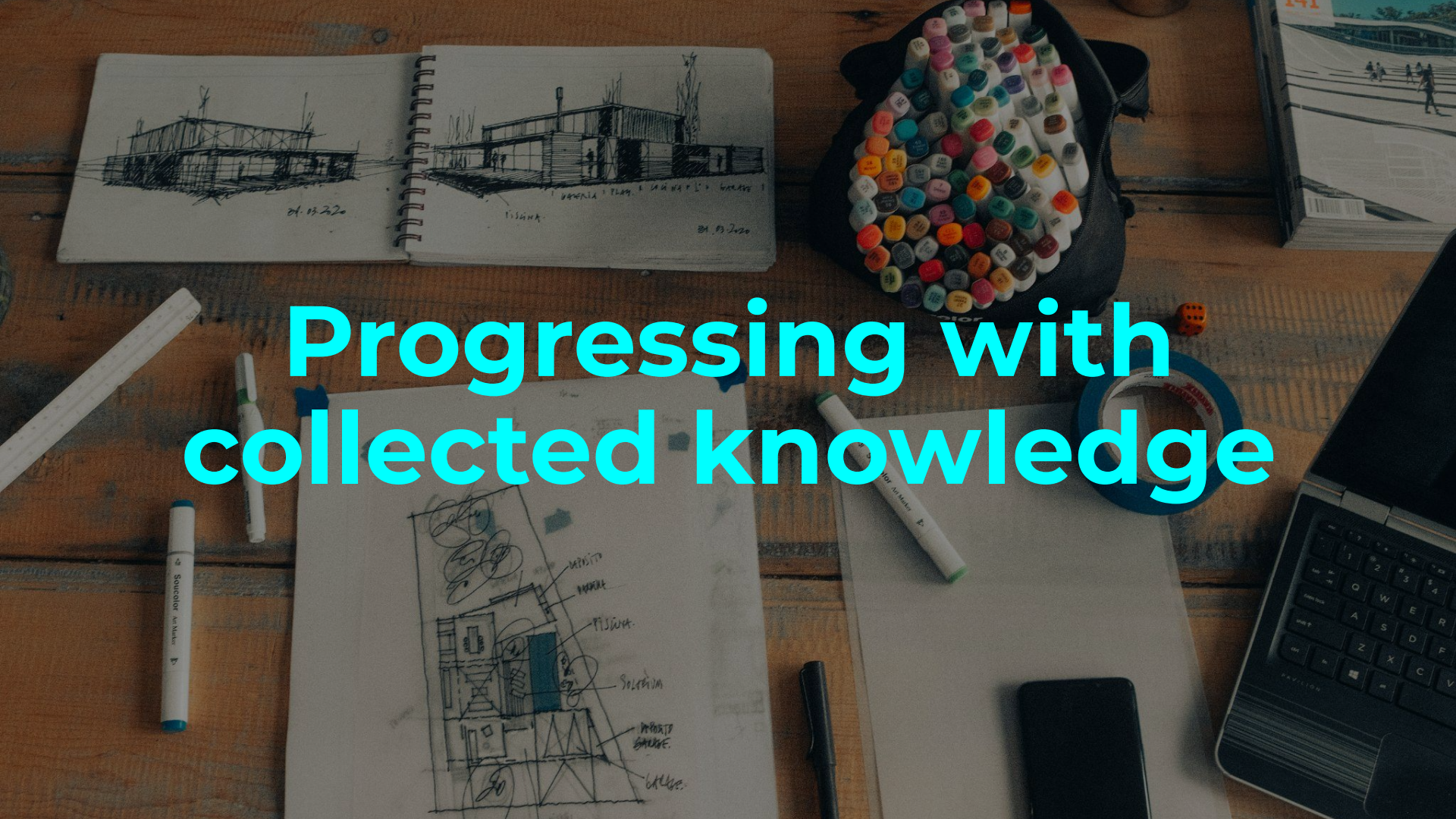
Connectors are frequently **control flow (28%)**, **generic (27%)**, and **data flow (25%)**



35% views report at **least one technology**, with a maximum of 10 technologies per view



Design overlays are seldomly used **(18%)**. Mostly **text (17%)**, **screenshots (12%)**, **URLs (10%)**, and **code (9%)**


Progressing with collected knowledge

State of practice is far from ideal

Views are:

- **One** per project
- **Seldom updated**
- Mostly using **informal notation**
- **High level granularity**
- Using ***ad hoc*** **custom notation**
- **Not based on well established practices**

Potential causes:

- Views are **hard to create and maintain**
- **Value** of views is **not recognized**
- **Lack** of **intuitive templates**

What can we do to ease the adoption of architectural views in open-source practice?

Potential answers:

- **Make views versionable** (e.g., by encouraging adoption of view file formats)
- **Move away from immutable hard-coded images** (e.g., by rendering architectural views in repository web interfaces)
- Provide **intuitive view templates, icon sets, and connectors** implying **syntactic consistency**
- **Integrate view editing** in **collaborative programming environments** (e.g., **dedicated GitHub Actions**)
- Long-term aspiration:

  Establishment of ***`ARCHITECTURE.md`*** *files*