|  |
| --- |
| FAVV – AFSCA |
| favv.gif |
| **[Technical Fiche : MediatR]** |

Document Information

|  |  |
| --- | --- |
| Information | Description |
| Document owner | Emmanuel Nuyttens |
| Creation date | 14/02/2020 |
| Last update date |  |
| Document version | 1.0 |

Document history

|  |  |  |
| --- | --- | --- |
| Version | Date | Modification |
| 1.0 | 14/02/2020 | Initial Created |
|  |  |  |
|  |  |  |
|  |  |  |

Document validation

|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Role | Department | Date |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Document lecture references

|  |  |
| --- | --- |
| Title | Authors |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Contents

Aucune entrée de table des matières n'a été trouvée.

1. Introduction

In software development, the **mediator pattern** defines an **object that encapsulates how a set of related object interact**. This pattern **is considered to be a behavioral pattern** due to the way it can **alter the applications running behavior.**

With the mediator pattern**, communication between objects is encapsulated within a mediator object**. Objects no longer communicate directly with each others, but instead **communicate through the mediator**. This reduces the dependencies between communicating objects, thereby reducing tight-coupling.

1. Overview

What kind of problems can the Mediator pattern solve ?

1. Tight coupling between a set of interacting objects.
2. Change interaction between a set of objects independently.

Defining a set of interacting objects by accessing and updating each other directly is inflexible because it introduces tight-coupling to each other and makes it almost impossible to change the interaction independently from each others. This tight-coupling also make the concerned objects hard to be reusable and also makes them hard to be testable.

What kind of solution does the Mediator design pattern describe ?

* Define a separate (mediator) object that encapsulates the interaction between a set of objects.
* Objects delegate their interaction with other Objects to a mediator object instead of interacting with each other directly.

1. Definition

The essence of the Mediator Pattern is to “define an object that encapsulates how a set of objects interact”. It promotes loose-coupling by keep objects from referring to each other explicitly, and it allows their interactions to be varied independently. Client classes use the Mediator to send messages to other clients, and can receive messages from other Clients though an event on the Mediator class.