

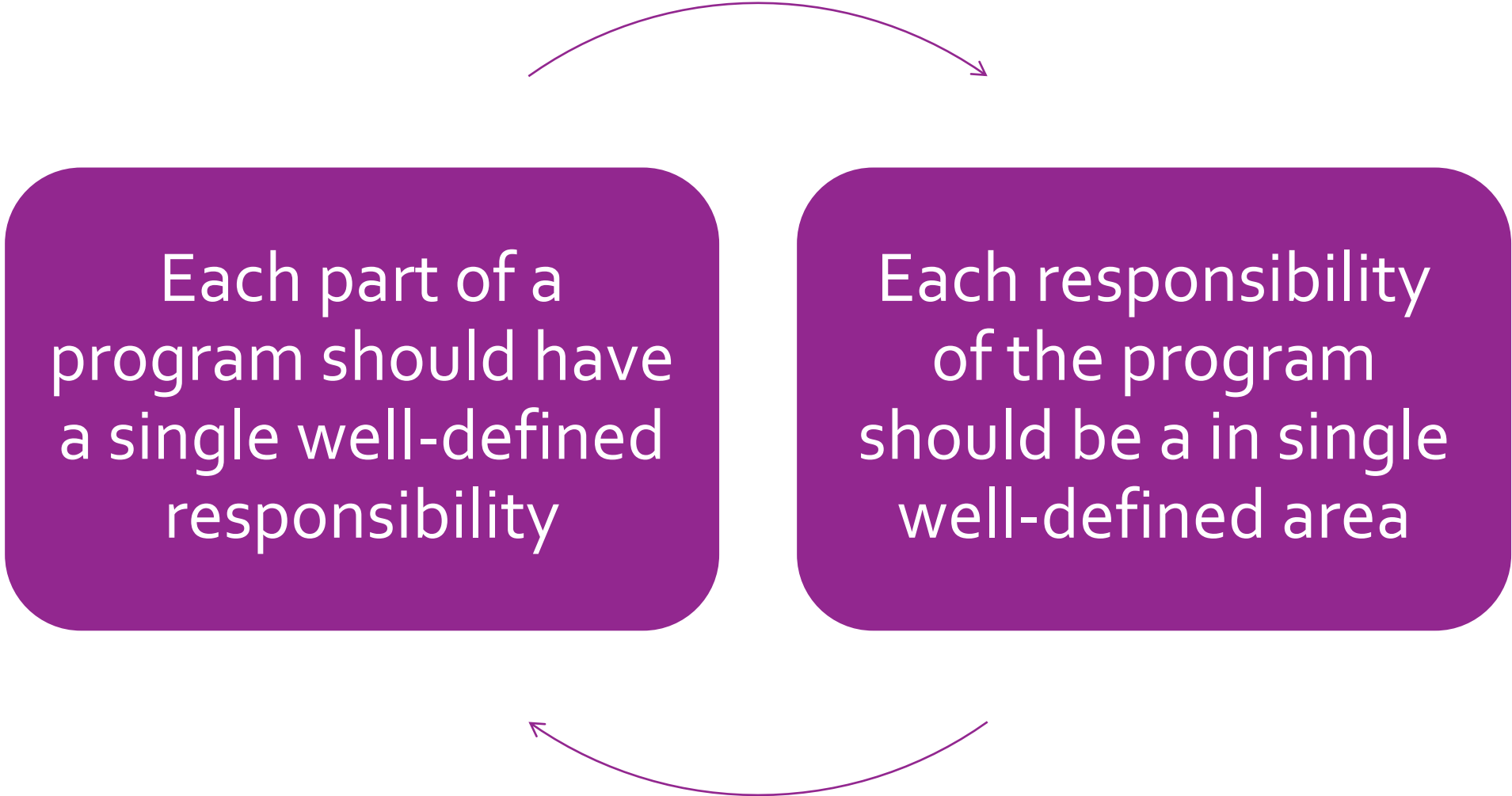


SOFTWARE ARCHITECTURE #1

How to build good quality software

Separation of Concerns

- There are many different architectural patterns
- One of the most important underlying principles is separation of concerns
- https://en.wikipedia.org/wiki/Separation_of_concerns



Each part of a
program should have
a single well-defined
responsibility

Each responsibility
of the program
should be a in single
well-defined area

One definition

Origin [\[edit \]](#)

The term *separation of concerns* was probably coined by [Edsger W. Dijkstra](#) in his 1974 paper "On the role of scientific thought".^[6]

Let me try to explain to you, what to my taste is characteristic for all intelligent thinking. It is, that one is willing to study in depth an aspect of one's subject matter in isolation for the sake of its own consistency, all the time knowing that one is occupying oneself only with one of the aspects. We know that a program must be correct and we can study it from that viewpoint only; we also know that it should be efficient and we can study its efficiency on another day, so to speak. In another mood we may ask ourselves whether, and if so: why, the program is desirable. But nothing is gained—on the contrary!—by tackling these various aspects simultaneously. It is what I sometimes have called "the separation of concerns", which, even if not perfectly possible, is yet the only available technique for effective ordering of one's thoughts, that I know of. This is what I mean by "focusing one's attention upon some aspect": it does not mean ignoring the other aspects, it is just doing justice to the fact that from this aspect's point of view, the other is irrelevant. It is being one- and multiple-track minded simultaneously.

How it pertains to GUI applications

The UI of an application should be interchangeable from the rest

A light purple arrow pointing downwards from the first box to the second box.

In fact, the best designed application can operate without UI

A light purple arrow pointing downwards from the second box to the third box.

If we plan for this up-front, we can implement more features easily.

Related to Two More Principles

Code coupling – quantity of dependencies between modules

Code cohesion – how focused code is on a single purpose

WHY DO YOU CARE?

Affects how easy it is
to write, modify,
understand, and
debug your code


Cross-Cutting Concerns

https://en.wikipedia.org/wiki/Cross-cutting_concern

Cross-cutting concerns are aspects of a program that affect several modules, without being easily encapsulated in any of them.



These concerns often cannot be cleanly decomposed from the rest of the system.



They can result in either code duplication, and or significant dependencies between systems

Examples of Cross Cutting Concerns

Logging

Error handling

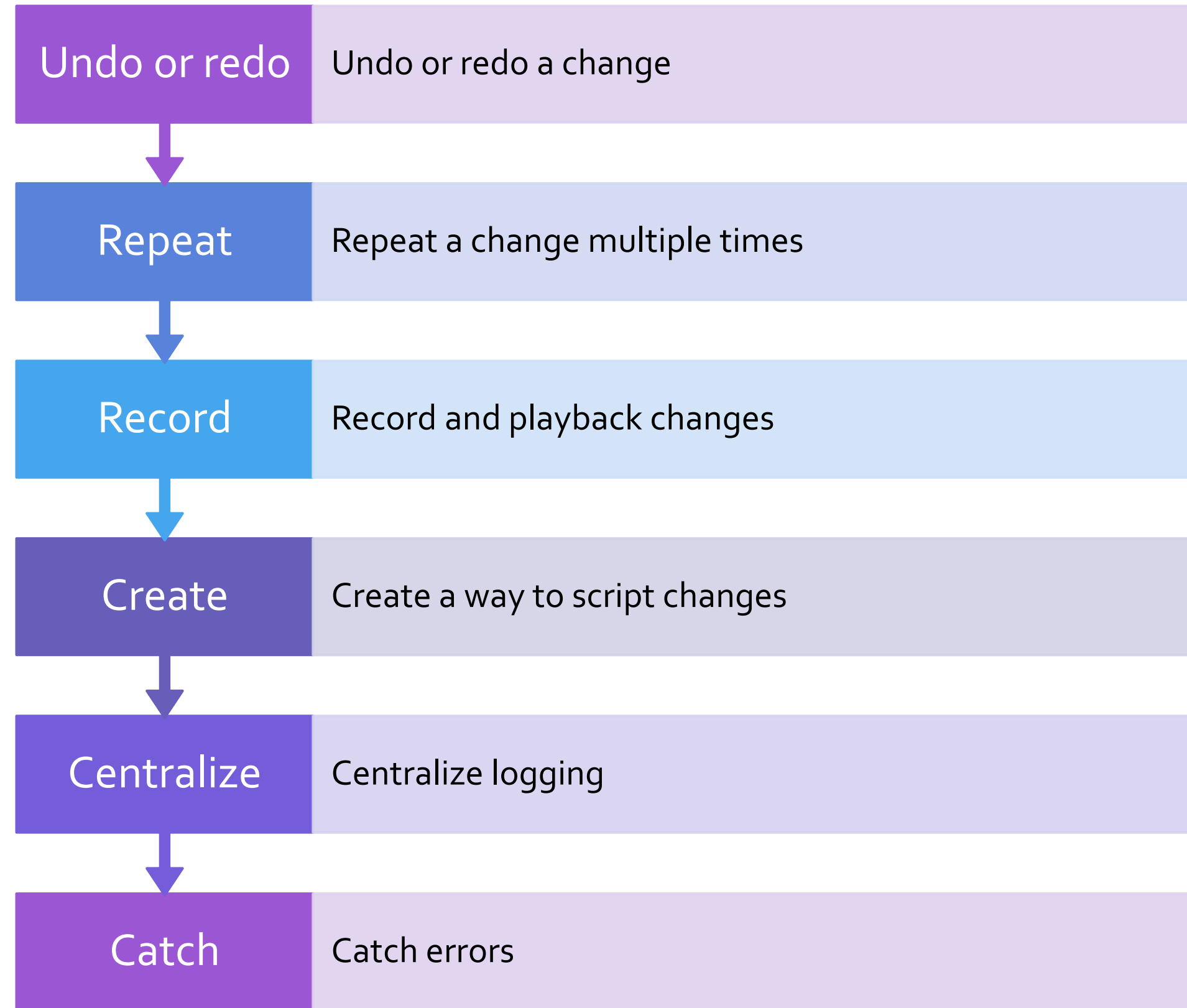
Macros

Help system

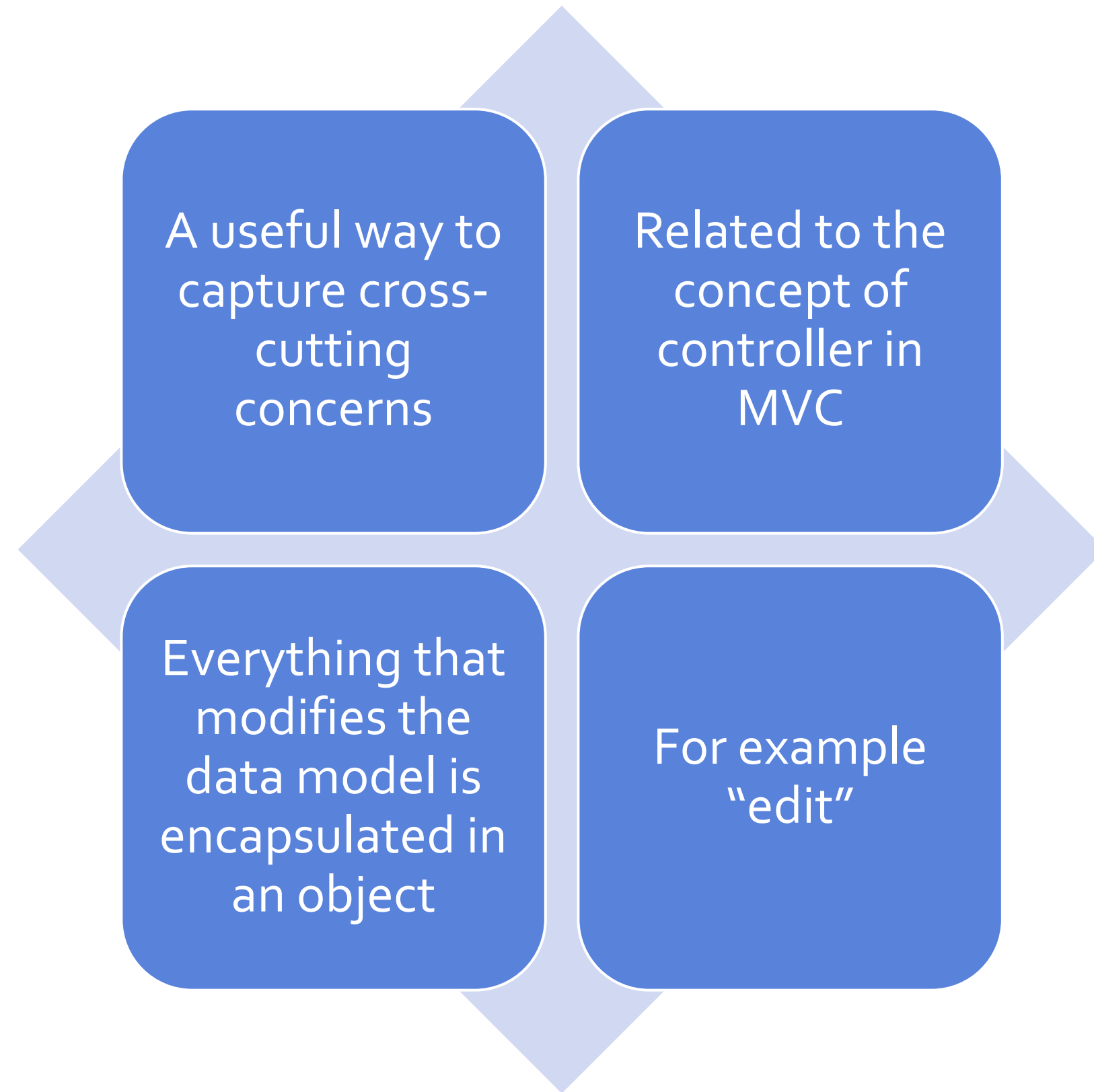
Undo/redo

Plug-in system

Scripting



Ask
yourself,
how would
you:



Command Pattern

Command Pattern according to Wikipedia

The **command pattern** is a [behavioral design pattern](#) in which an object is used to [encapsulate](#) all information needed to perform an action or trigger an event at a later time. This information includes the method name, the object that owns the method and values for the method parameters.