Strengthening development workflows by graphically communicating elements of software design

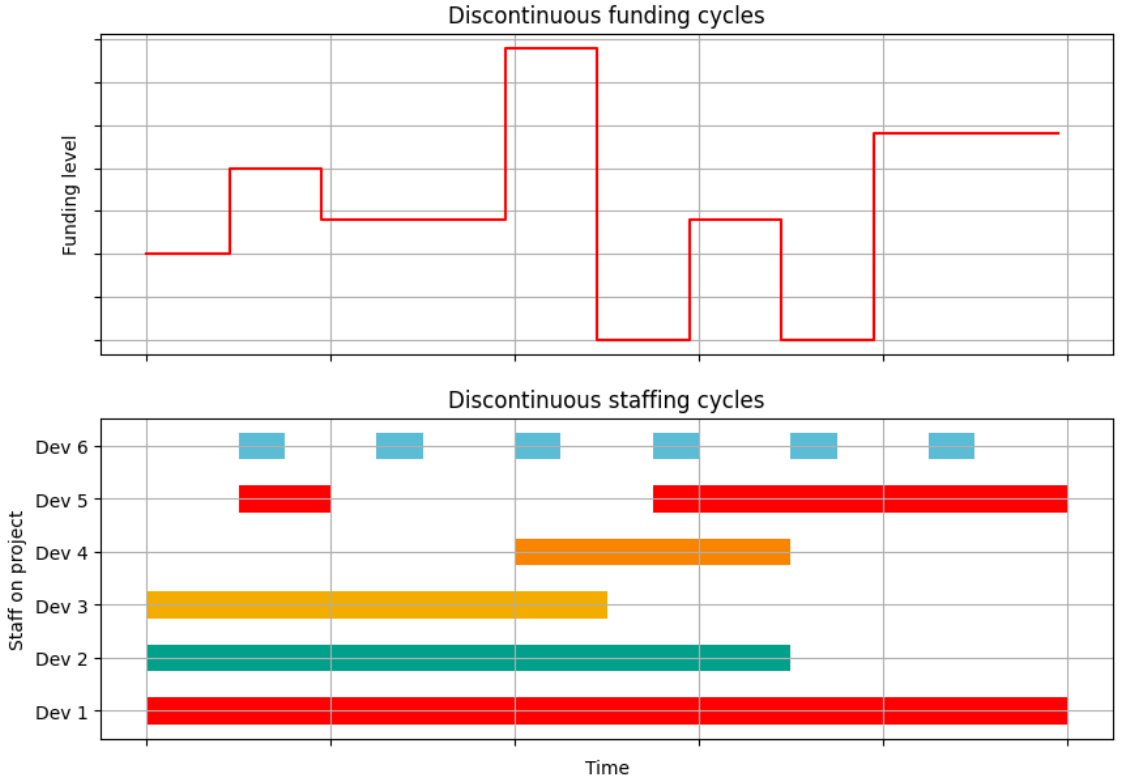
Rafael Mudafort

IDEAS HPC Best Practices

June 12, 2024

Photo from iStock-627281636

Funding and staffing cycles

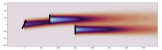


About Me

2017 2018 2019 2020 2021 2022 2023 2024

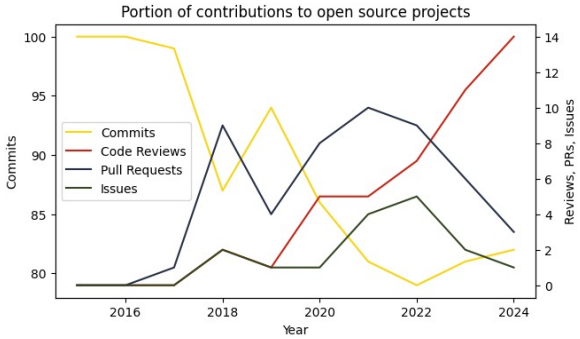
OpenFAST

KiteFAST 

FLORIS 

wcomp

WETO Software Stack

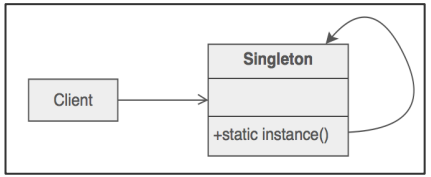


Other contributions

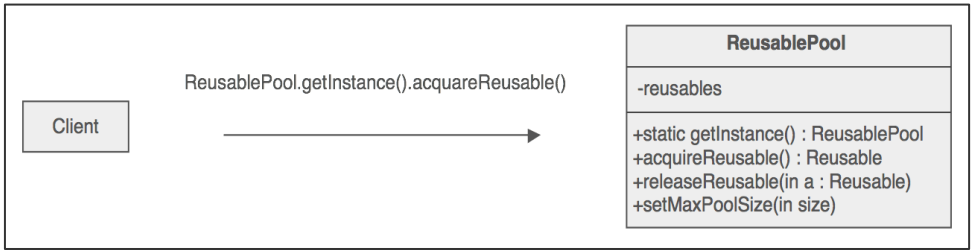
- ERF
- WEIS
- ExaWind
- OpenTurbine
- JOSS Reviewer

Implicit design in software

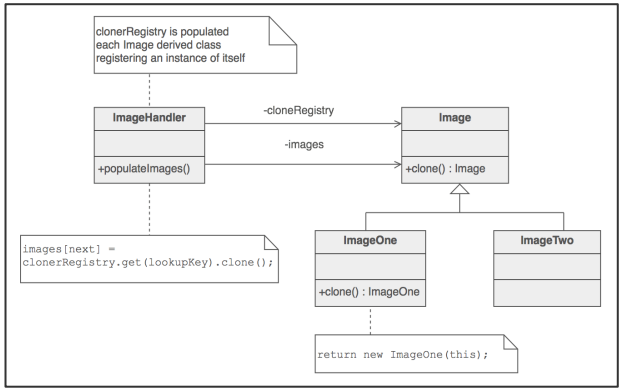
Singleton



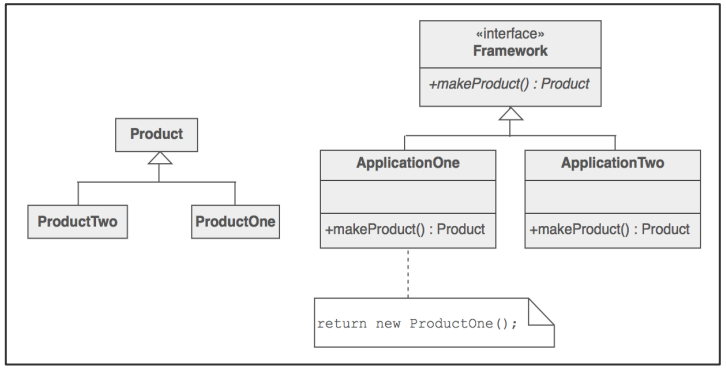
Object Pool



Prototype



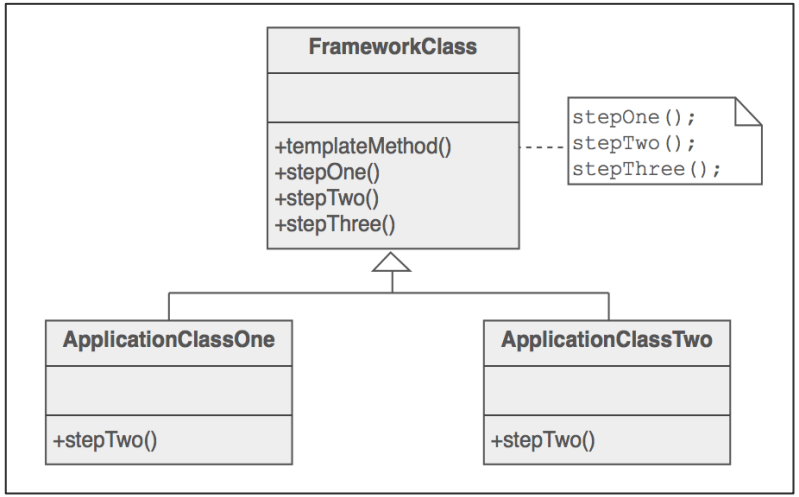
Factory



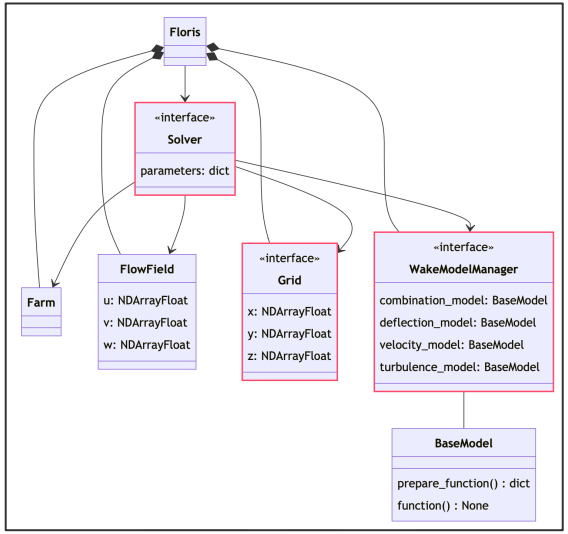
https://sourcemaking.com/design_patterns

Invest and scale

Abstract template design pattern



FLORIS template design pattern



1:1 → **1:N**

Designed components of software

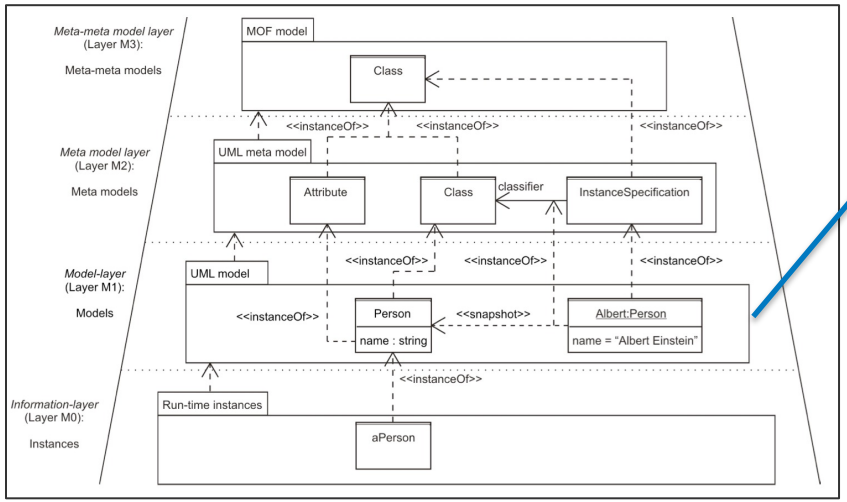
Component	Developer	User	Program Manager
System-level architecture	x	x	x
Package, module, subroutine architecture	x		
Design principles, overarching themes	x	x	x
Data structures and data flow	x	x	
Effective usage workflows		x	
Developer coordination and processes	x		

UML: Unified Modeling Language

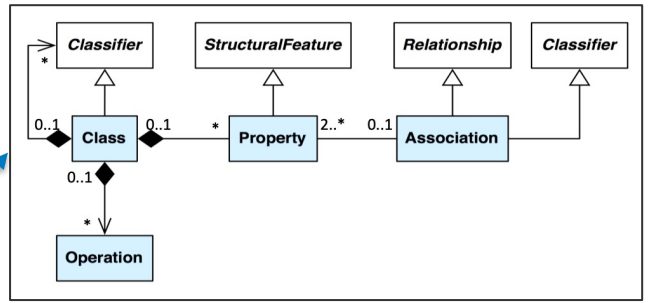
UML: Unified Modeling Language

The Unified Modeling Language (UML) is a family of graphical notations, backed by single metamodel, that help in describing and designing software systems, particularly software systems built using the object-oriented (OO) style.

Meta Object Facility (MOF)



Class Diagram Model



UML Distilled

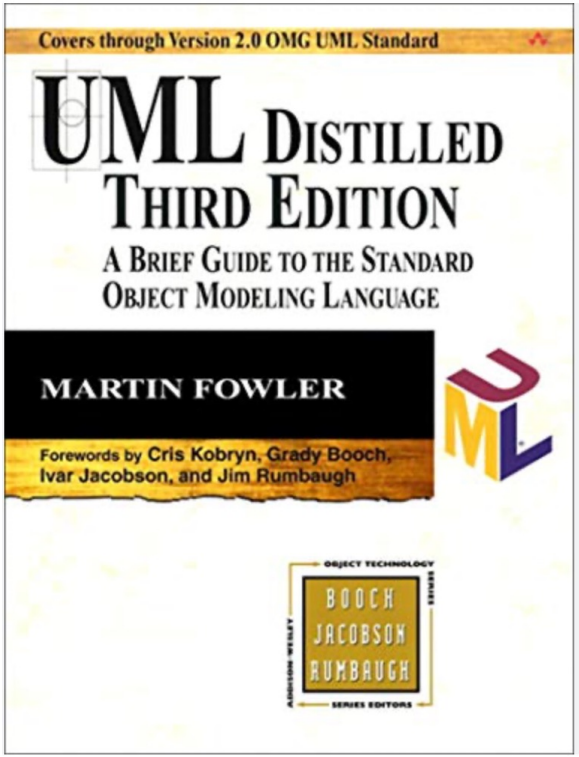


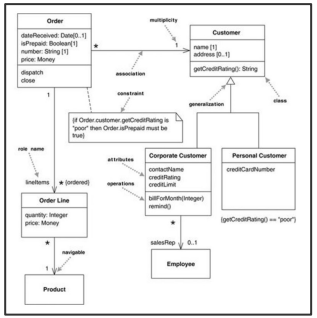
Table of Contents

Key features:

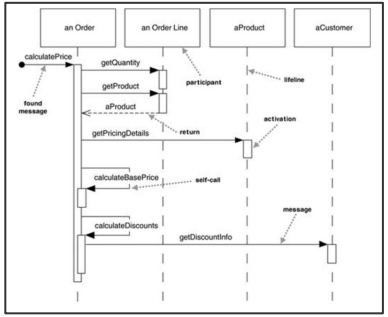
- *Development Processes* chapter
- *When to use X* for each diagram
- Short: 178 pages total
- Acts as a Reference and Explanation
- 1st Edition has a chapter on design

UML: Unified Modeling Language

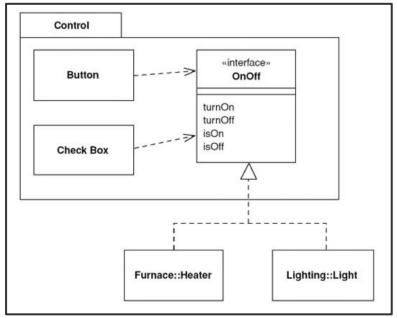
Class



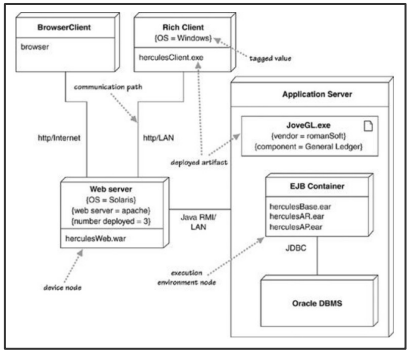
Sequence



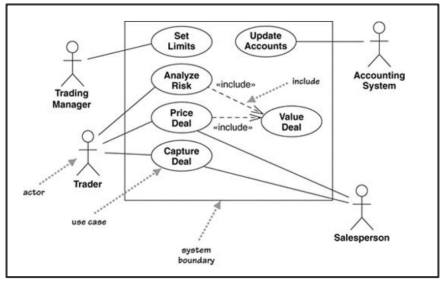
Package



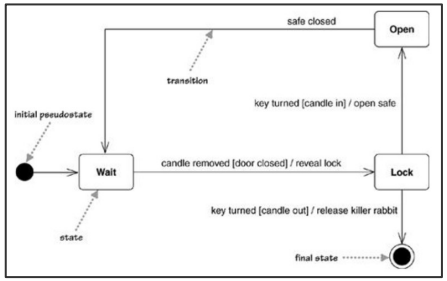
Deployment



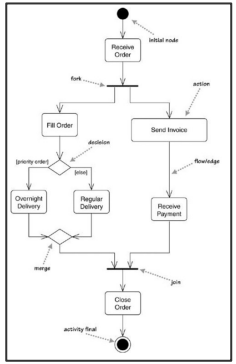
Use Case



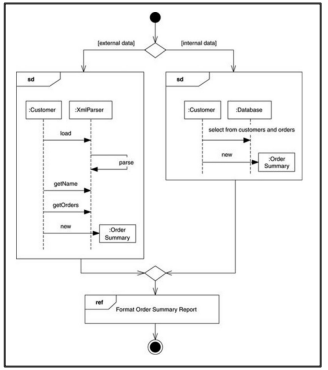
State



Activity

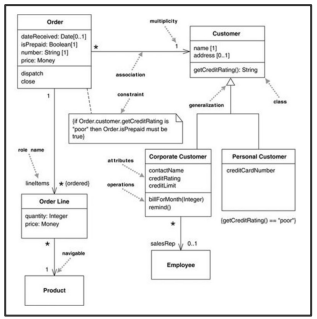


Interaction

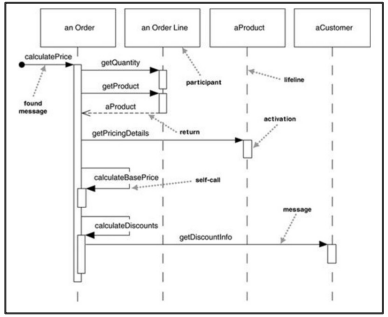


UML: Unified Modeling Language

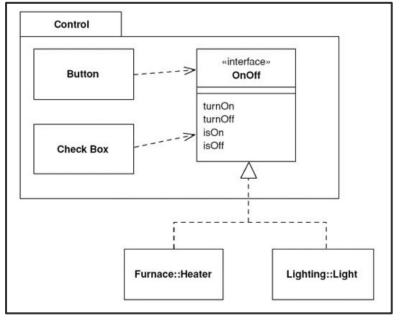
Class



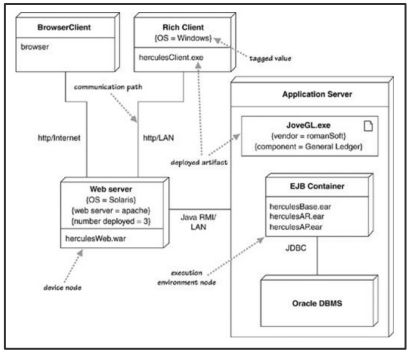
Sequence



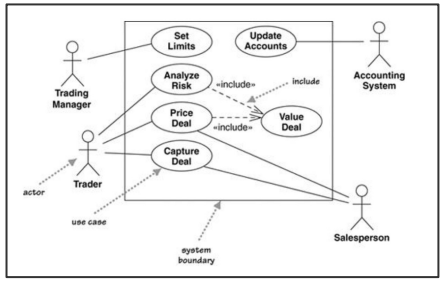
Package



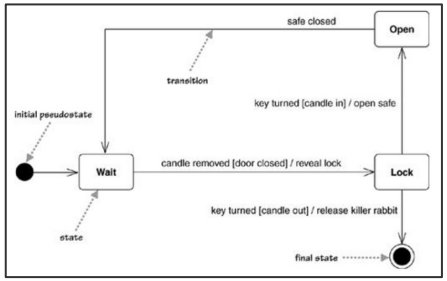
Deployment



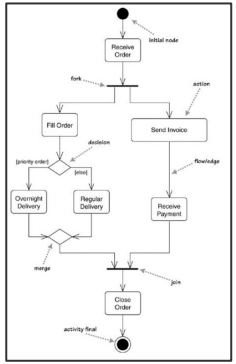
Use Case



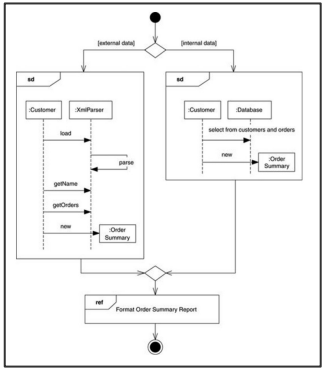
State



Activity

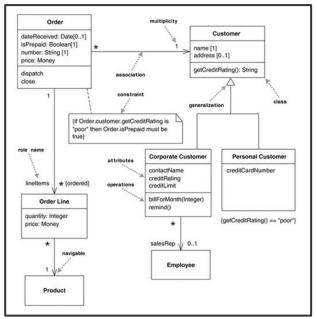


Interaction

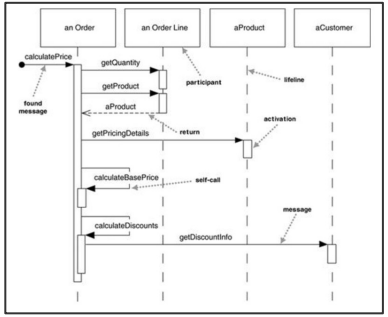


UML: Unified Modeling Language

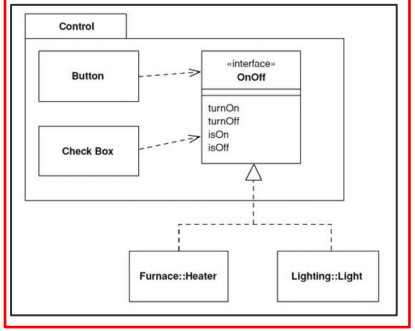
Class



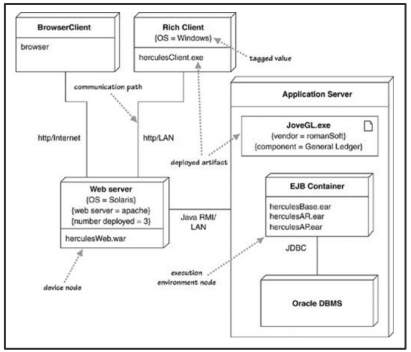
Sequence



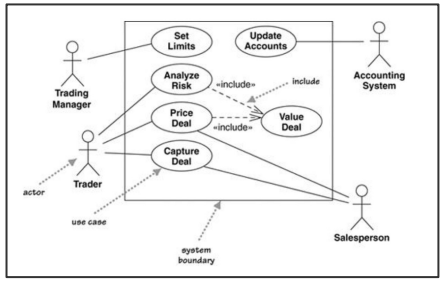
Package



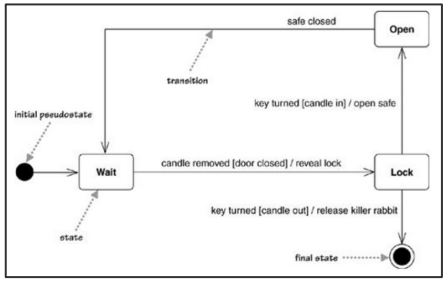
Deployment



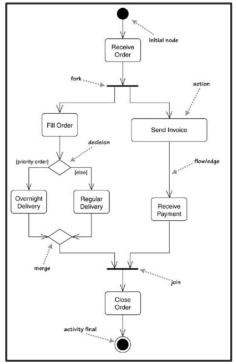
Use Case



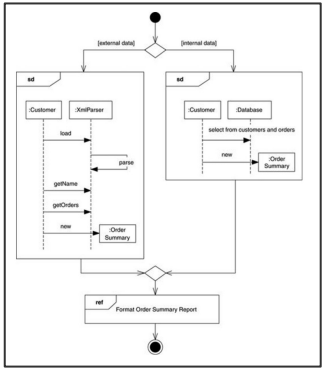
State



Activity

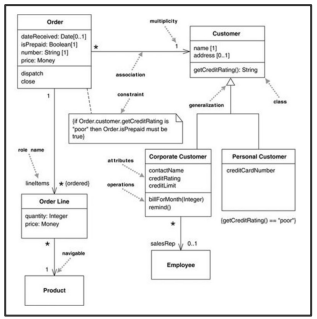


Interaction

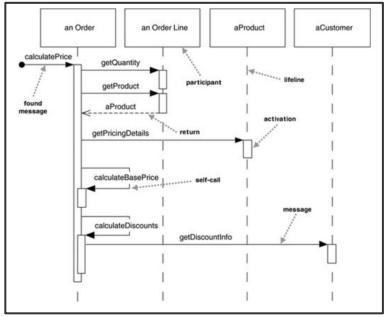


UML: Unified Modeling Language

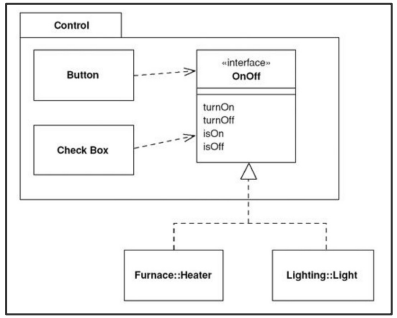
Class



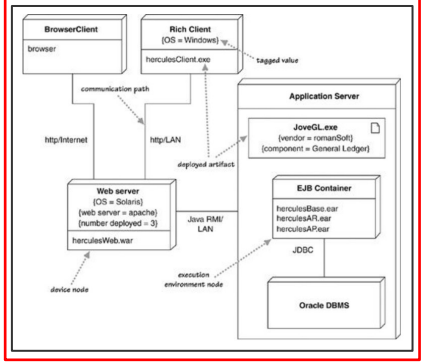
Sequence



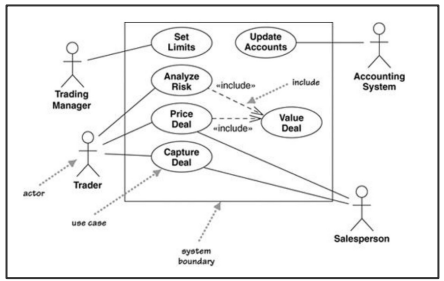
Package



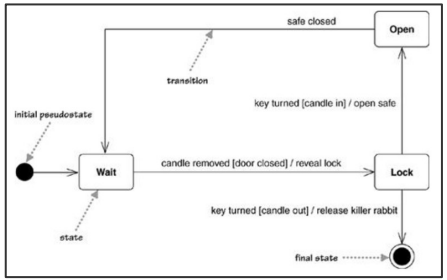
Deployment



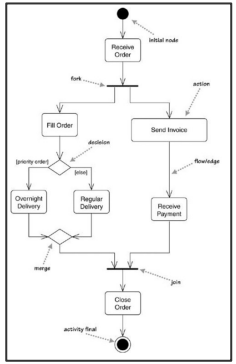
Use Case



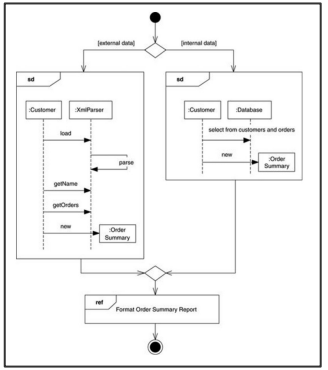
State



Activity

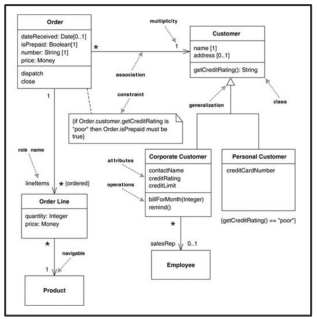


Interaction

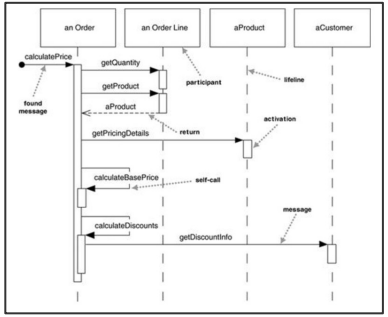


UML: Unified Modeling Language

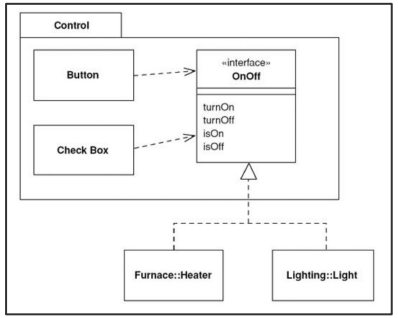
Class



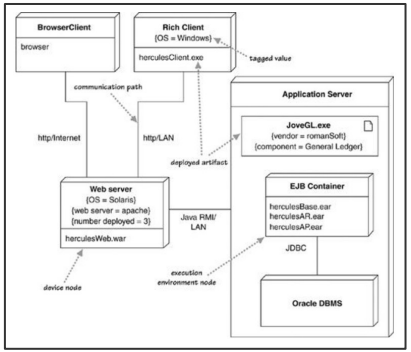
Sequence



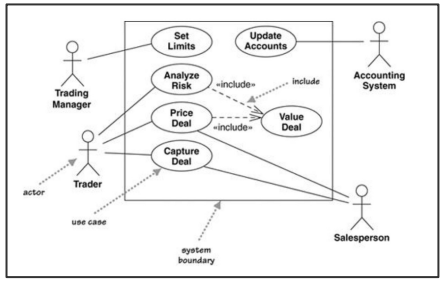
Package



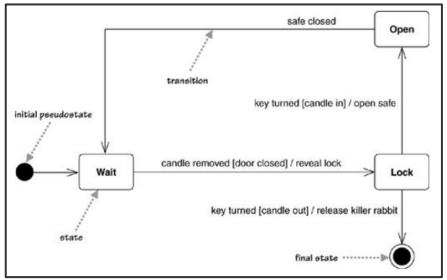
Deployment



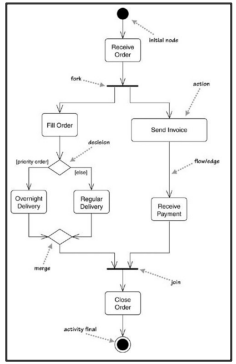
Use Case



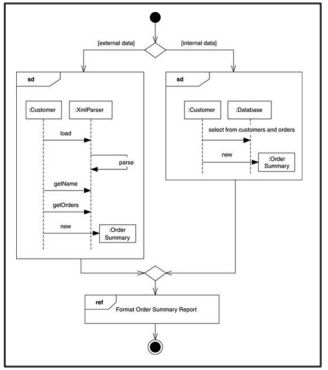
State



Activity

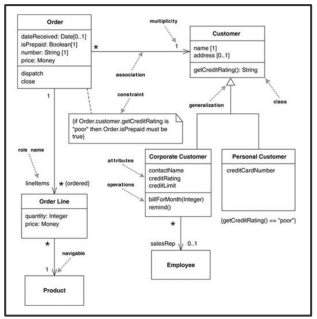


Interaction

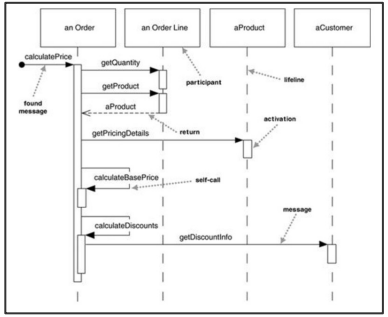


UML: Unified Modeling Language

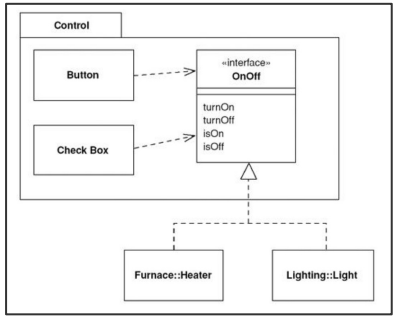
Class



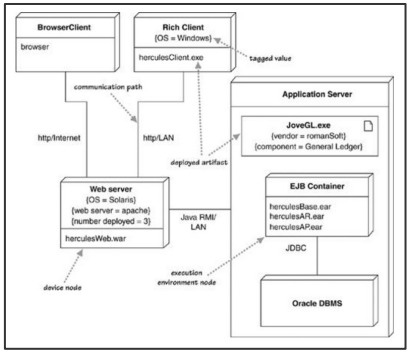
Sequence



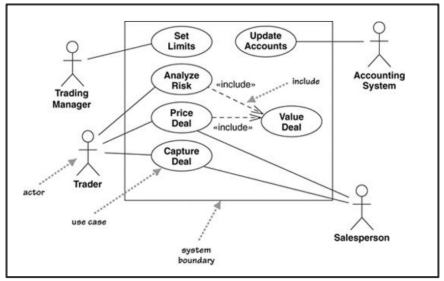
Package



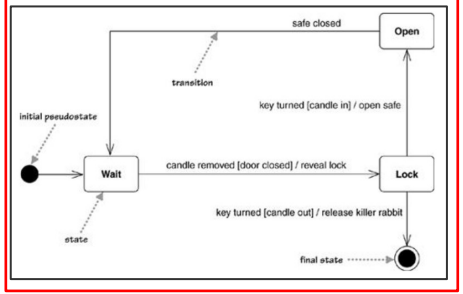
Deployment



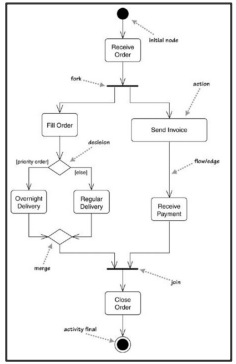
Use Case



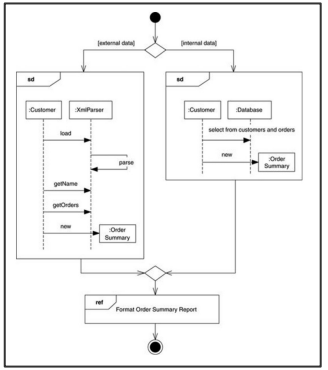
State



Activity

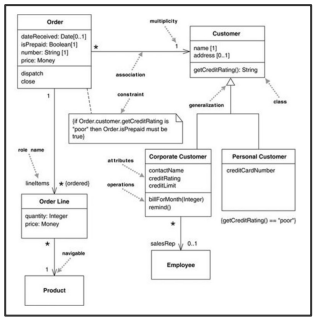


Interaction

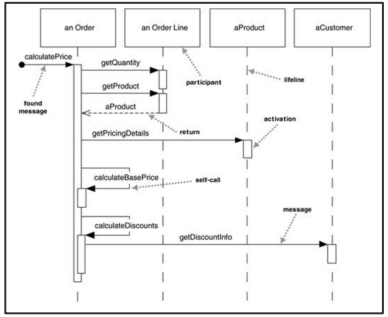


UML: Unified Modeling Language

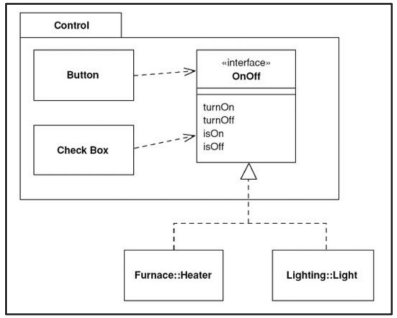
Class



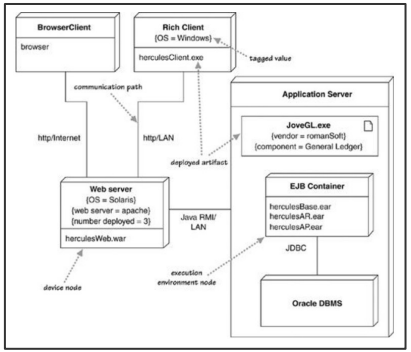
Sequence



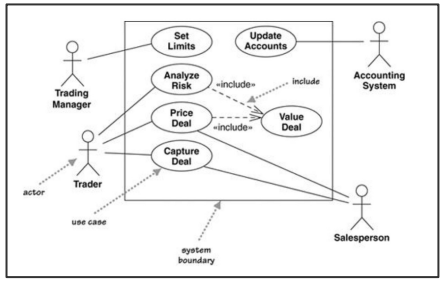
Package



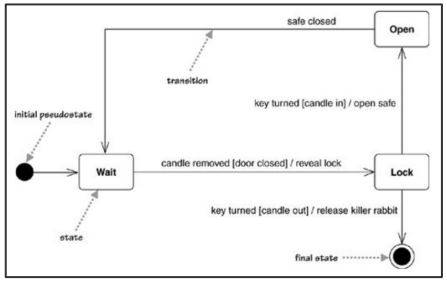
Deployment



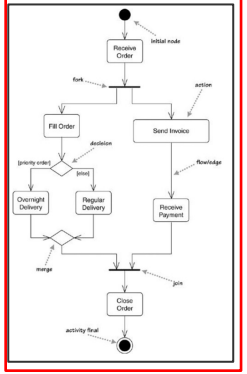
Use Case



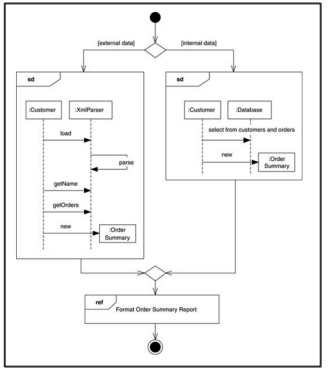
State



Activity

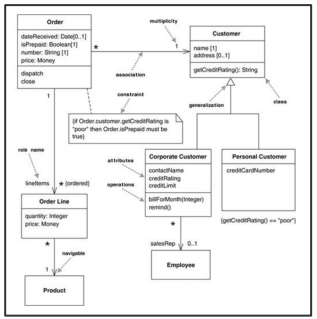


Interaction

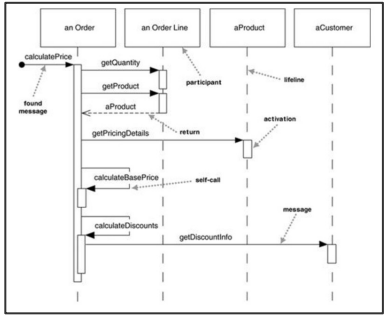


UML: Unified Modeling Language

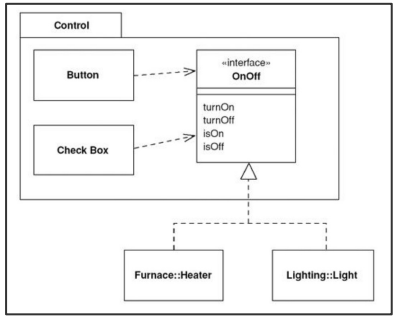
Class



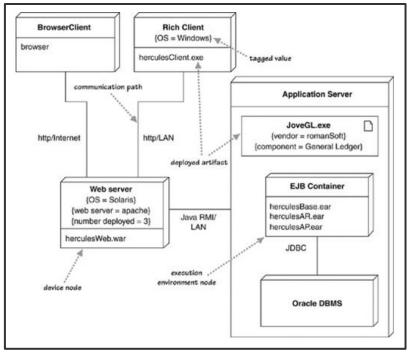
Sequence



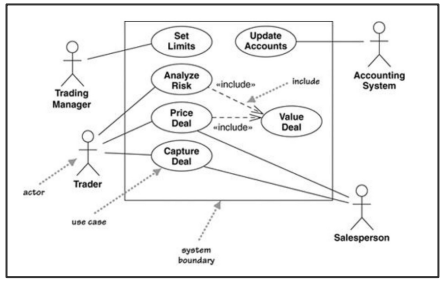
Package



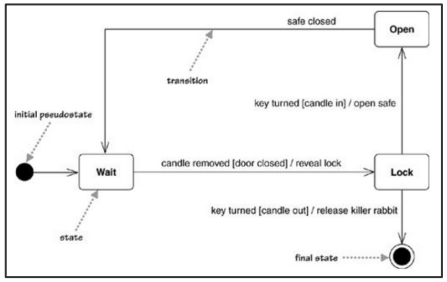
Deployment



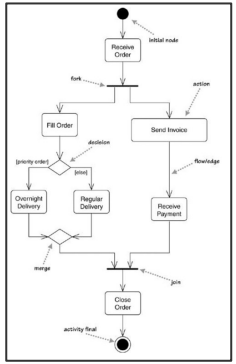
Use Case



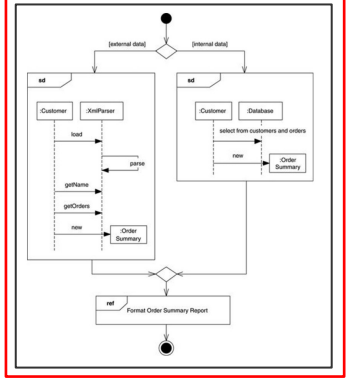
State



Activity

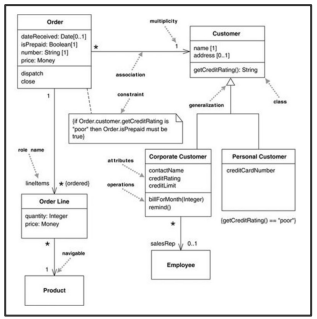


Interaction

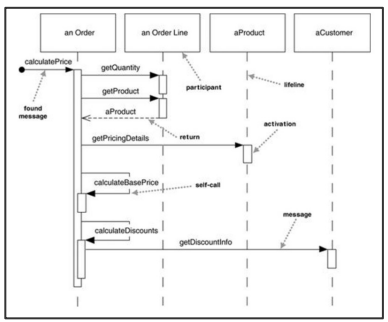


UML: Unified Modeling Language

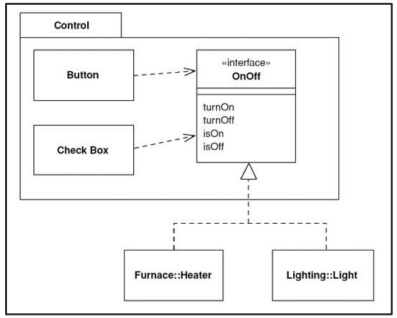
Class



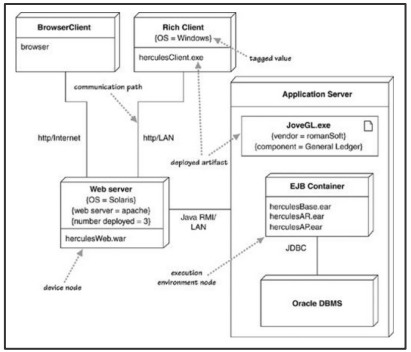
Sequence



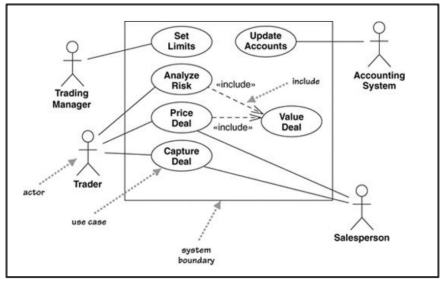
Package



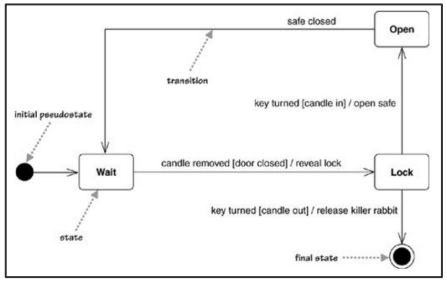
Deployment



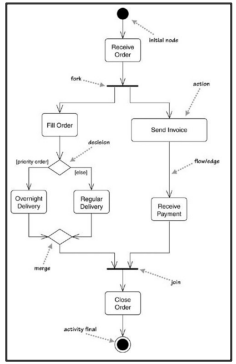
Use Case



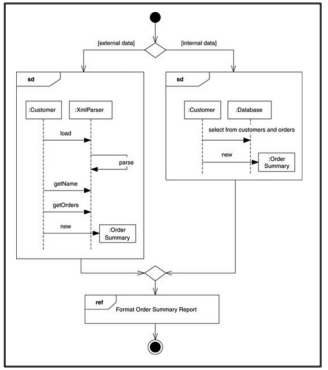
State



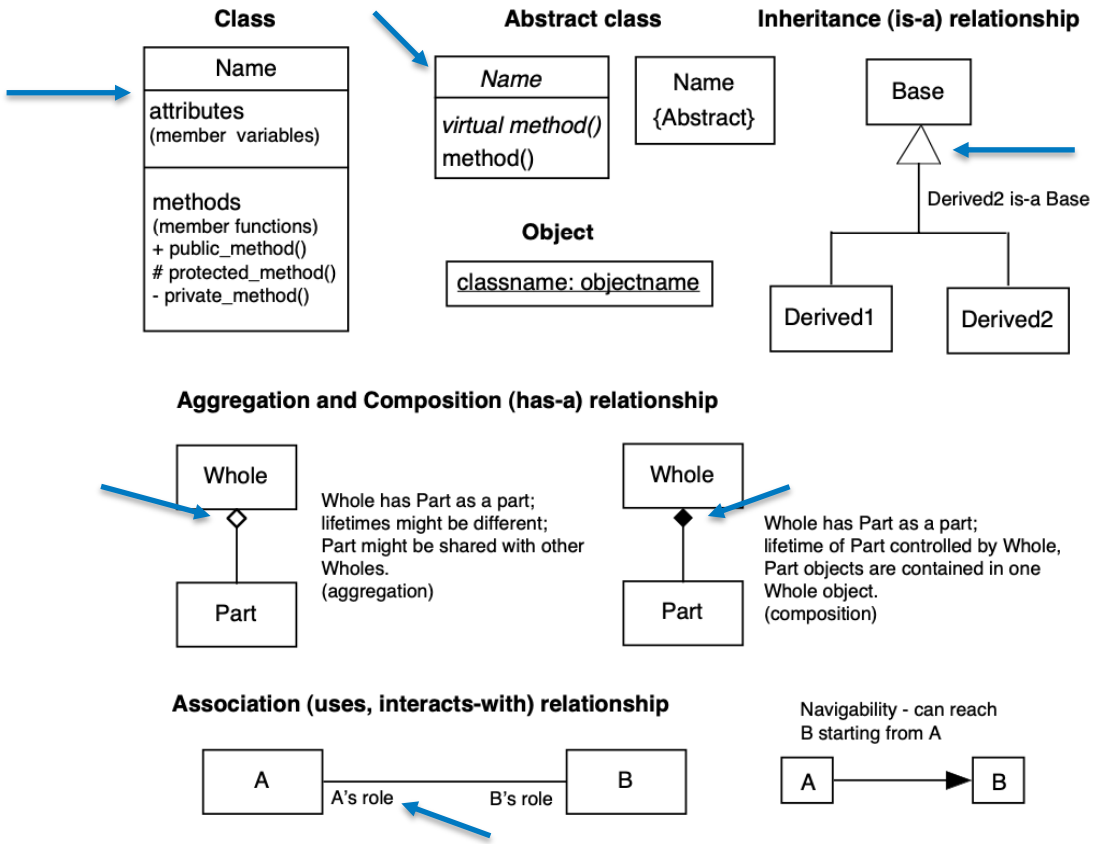
Activity



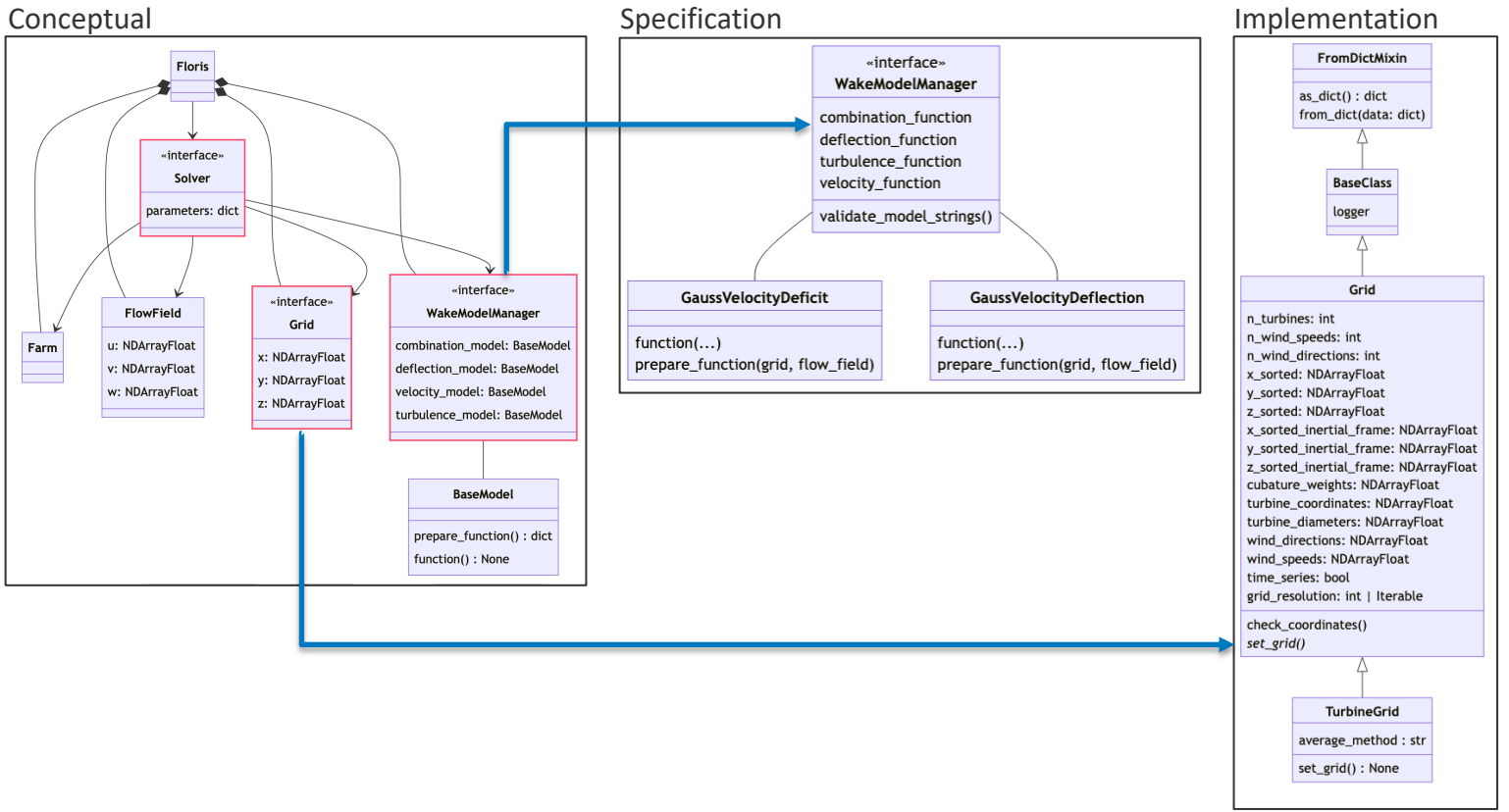
Interaction



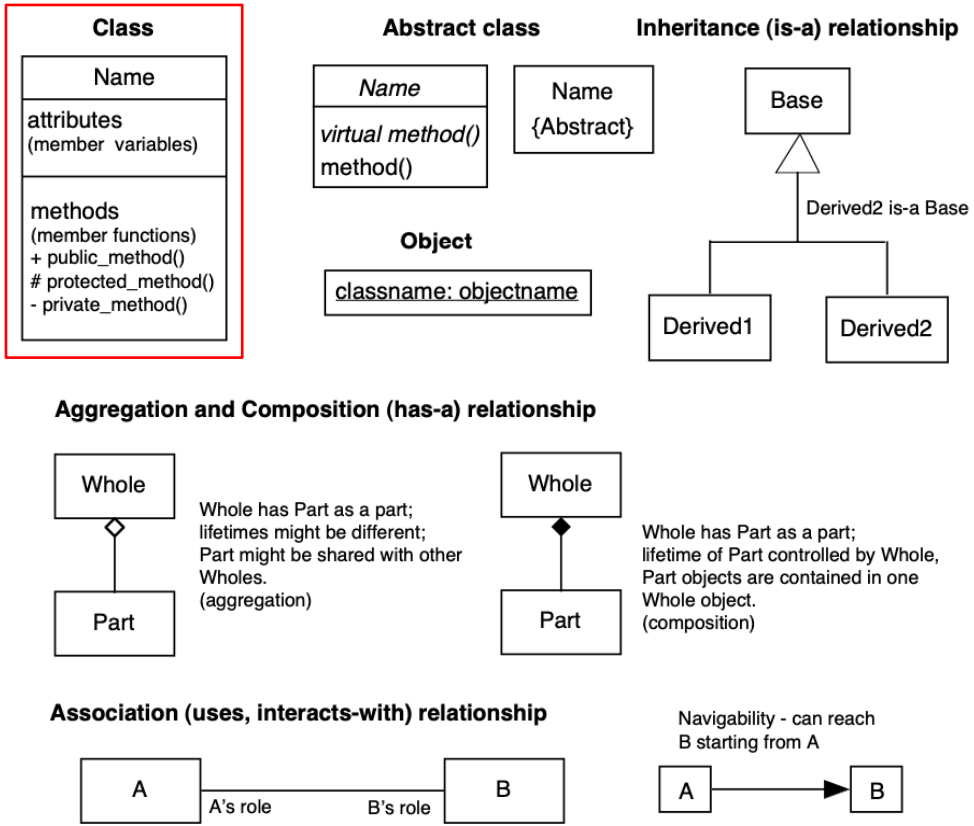
UML: Class Diagram Model



UML: A note on perspective



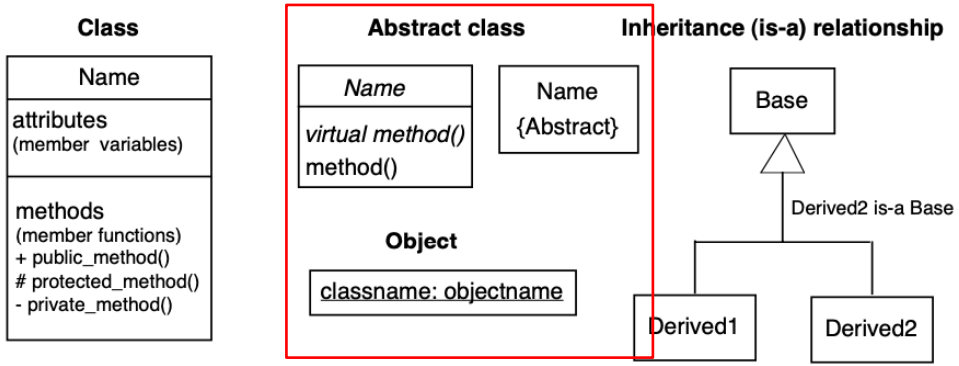
UML: Class Diagram

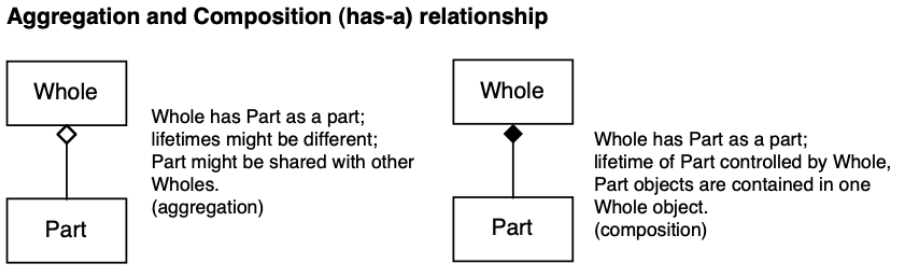


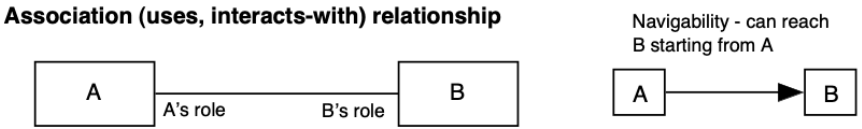
Example

WCompFloris
LEGEND : str LINE_PLOT_COLOR : str LINE_PLOT_LINESTYLE : str LINE_PLOT_MARKER : str deflection_model_string : str deflection_parameters : Optional[dict] fi : FlorisInterface floris_dict : dict hub_height rotor_diameter velocity_deficit_model_string : str velocity_deficit_parameters : dict yaw_angles : ndarray
AEP() horizontal_contour(wind_direction: float, resolution: tuple) : WakePlane streamwise_profile_plot(wind_direction: float, y_coordinate: float, xmin: float, xmax: float) vertical_profile_plot(wind_direction: float, x_coordinate: float, y_coordinate: float, zmax: float) xsection_contour(wind_direction: float, resolution: tuple, x_coordinate: float) : WakePlane xsection_profile_plot(wind_direction: float, x_coordinate: float, ymin: float, ymax: float)

UML: Class Diagram



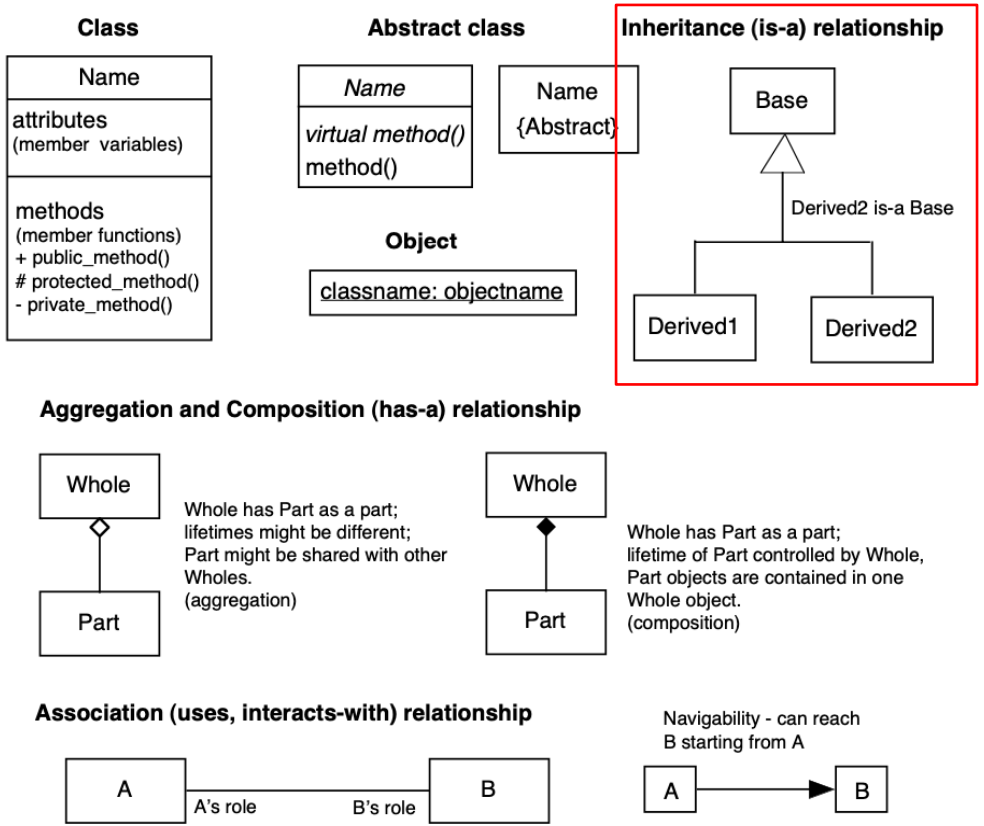




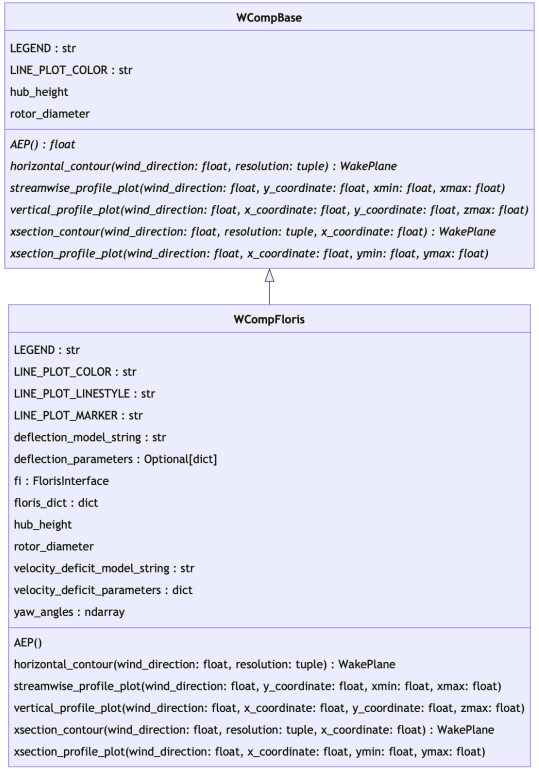
Example

WCompBase
LEGEND : str LINE_PLOT_COLOR : str hub_height rotor_diameter
AEP() : float *horizontal_contour(wind_direction: float, resolution: tuple) : WakePlane* *streamwise_profile_plot(wind_direction: float, y_coordinate: float, xmin: float, xmax: float)* *vertical_profile_plot(wind_direction: float, x_coordinate: float, y_coordinate: float, zmax: float)* *xsection_contour(wind_direction: float, resolution: tuple, x_coordinate: float) : WakePlane* *xsection_profile_plot(wind_direction: float, x_coordinate: float, ymin: float, ymax: float)*

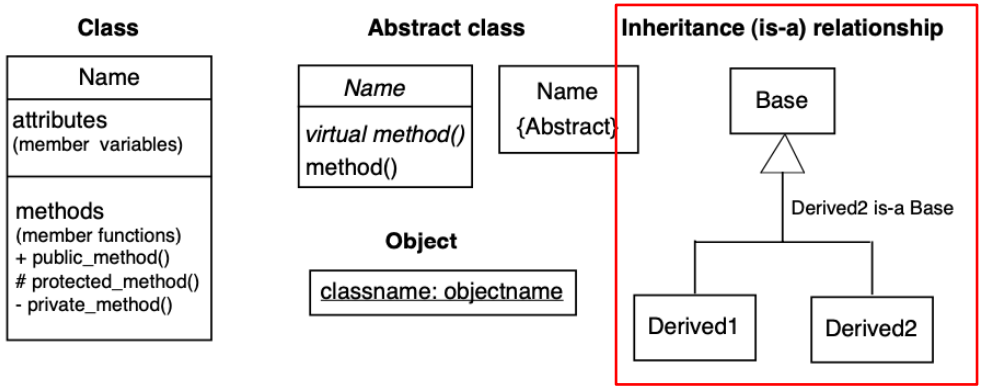
UML: Class Diagram

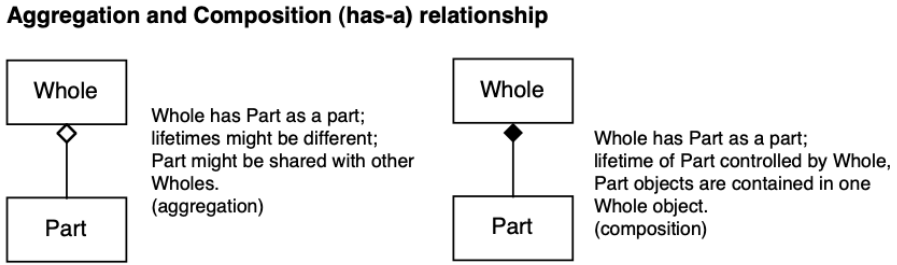


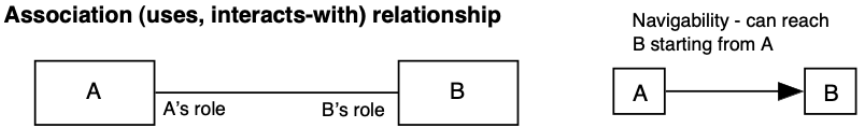
Example



UML: Class Diagram



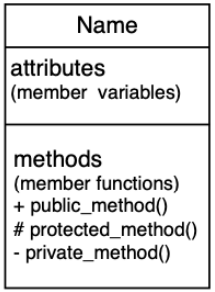




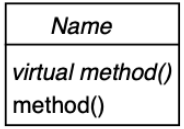
Example

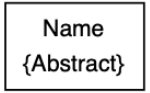
UML: Class Diagram

Class

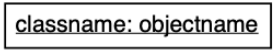


Abstract class

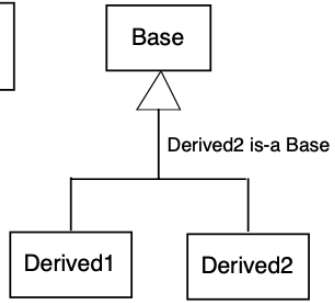




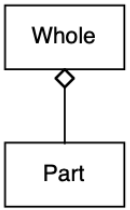
Object



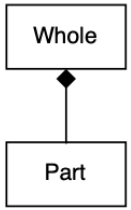
Inheritance (is-a) relationship



Aggregation and Composition (has-a) relationship

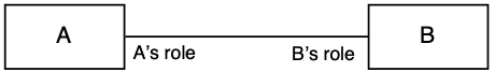


Whole has Part as a part;
lifetimes might be different;
Part might be shared with other
Wholes.
(aggregation)

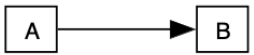


Whole has Part as a part;
lifetime of Part controlled by Whole,
Part objects are contained in one
Whole object.
(composition)

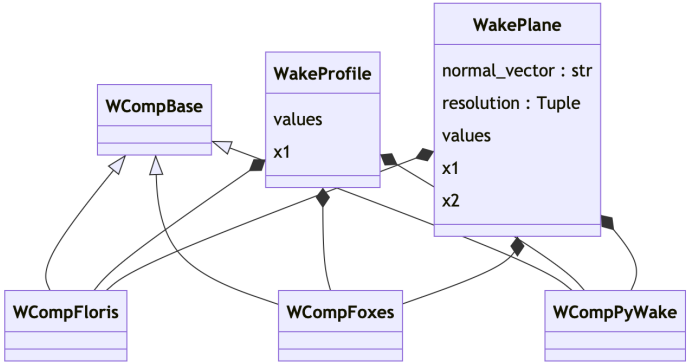
Association (uses, interacts-with) relationship



Navigability - can reach
B starting from A

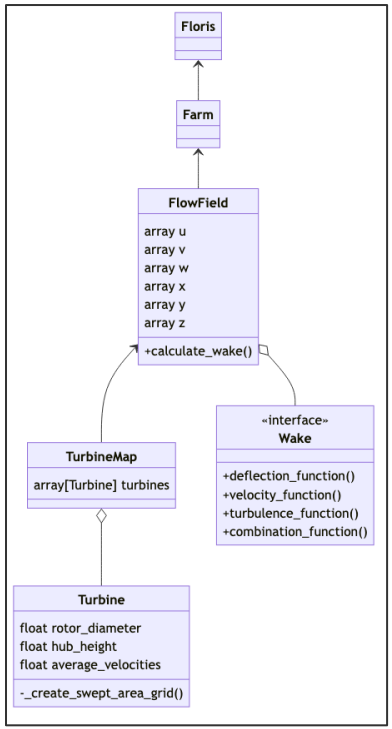


Example

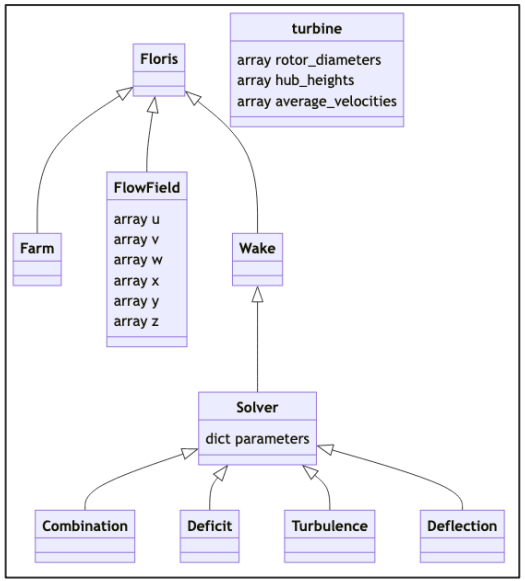


UML: Class Diagram – In Design

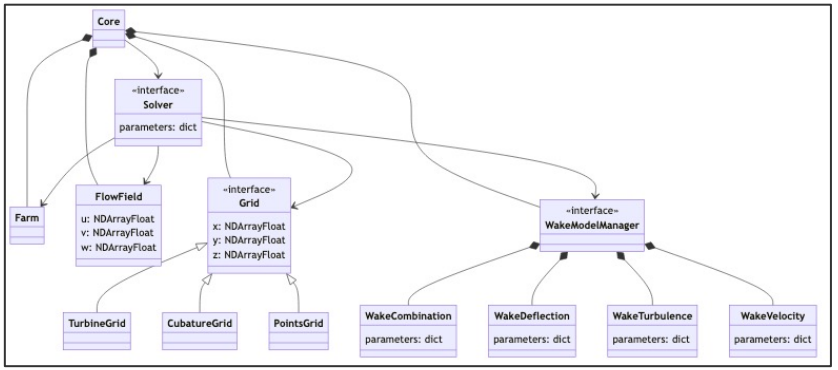
FLORIS v2



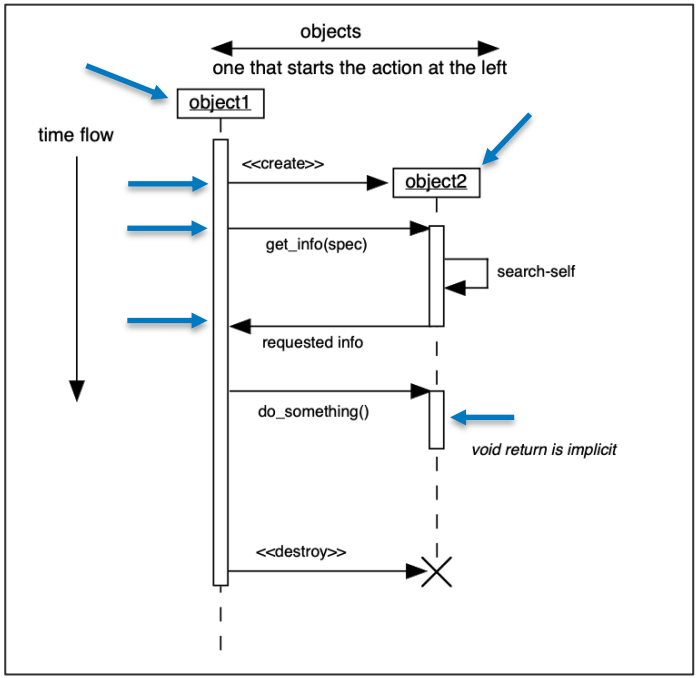
FLORIS v3

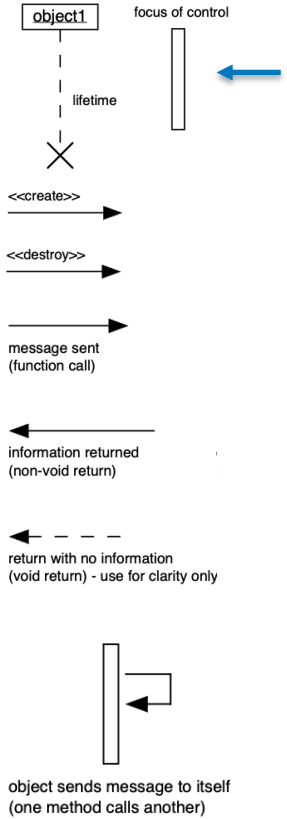


FLORIS v4

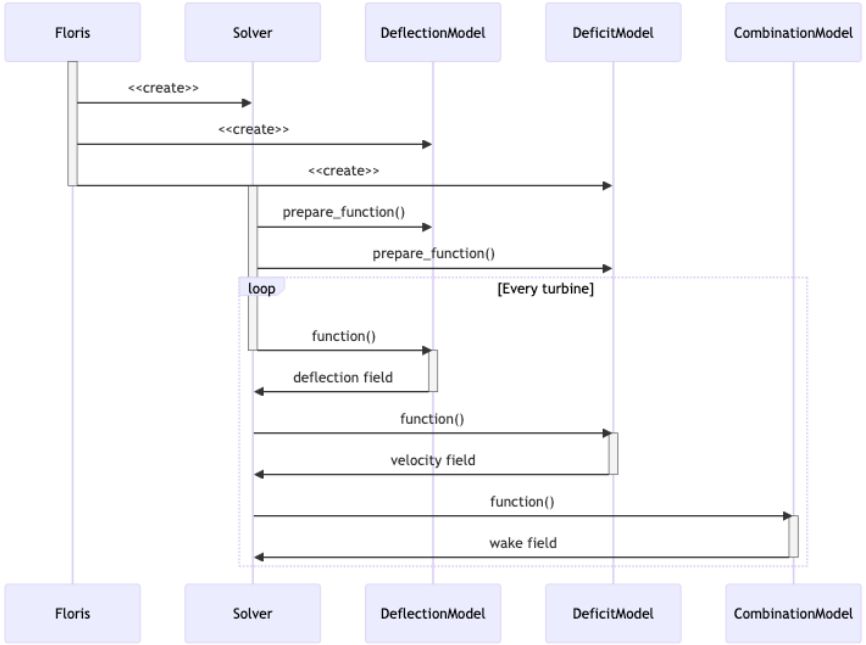


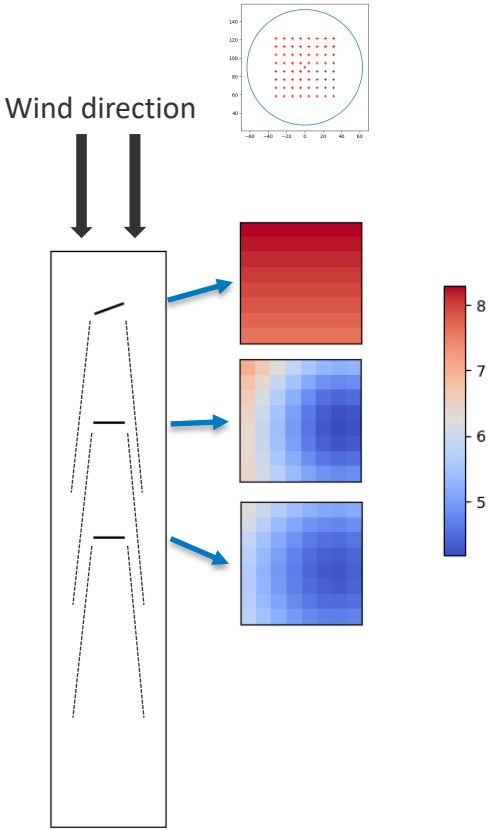
UML: Sequence Diagram Model





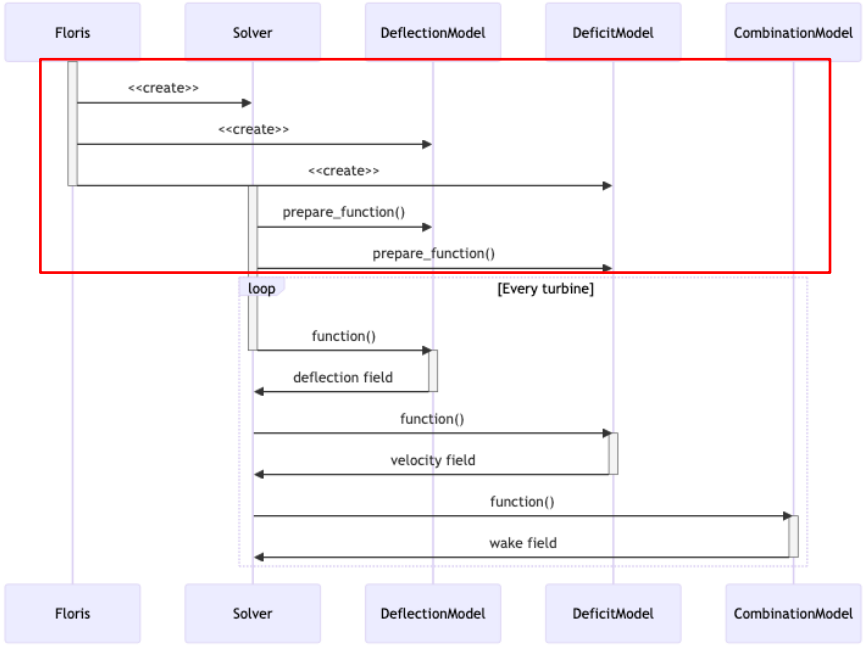
UML: Sequence Diagram

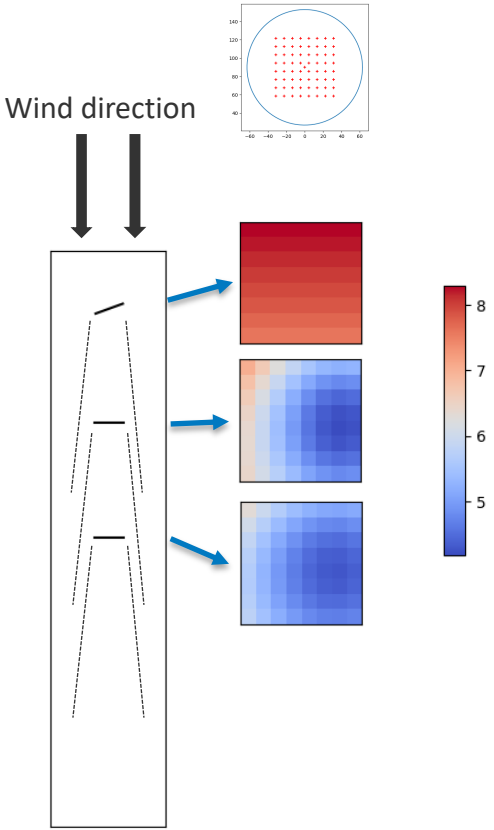




Rotor grid only

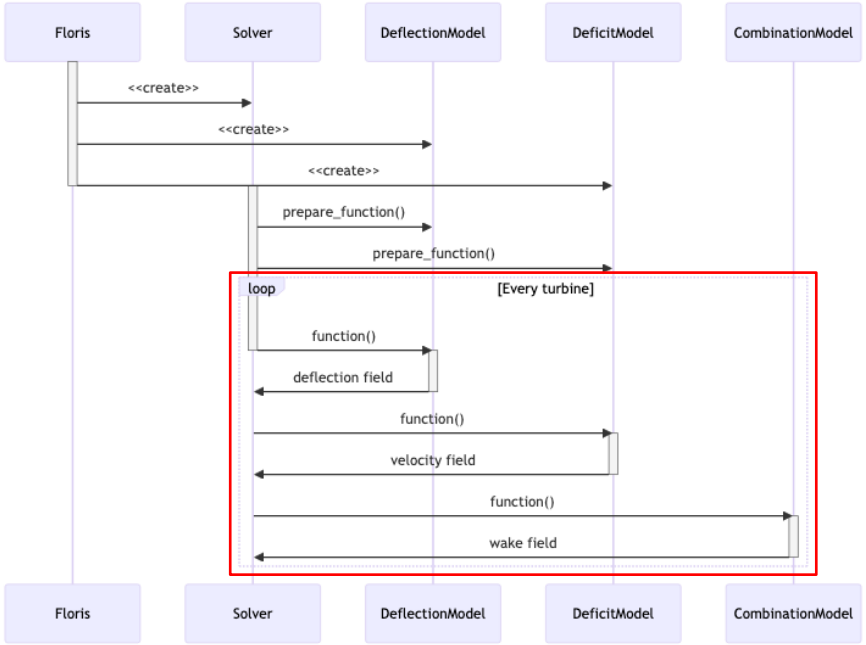
UML: Sequence Diagram

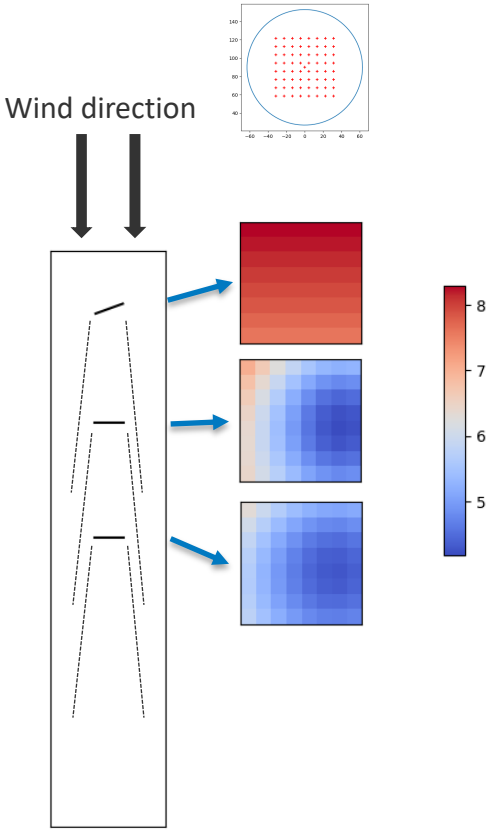




Rotor grid only

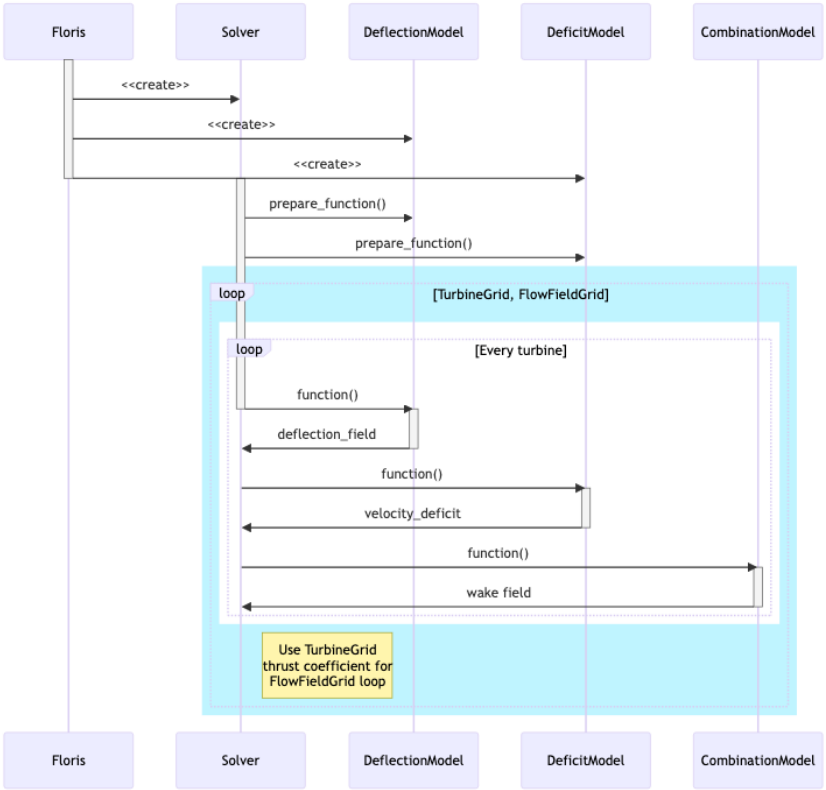
UML: Sequence Diagram

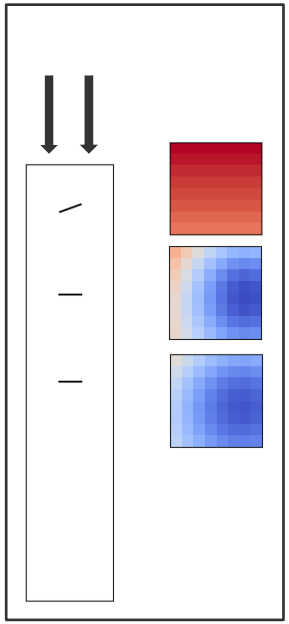




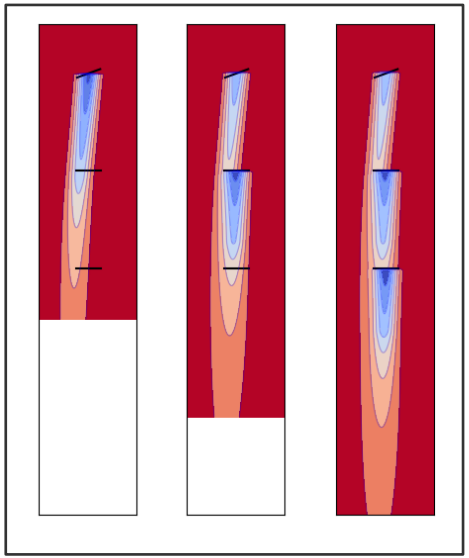
Rotor grid only

UML: Sequence Diagram



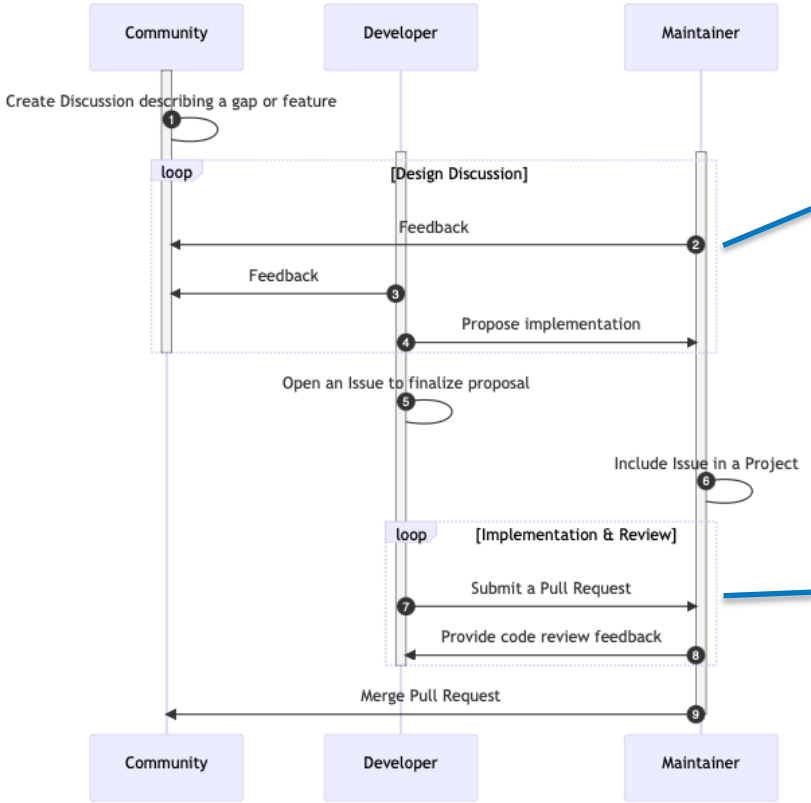


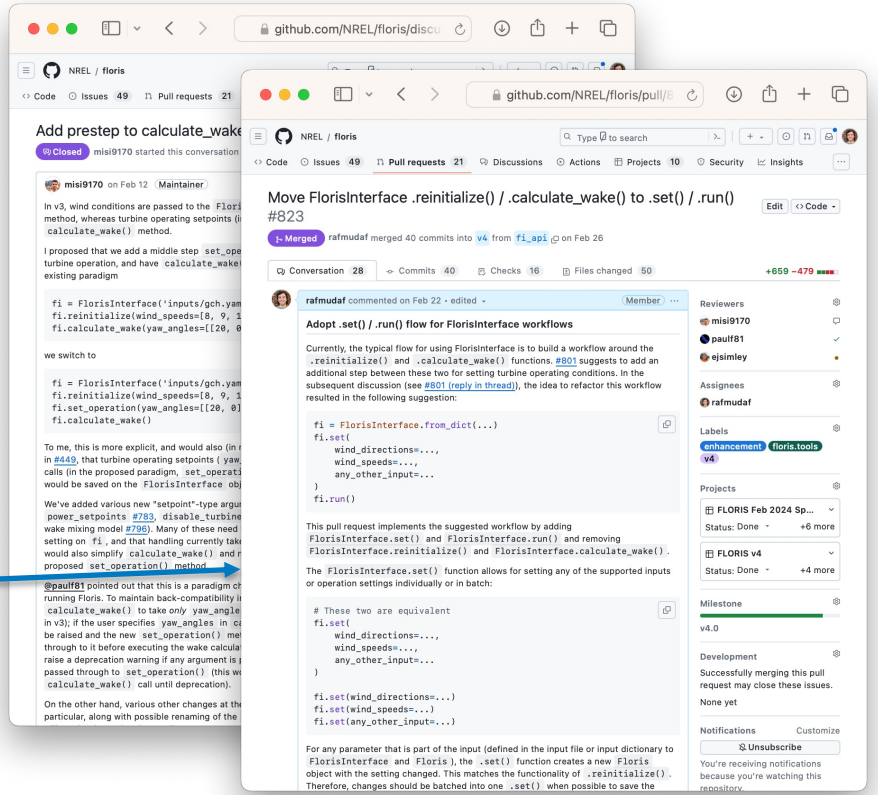
Inner loop



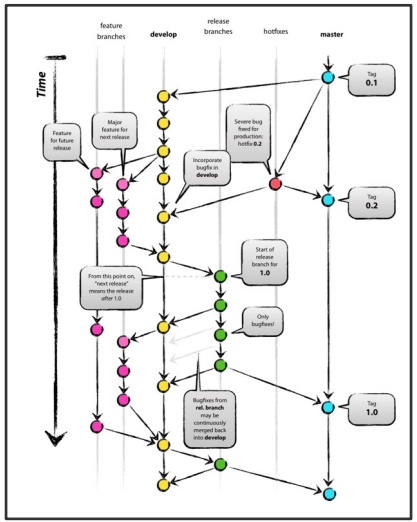
Outer loop

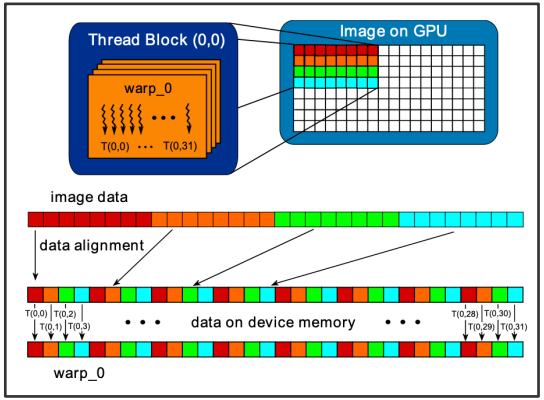
UML: Sequence Diagram

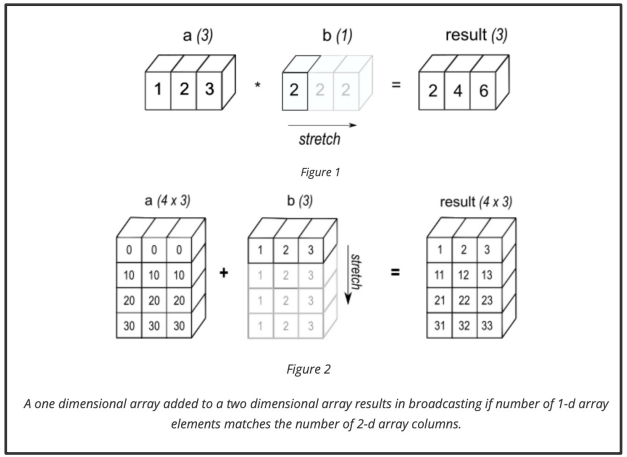




Other diagrams outside of UML







Use UML and anything else to get your message out there!

Diagramming in the development workflow

Documentation Driven Development

Documentation as Code (*Docs as Code*) refers to a philosophy that you should be writing documentation with the same tools as code:

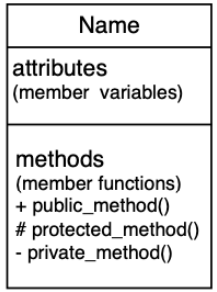
- Issue Trackers
- Version Control (Git)
- Plain Text Markup (Markdown, reStructuredText, Asciidoc)
- Code Reviews
- Automated Tests

This means following the same workflows as development teams, and being integrated in the product team. It enables a culture where writers and developers both feel ownership of documentation, and work together to make it as good as possible.

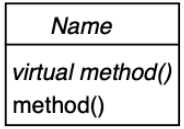
Write the Docs
https://www.writethedocs.org/guide/docs-as-code/

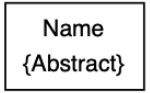
UML: Class Diagrams

Class

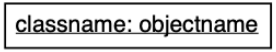


Abstract class

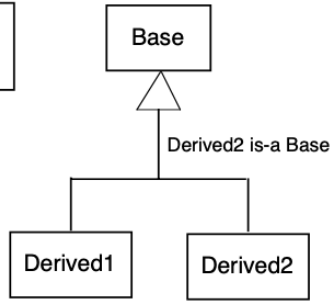




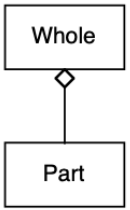
Object



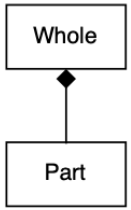
Inheritance (is-a) relationship



Aggregation and Composition (has-a) relationship

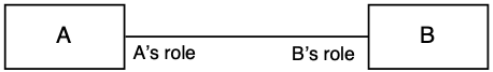


Whole has Part as a part;
lifetimes might be different;
Part might be shared with other
Wholes.
(aggregation)

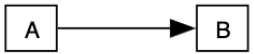


Whole has Part as a part;
lifetime of Part controlled by Whole,
Part objects are contained in one
Whole object.
(composition)

Association (uses, interacts-with) relationship

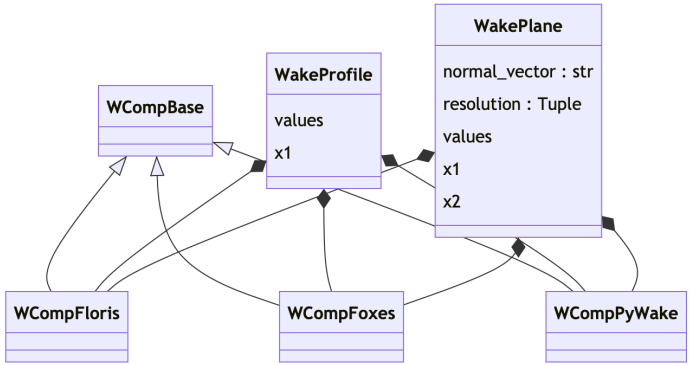


Navigability - can reach
B starting from A



Example

INCORRECT



In Practice: Automated Tools

Treat diagrams like other software infrastructure:

- Incorporate it into the development process
- Build tools, trust them, and lean on them heavily
- Automate – let the computers do the hard work

doxygen

Graphviz

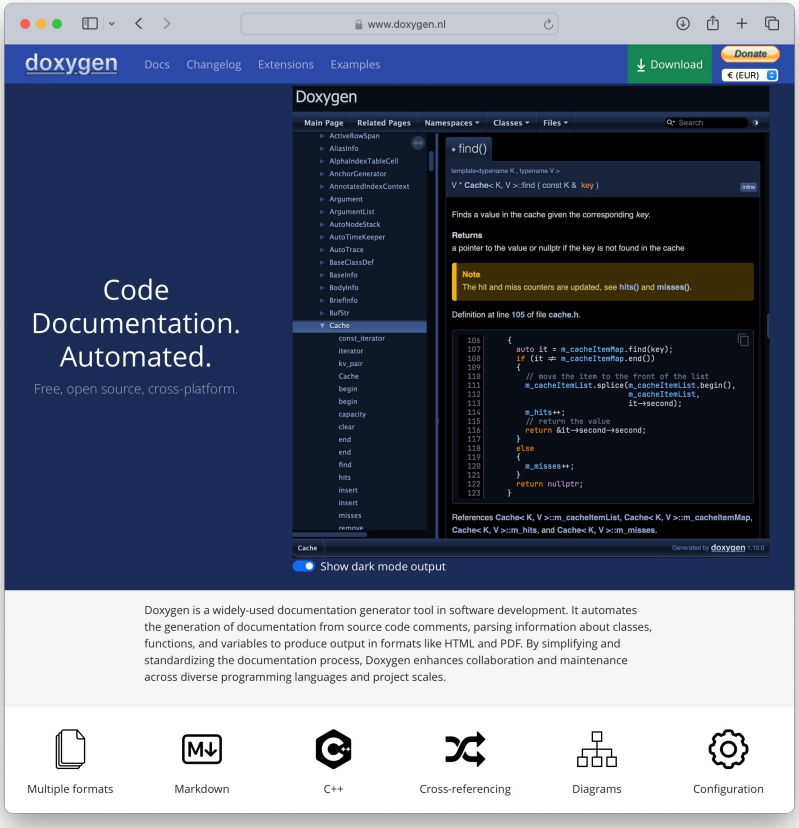
Mermaid

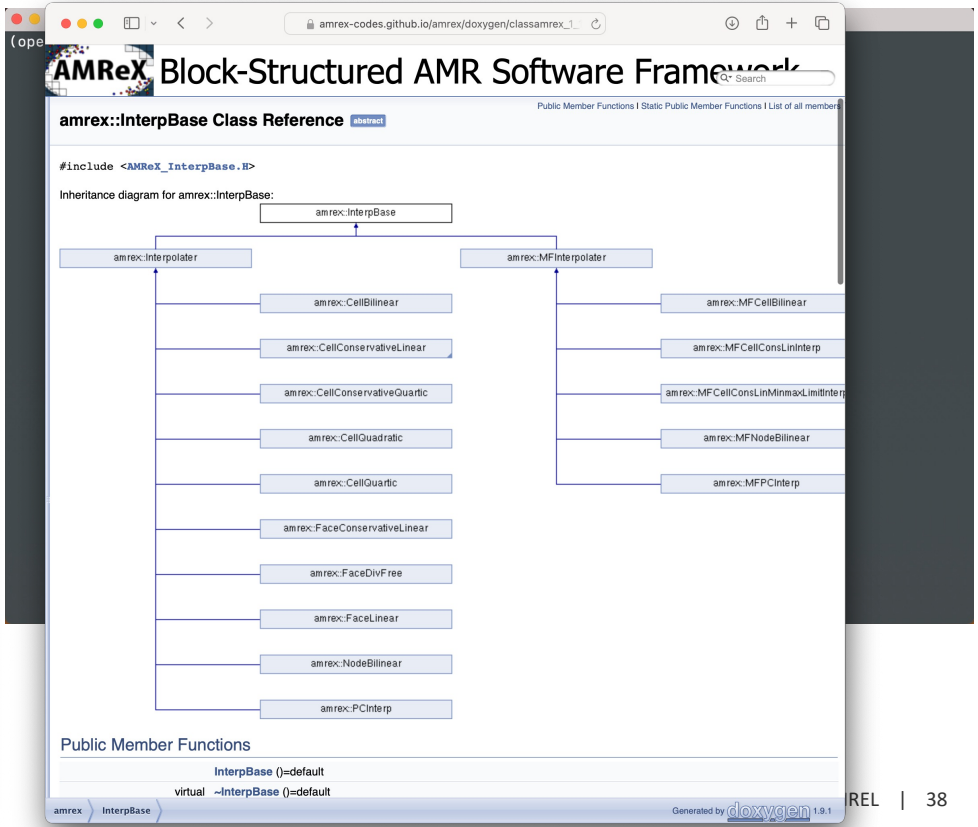
linting pylint

pyreverse

In Practice: Doxygen

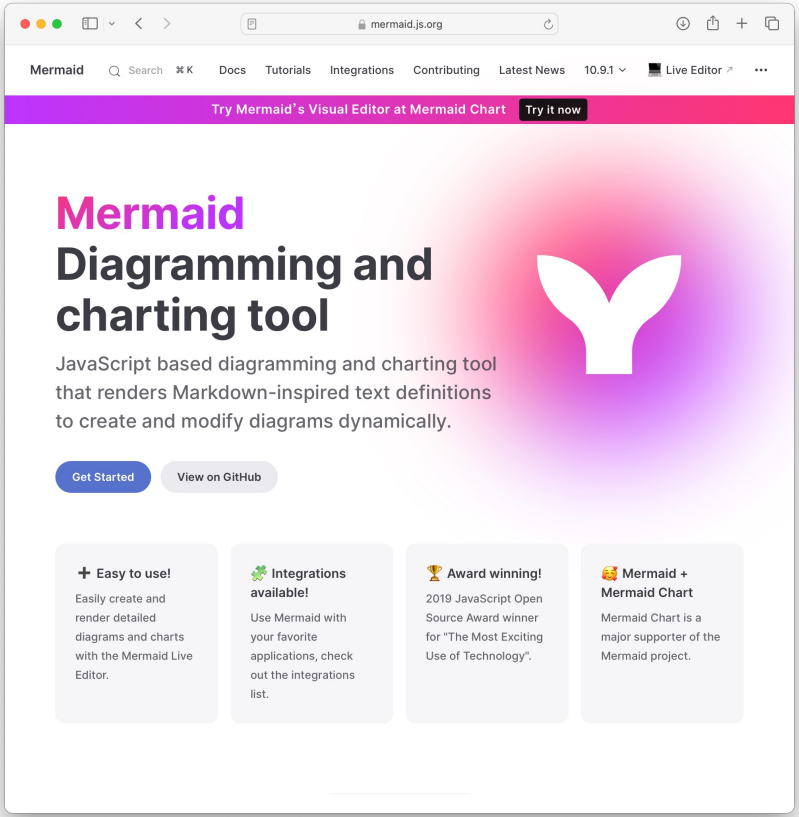
https://www.doxygen.nl

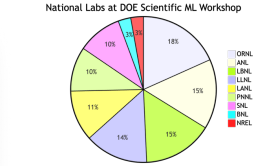


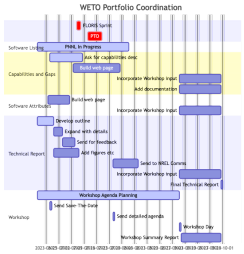


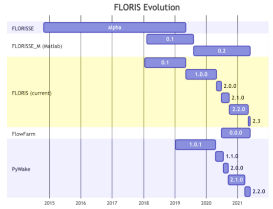
In Practice: Mermaid

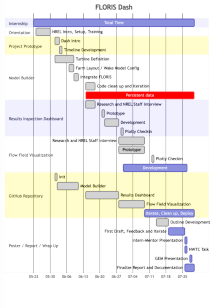
https://mermaid.js.org

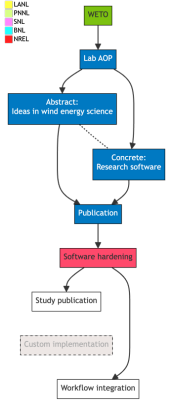


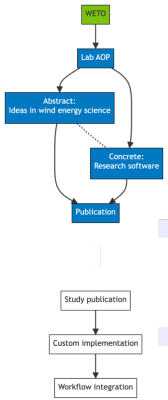


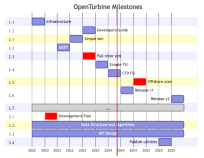


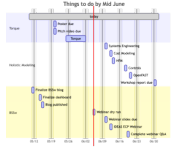


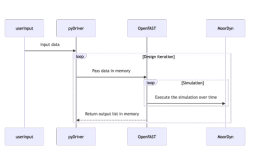


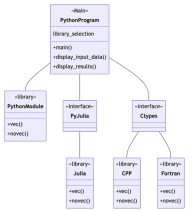


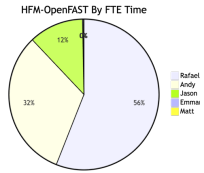


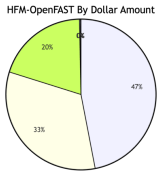


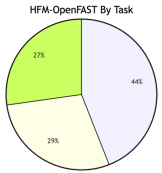


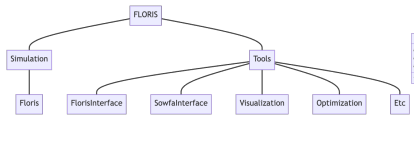


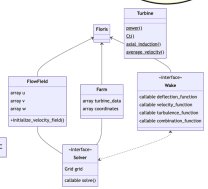


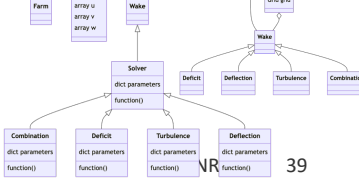






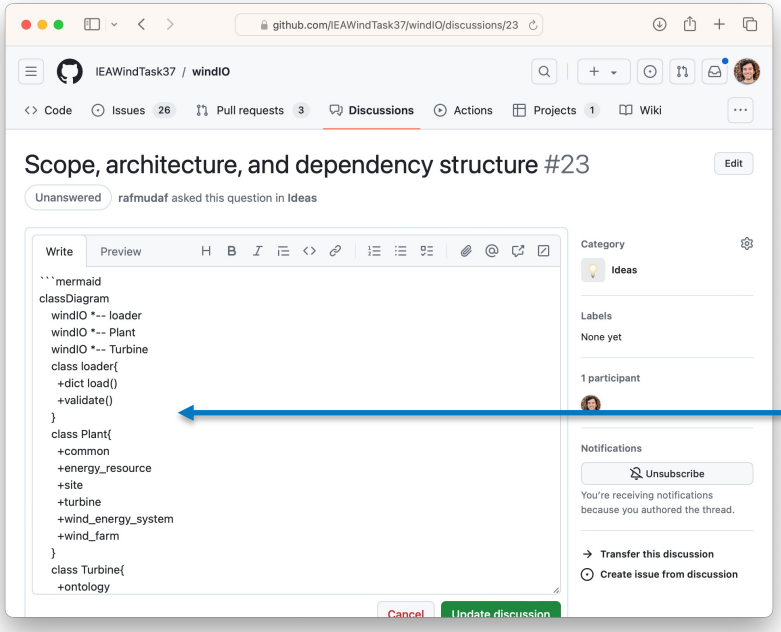


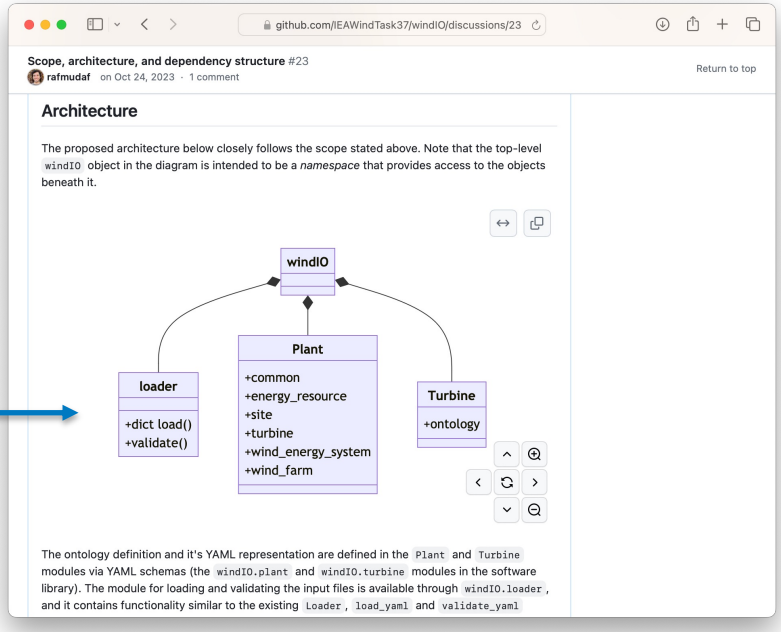




In Practice: Mermaid

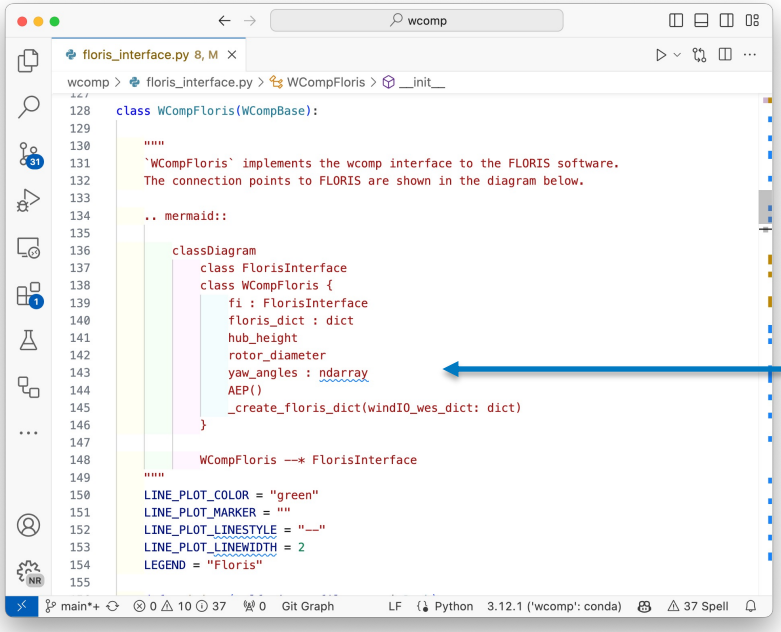
GitHub Discussion – same for any GitHub-flavored Markdown product

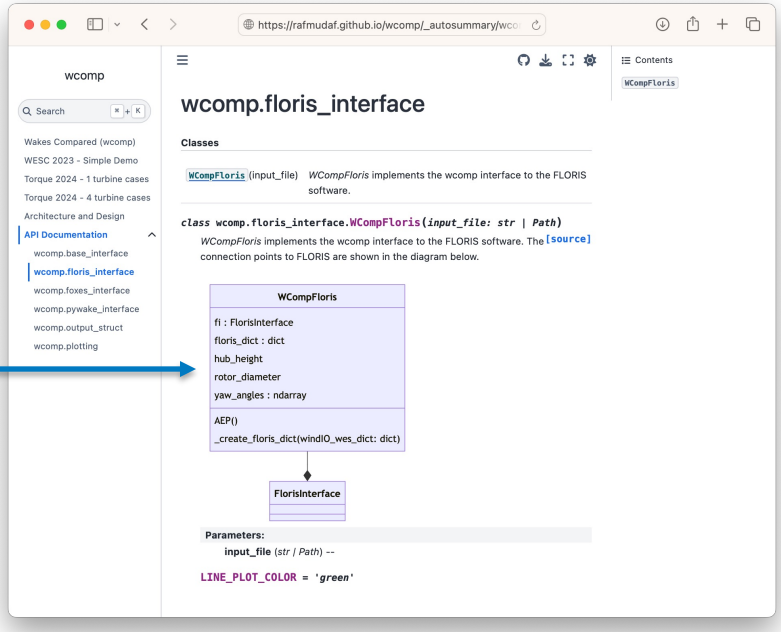




In Practice: Mermaid

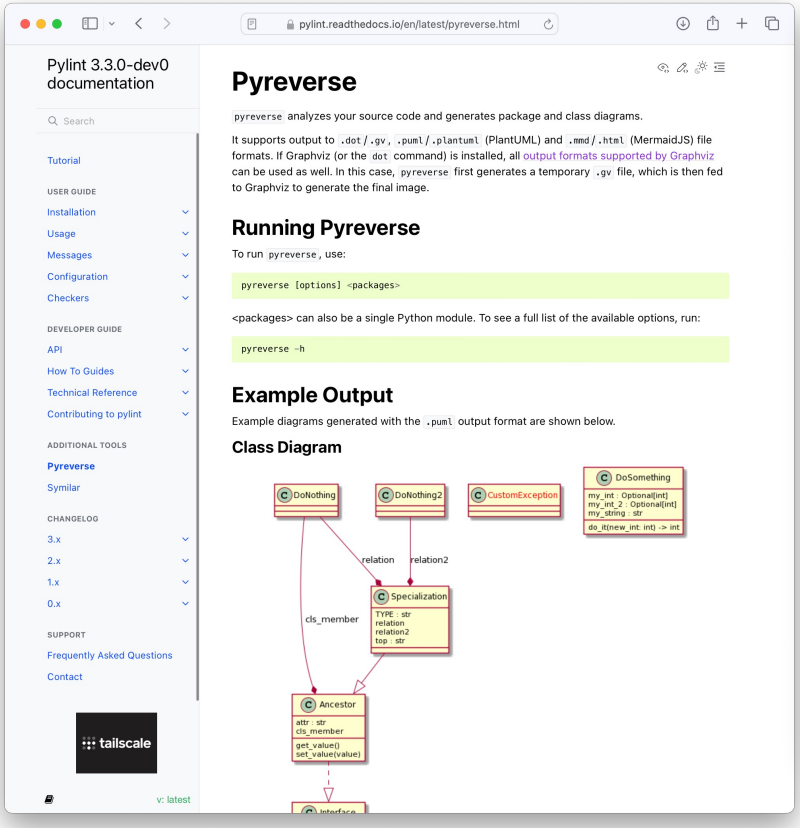
Sphinx-based documentation – add diagrams anywhere including in API docs

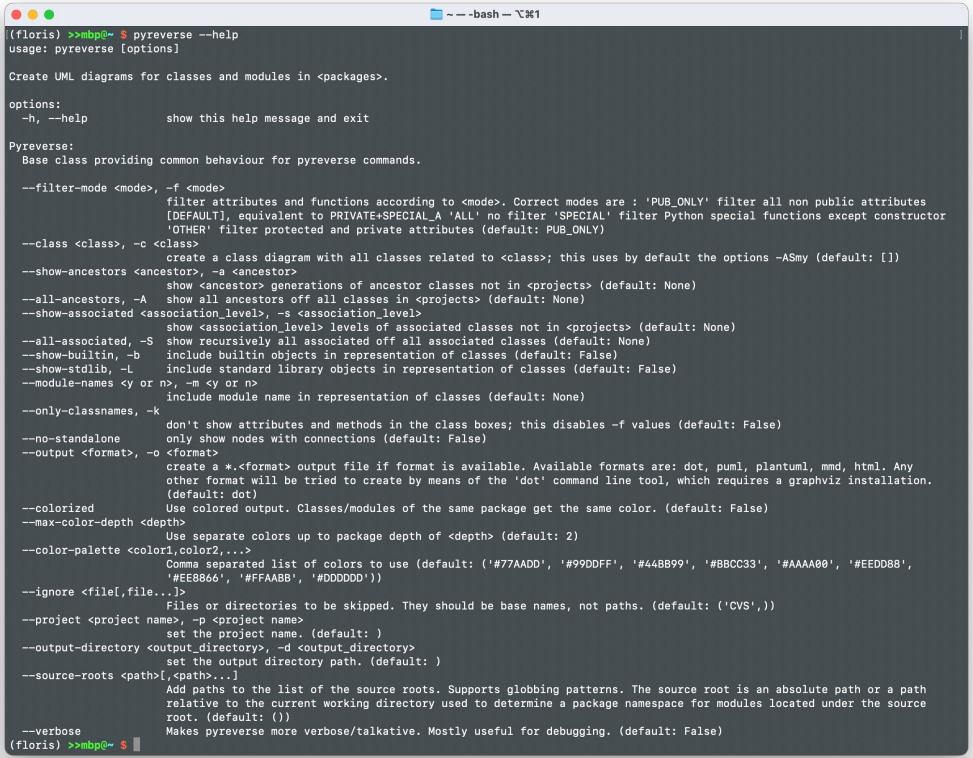




In Practice: pyreverse

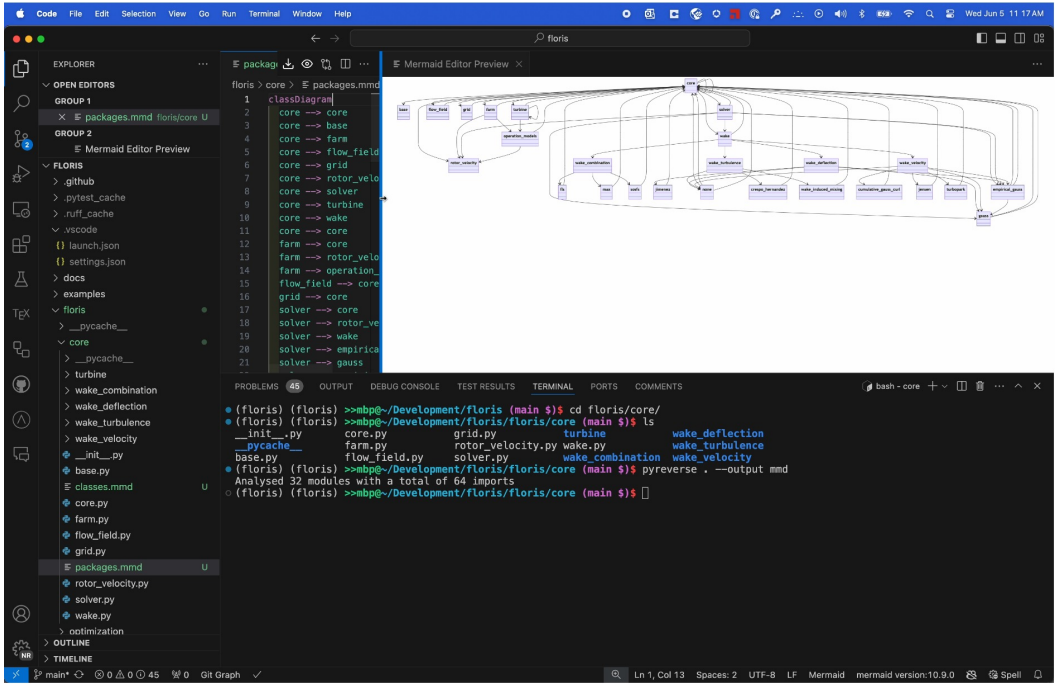
https://pylint.readthedocs.io/en/latest/pyreverse.html

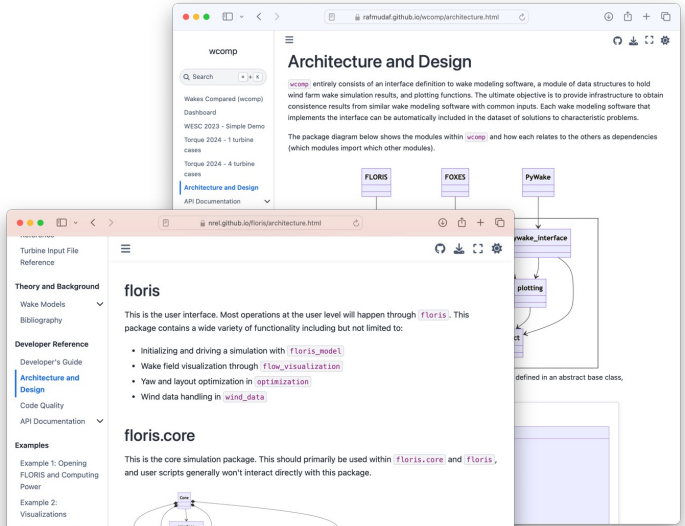




In Practice: pyreverse

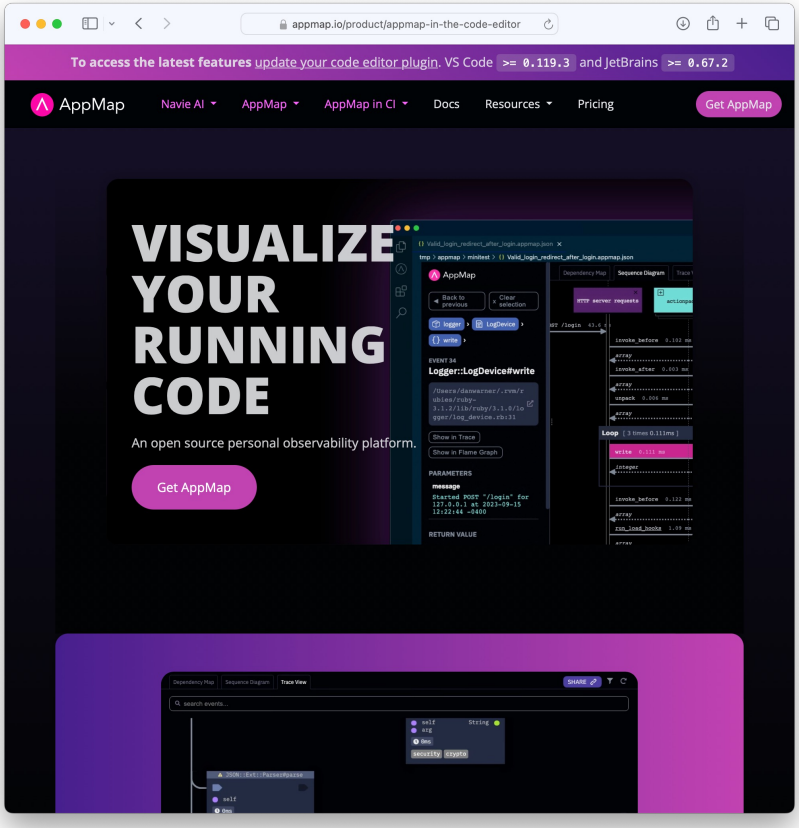
Static analysis of Python source code to generate package and class diagrams

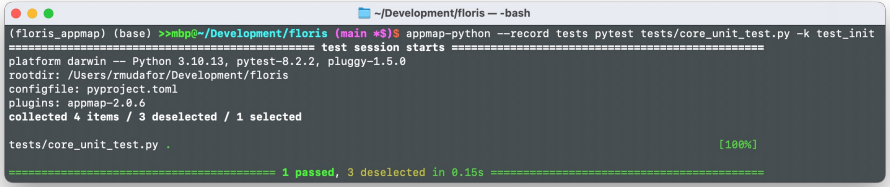


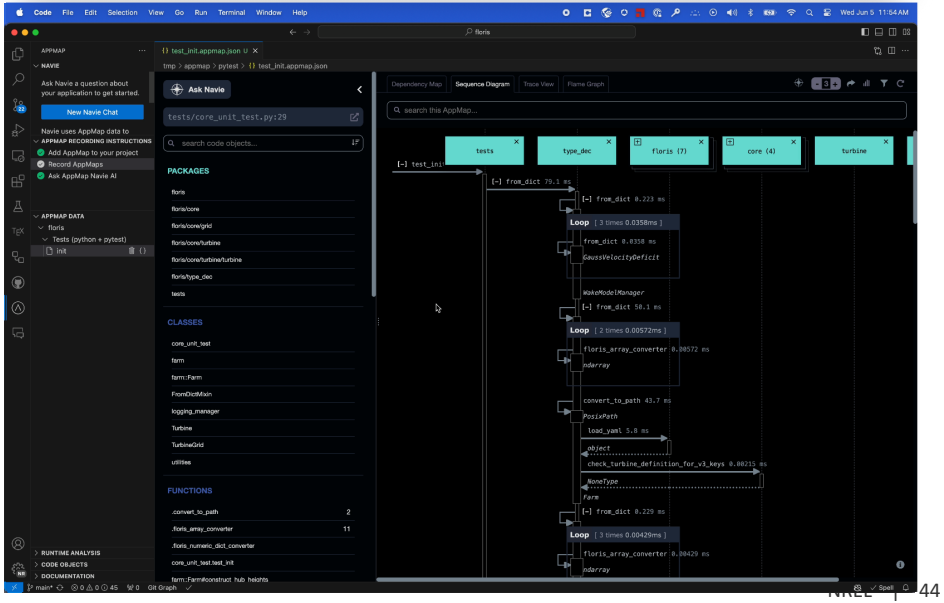


In Practice: AppMap

https://appmap.io/product/appmap-in-the-code-editor

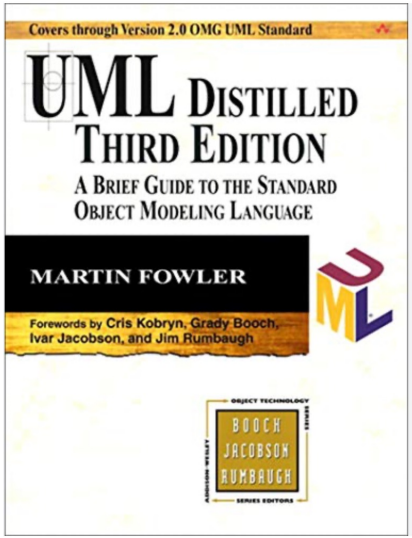




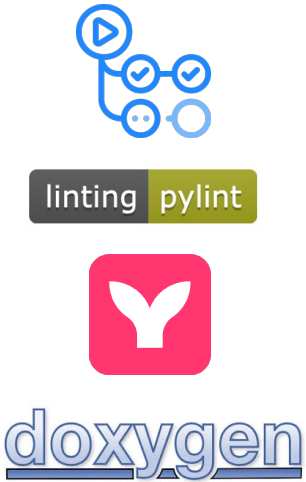


Summary

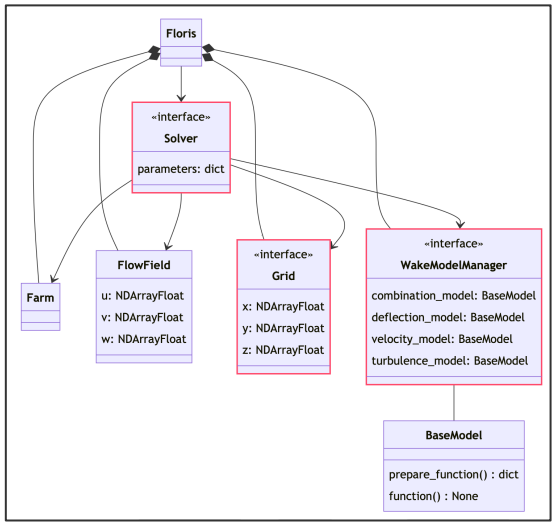
1. Read this book



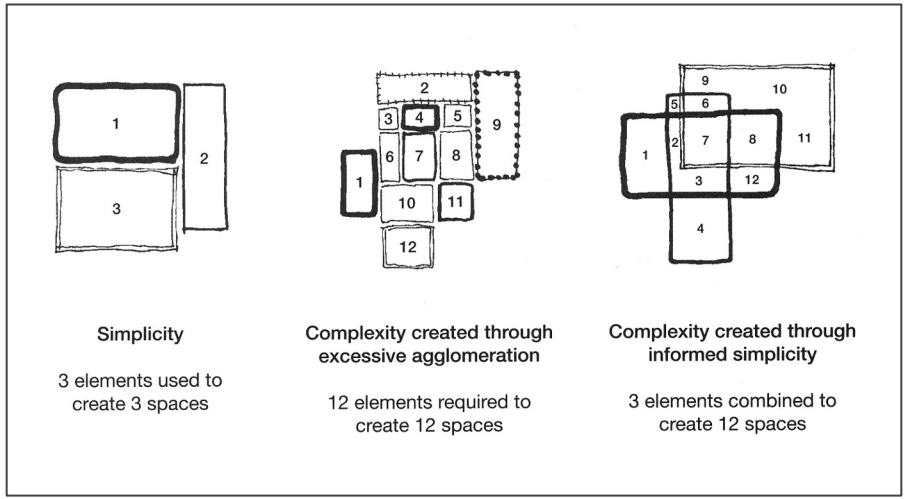
2. Use these tools



3. Directly communicate your software design



Stepping out to the big picture



101 Things I Learned in Architecture School

Thank you!

rafael.mudafort@nrel.gov
@rafmudaf
rafmudaf.github.io/communicating-design

NREL
Transforming ENERGY

Photo from iStock-627281636