

The S of SOLID

The Single Responsibility Principle

Maximilian Meffert

The Single Responsibility Principle

- Probably around since the 1970's
 - David L. Parnas, *On the Criteria To Be Used in Decomposing Systems into Modules*. Commun. ACM 15(12): 1053-1058 (1972)
- What does it say?
 - A module / class should only have one responsibility
 - A module / class should only have one reason to change (Robert C. Martin)
- What does it mean?
 - Separate things which are likely to change because of different reasons
 - Group things together which are likely to change for the same reason

The Single Responsibility Principle

- Why is it a “good” thing to adhere to?
 - Change of requirements is immanent through the life cycle of most software
 - Decreases change impact, i.e. number of modules to alter
 - Decreases risk of regression because of human error, i.e. developer faults
 - ... uhm, basically aims to protect software against faulting developers?
 - Increases a software designs capability for adaption to change
 - ... uhm, “agile”?

The Single Responsibility Principle

- What is a “reason to change”?
 - Changes to software are necessary because...
 - a) ... requirements have literally changed
 - b) ... of bugs, where requirements have not been met
 - Requirements originate from stake holders
 - Stake holders are who software must **respond** to
 - Hence: **Responsibility** Principle

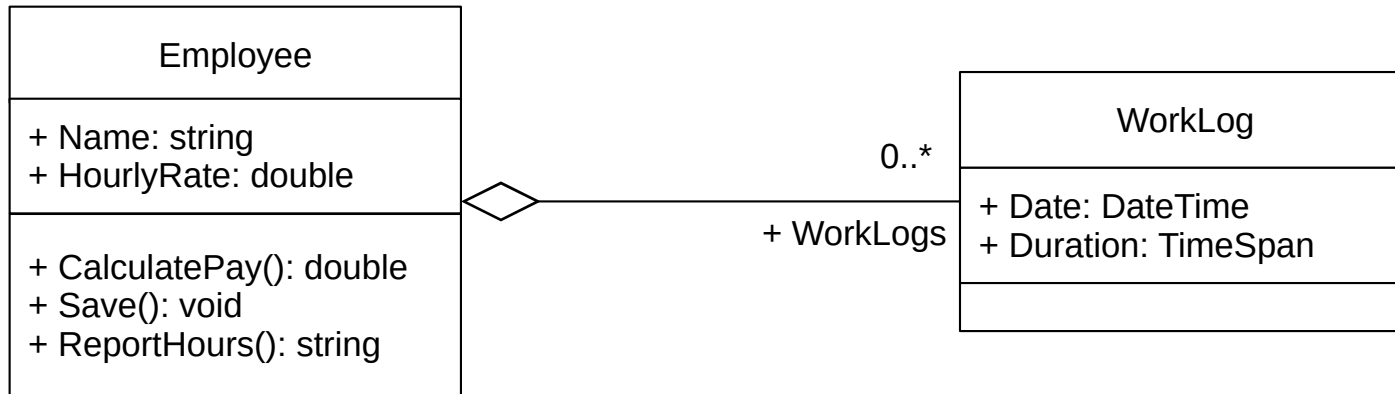
Example

The Acme Corp.

Employee Management System

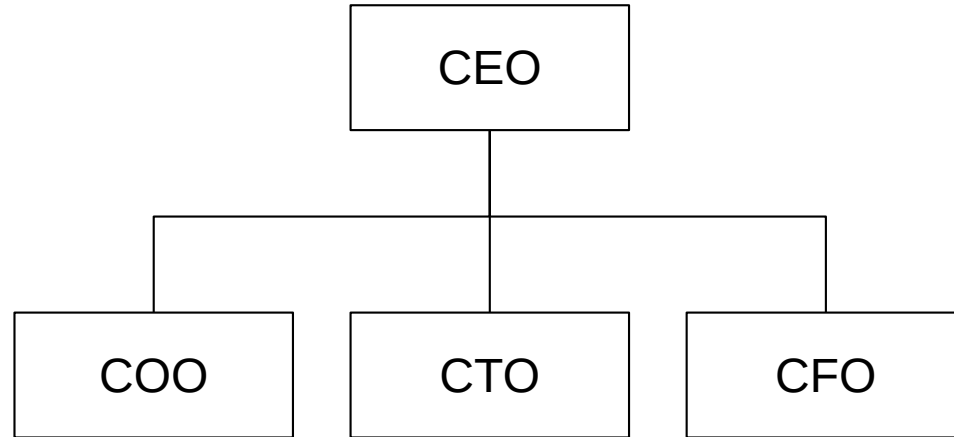
Example

Acme Corp. Employee Management System



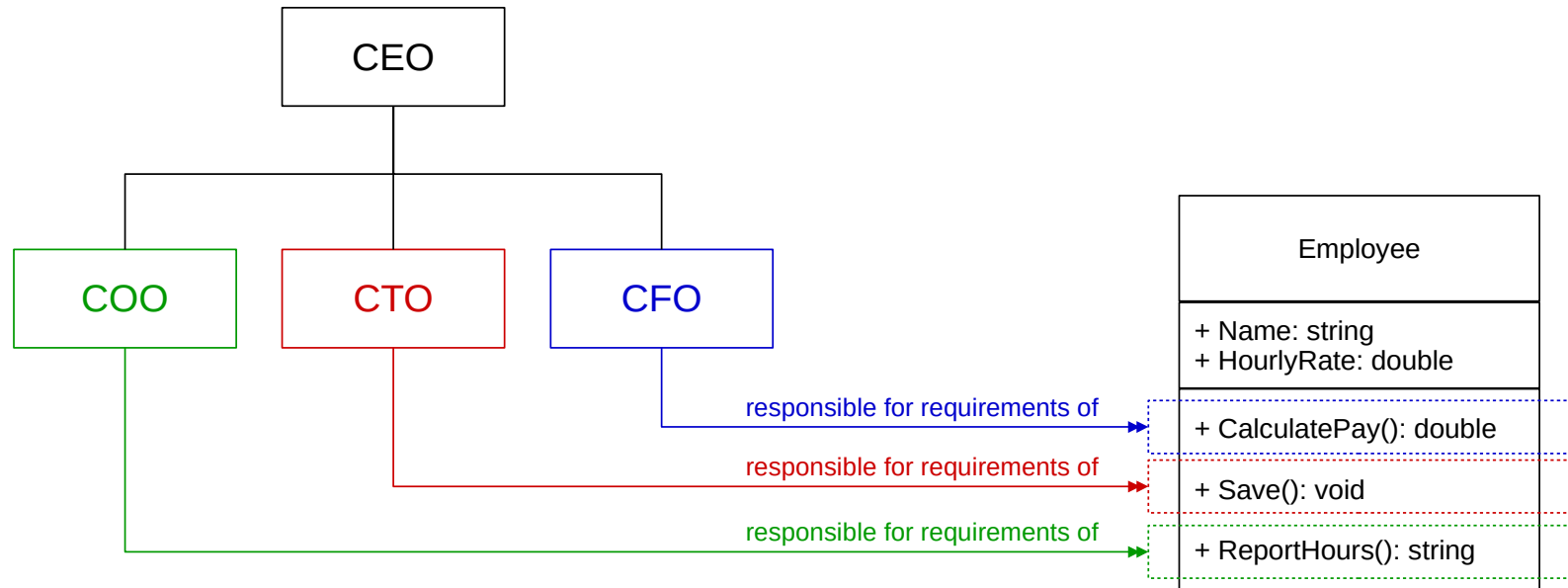
Example

Acme Corp. Organization Chart



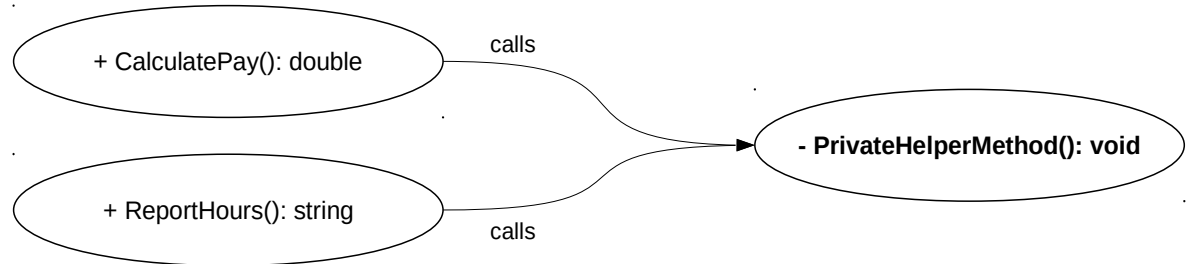
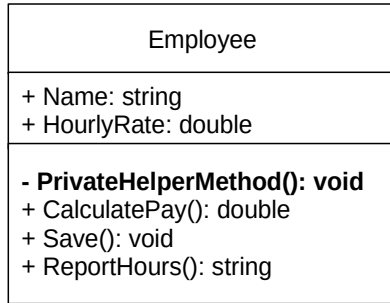
Example

Responsibilities / Reasons for Change



Example

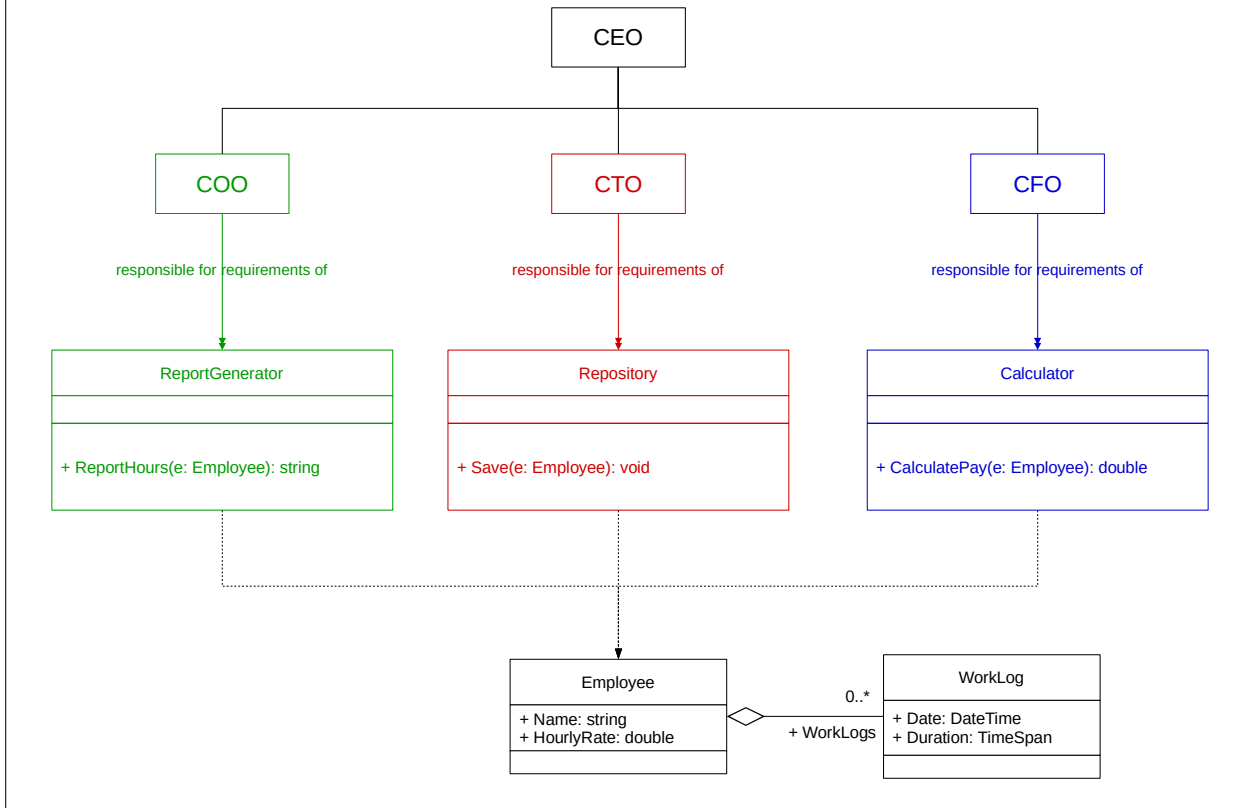
Method Coupling



Aggregation of responsibility may lead to hidden coupling which facilitates human error / developer faults

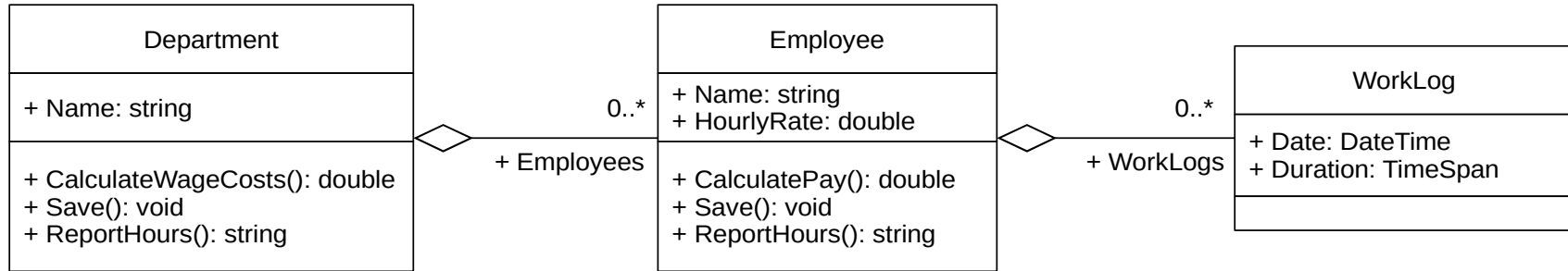
Example

Reorganized Design



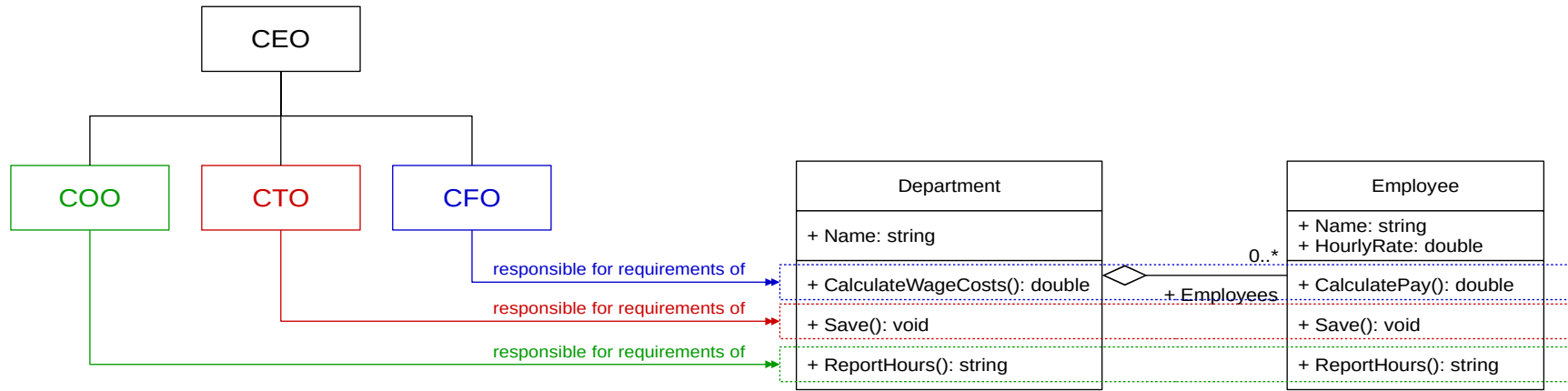
Example

Acme Corp. Employee Management System 2.0



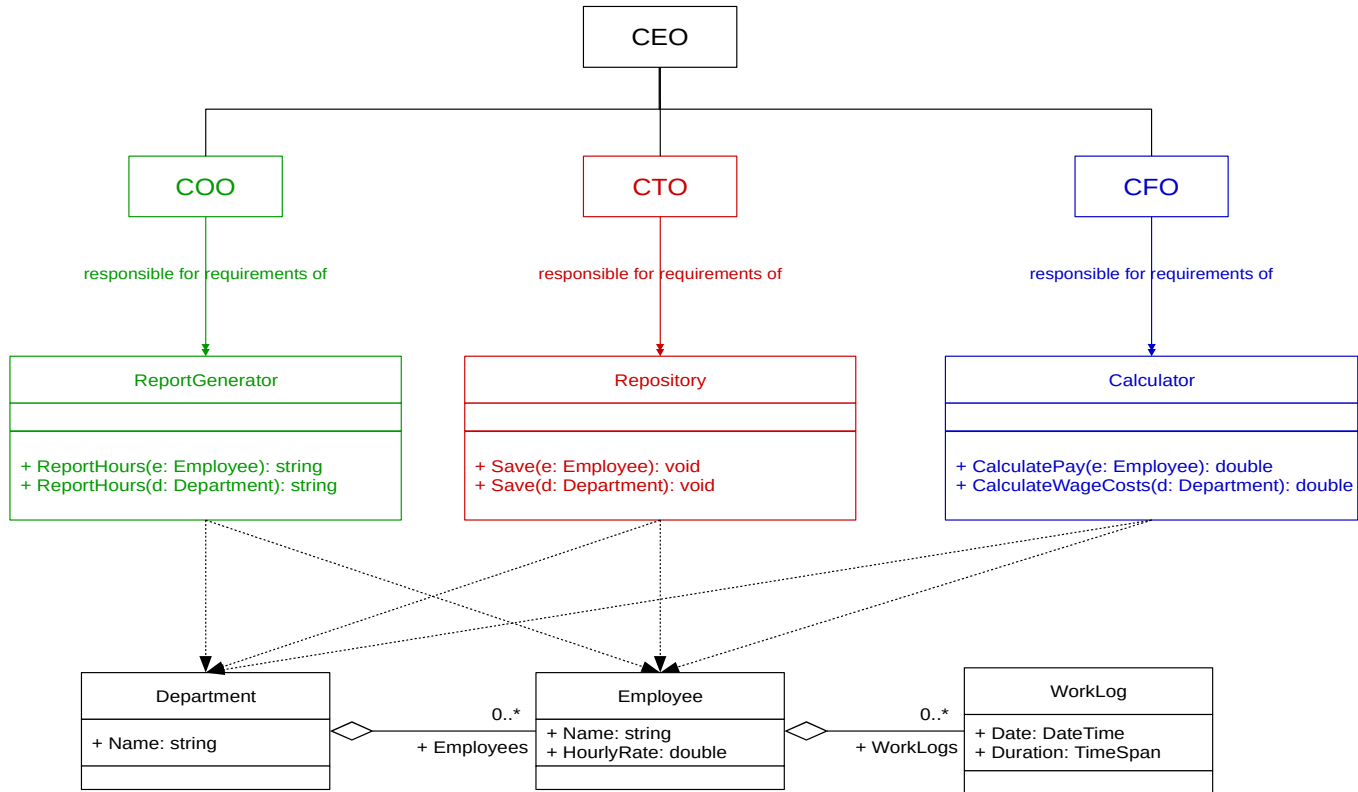
Example

Responsibilities / Reasons for Change



Example

Reorganized Design



Thanks!

References

- David L. Parnas, On the Criteria To Be Used in Decomposing Systems into Modules. Commun. ACM 15(12): 1053-1058 (1972)
- Robert C. Martin. 2003. Agile Software Development, Principles, Patterns, and Practices, Prentice Hall.
- Robert C. Martin. 2014. The Single Responsibility Principle. (May 2014). Retrieved April 22, 2019 from <http://blog.cleancoder.com/uncle-bob/2014/05/08/SingleReponsibilityPrinciple.html>
- James Ellis-Jones. 2017. Think you understand the Single Responsibility Principle? (October 2017). Retrieved April 22, 2019 from <https://hackernoon.com/you-dont-understand-the-single-responsibility-principle-abfdd005b137>