No projeto desenvolvido, diversos conceitos de programação orientada a objetos foram aplicados de forma prática e integrada, o encapsulamento foi utilizado ao proteger os atributos das classes, como `MediaItem`, `Movie` e `Series`, Isso garante que os dados internos não sejam manipulados diretamente, mantendo a integridade das informações.

A herança aparece na relação entre `Movie` e `MediaItem`, assim como `Series` e `MediaItem`. Essas subclasses herdam atributos e métodos da classe abstrata `MediaItem`, evitando repetição de código e permitindo especializações, como o atributo `director` em `Movie` ou `seasons` em `Series`.

O polimorfismo se manifesta principalmente no método `play()`, que é definido na classe abstrata `MediaItem` e implementado de formas diferentes em `Movie`, `Series` e `Episode`. Isso permite que, independentemente do tipo de mídia, possamos chamar `play()` em qualquer objeto `MediaItem` e o comportamento será específico para cada tipo.

As interfaces também foram utilizadas através de `Streamable`, que define o método `stream()`. Classes como `Movie` e `Episode` implementam essa interface, garantindo que qualquer mídia que possa ser transmitida siga a mesma assinatura de método, possibilitando tratar diferentes tipos de mídia de forma uniforme.

Por fim, a agregação é visível nas relações entre `Series` e `Episode`, e entre `Playlist` e `MediaItem`, um objeto `Series` contém múltiplos `Episode`, mas cada episódio também mantém referência à sua série, de forma similar, `Playlist` agrega vários objetos `MediaItem`, essas relações mostram que os objetos podem existir de forma independente, mas estão logicamente conectados para formar uma estrutura mais complexa.