**Template Method**

Шаблонният метод (Template Method) позволява разделянето на имплементацията на алгоритъм на 2 нива. Логиката на алгоритъма е реализирана в базов родителски клас. Поведението на обектите в стъпките, специфични за конкретните наследници, е отложено за имплементация в самите тях. Използва се когато два или повече класа използват съвместим алгоритъм.

**Mediator Pattern**

Шаблонът за дизайн "посредник", дефинира обект, капсулиращ взаимодействията в набор от обекти. Той съдейства за намаляване на връзките между обектите и не позволява директно взаимодействие между тях. Също така дава възможност за свободна промяна на взаимоотношенията между обектите. Шаблонът се използва, когато множество обекти комуникират добре, но по сложен начин, когато многократното използване на обект е трудно, защото той има връзки с множество други обекти и когато едно поведение, разпределено между няколко класа, трябва да бъде представено без създаване на много подкласове. "Посредникът" спомага за откриване на проблем като обектите го уведомяват за промяна в състоянието си и той решава какво действие да се извърши, след което уведомява обектите как да го изпълнят.

**Memento Pattern**

Шаблонът "Мементо" запазване на състоянието на обекта, като на по-късен етап може да го възстанови. При него не се нарушава капсулацията на данните и се елиминира нуждата от копиране на целия обект с цел запазване на състоянието му. Също така обектът се подсигуряване обекта чрез пропъртита. В него им 2 типа обекти - създател и спомен.

**State Pattern**

Шаблонът "състояние" отделя различните състояния на обекта в собствен клас. Клиентът не разбира за различните състояния, защото обектът, с който работи не се променя – променя се неговото състояние и поведение. Този шаблон помага да избегнем чести, дълги и повтарящи се проверки за извикването на правилен метод, като капсулира специфичното поведение на всяко състояние в нов клас.