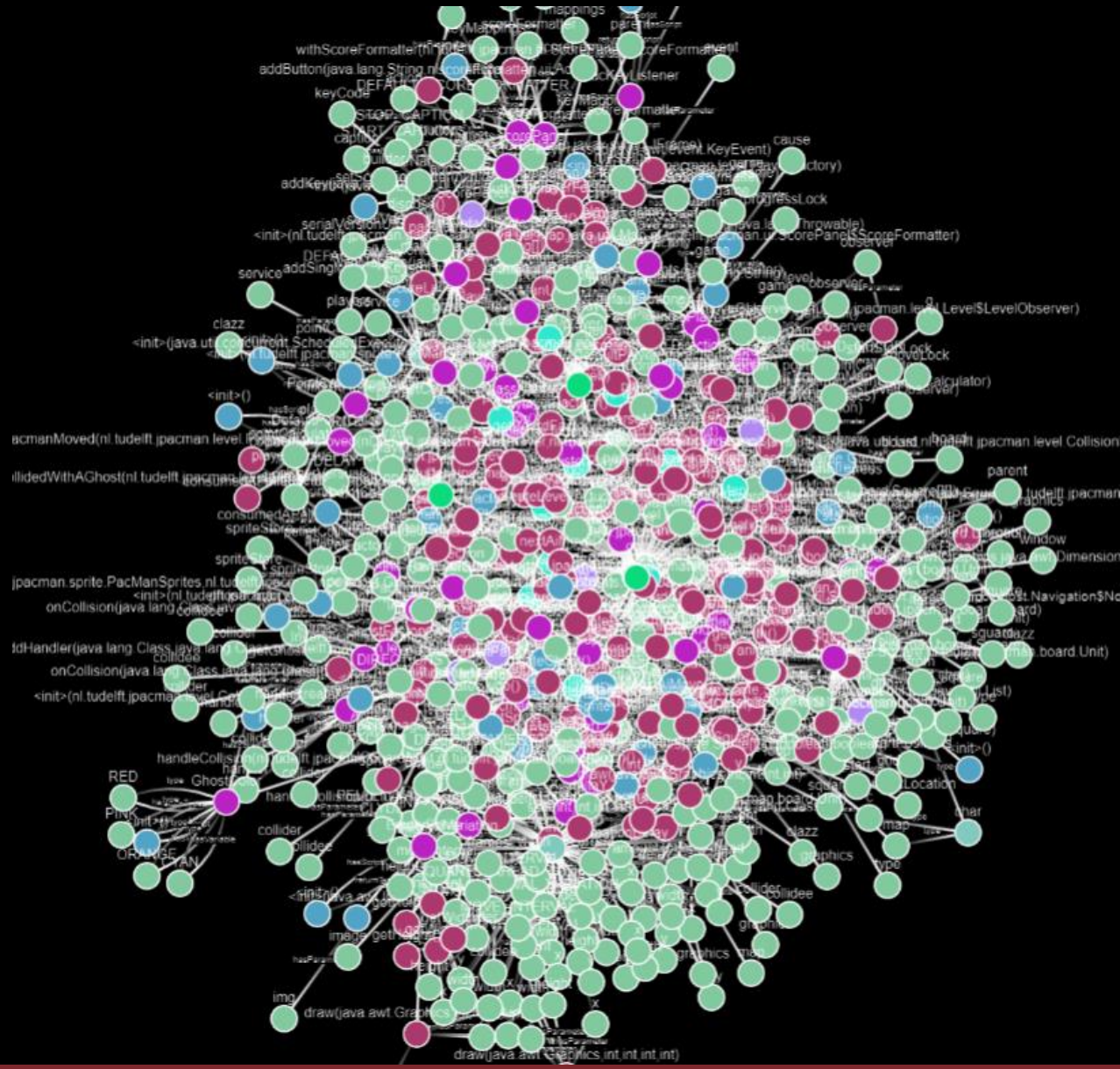
ClassViz:
From Inspection Tool to Research Vessel

Satrio Adi Rukmono • TU Eindhoven • s.a.rukmono@tue.nl / satrio@rukmono.id

BENEVOLES TU/e

Why Visualisation?

- Visualisation helps bridge *complex structure* and *human understanding*.
- Code graphs are precise but large and hard to inspect.
- ClassViz was born to *make structure visible* during development and debugging of code-to-graph instantiation tools.



Research Questions ... ?

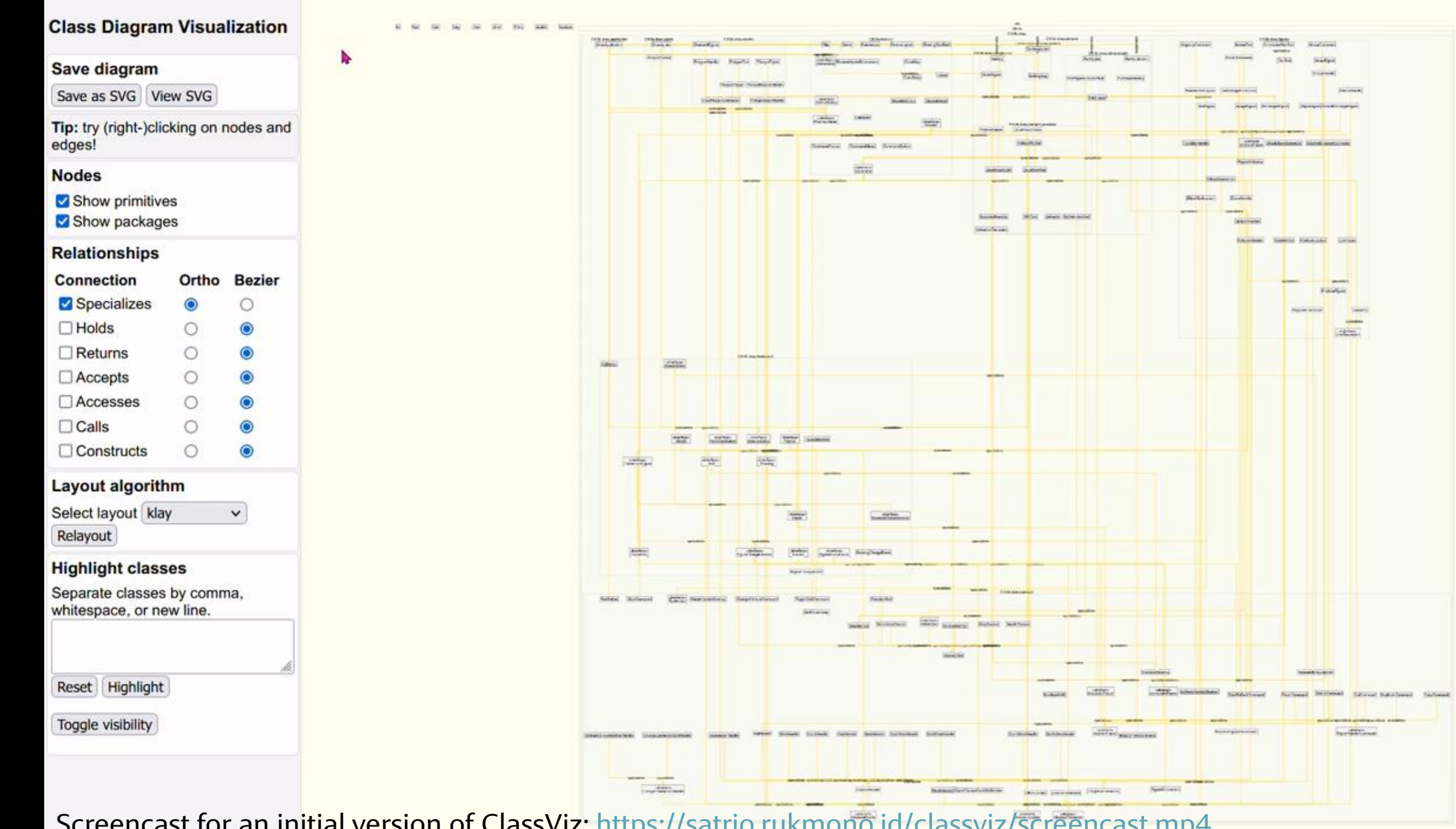
RQ1: What *visual affordances* support effective *lightweight inspection of labelled-property code graphs* for correctness and usability assessment?

RQ2: What *factors* influence the *adoption, extension, and appropriation of software structure visualisation tools* in educational and industrial contexts?

RQ3: How can software visualisation tools be designed to serve as *effective frontends* for diverse *automated analyses* such as architectural recovery and summarisation?

RQ1: Lightweight Visual Inspection

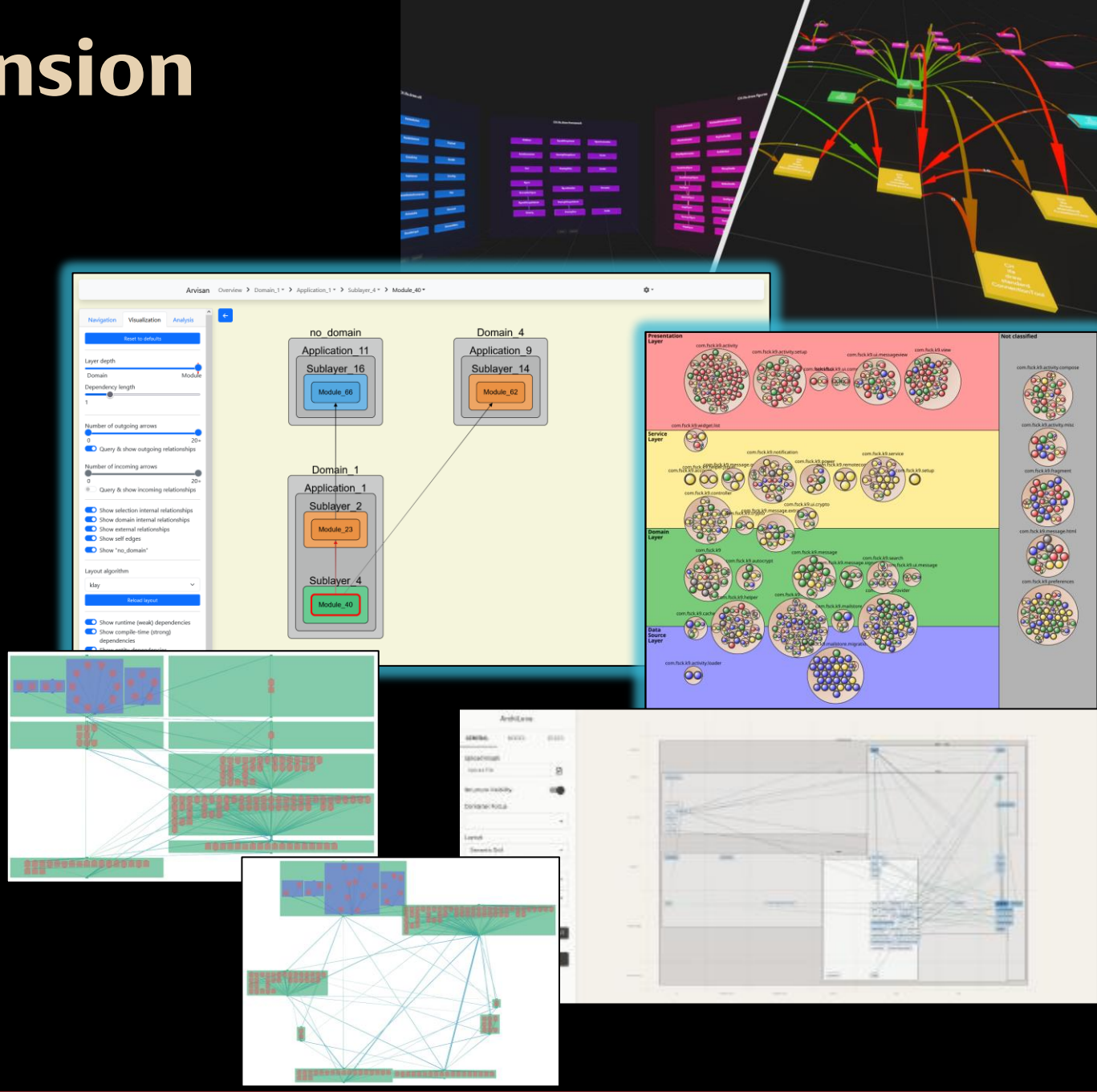
- ClassViz started as a browser tool to sanity-check Labelled Property Graphs (LPGs).
- Design principles: nested boxes (packages/classes), UML-style arrows, filters, click-to-explore.
- Visual affordances prioritised clarity and quick feedback over complexity.



Screencast for an initial version of ClassViz: https://satrio.rukmono.id/classviz/screencast.mp4

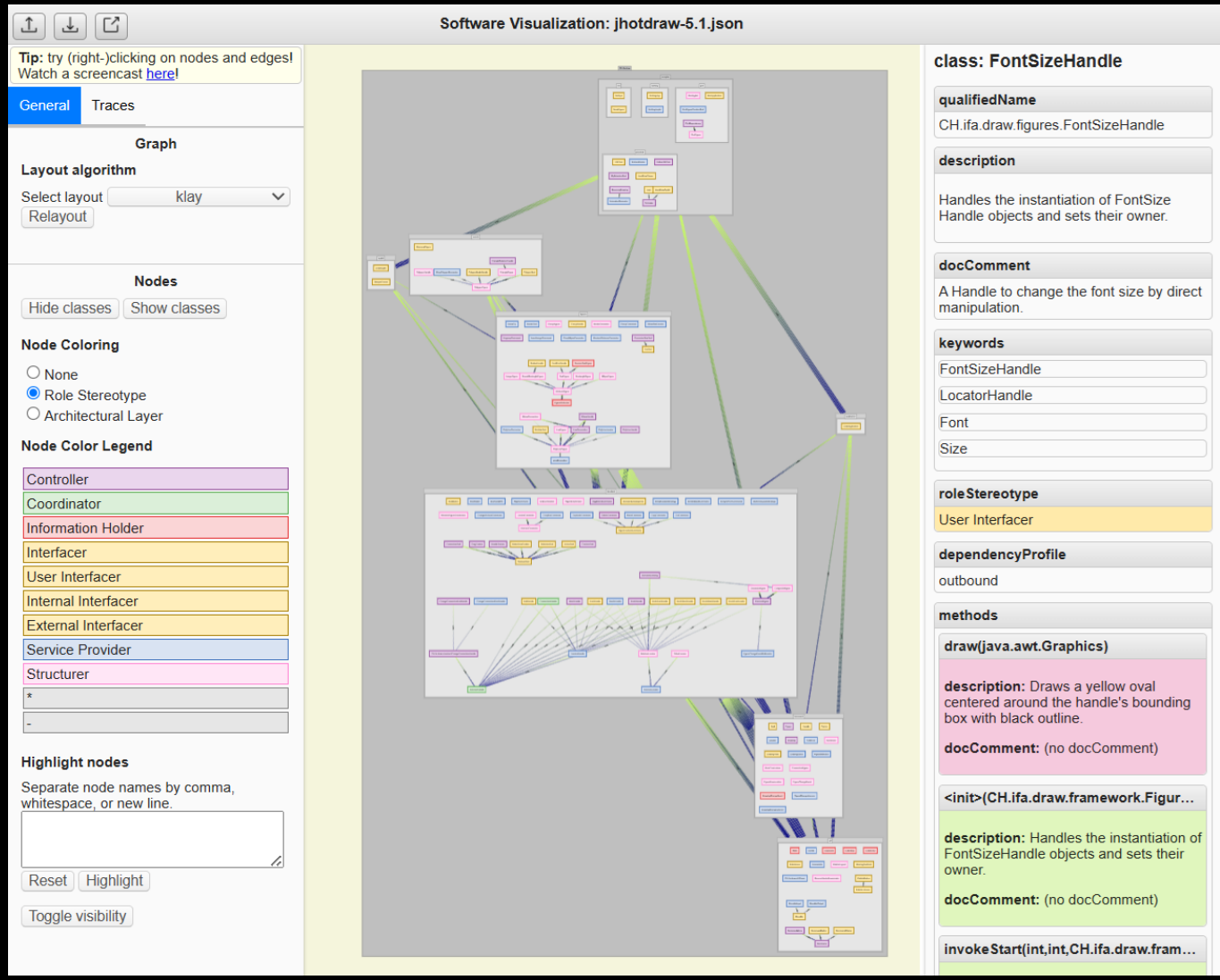
RQ2: Adoption & Extension

- ClassViz spread through student and industrial use due to:
  - Minimal infra, low barrier to entry,
  - Open-ended and modifiable architecture.
- Spawns 11 student projects & helps several graduation projects!
  - Behavioural overlays (Fung, Tanis, He, van Esch),
  - Layout experiments (Kloet, Jeffrey, Atisomya),
  - Industry tools like Arvisan (Kakkenberg et al.), etc.



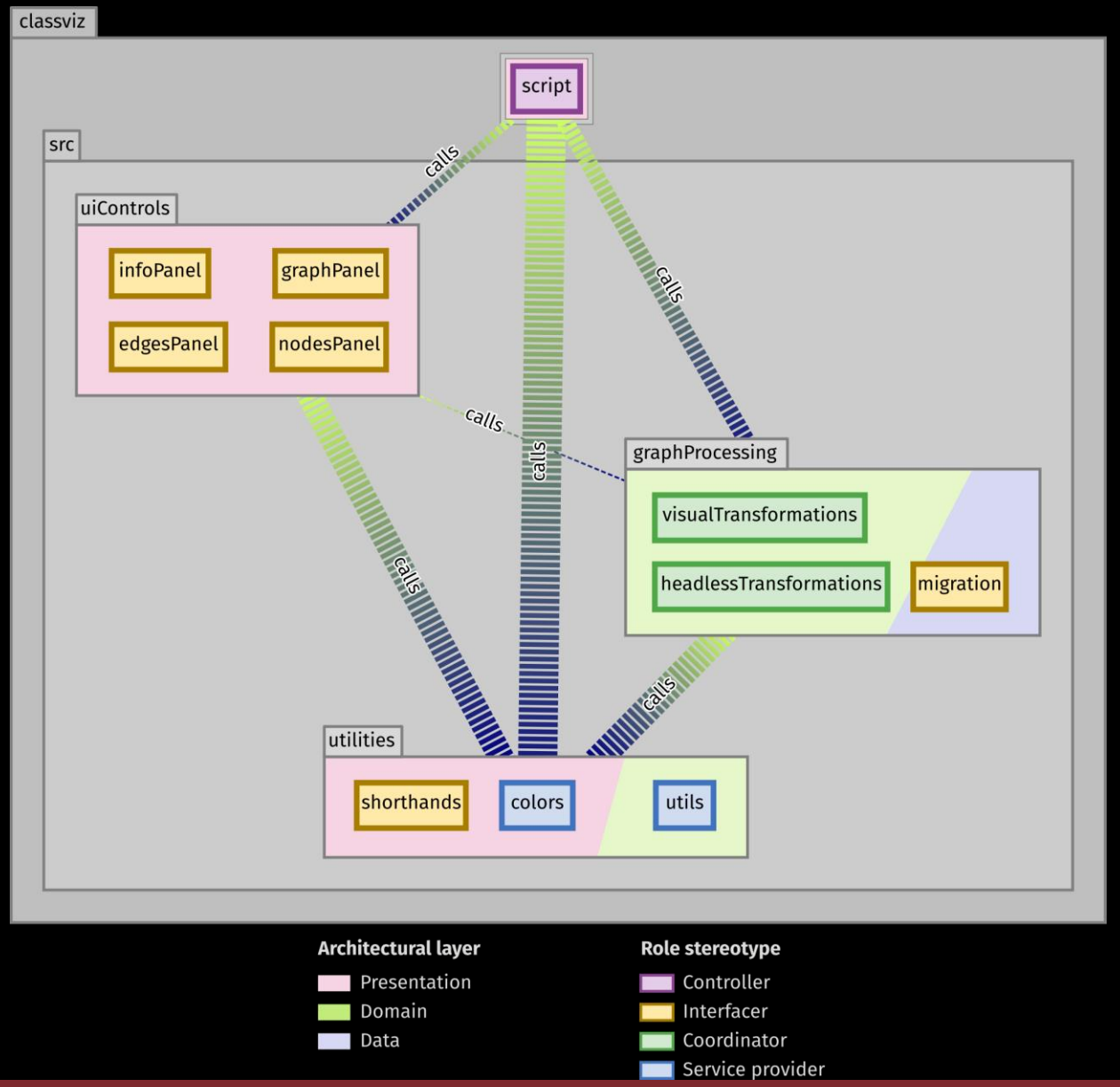
RQ3: Visual Frontend for Explanation

- ClassViz became a frontend for DSAR, summarisation, role classification.
- Graph properties → visual variables: colour, position, edge thickness.
- Supported ASML case study as explanatory interface.
- Example: coloured nodes for roles/layers, detail panes for summaries.



ClassViz in ClassViz

- Self-inspection: visualising the ClassViz codebase itself.
- Reveals structure, dependencies, module roles.
- Also gives feedback on design (e.g., overcentralisation of utils).



Reflection

- ClassViz shows how small tools can evolve into research infrastructure.
- Emphasises:
  - Pragmatism over polish
  - Open architecture over rigid features
- Useful not just to show results, but to *discover* and *shape* them.

Live Demo

https://satrio.rukmono.id/classviz/?p=jhotdraw-5.1