Diferencias entre estándares onion, hexagonal y clean

A grandes rasgos se trata de una arquitectura multicapa construida en torno a un modelo de dominio independiente de todo lo demás. Las dependencias van hacia el centro, por lo que todo depende de ese modelo de dominio. A su alrededor se organizan varias capas, estando en las más cercanas las interfaces de repositorio, es decir, las que definen el comportamiento del almacenamiento de los datos pero no lo implementan. En las capas siguientes está la lógica de negocio que usa estas interfaces y que en tiempo de ejecución tendrá las implementaciones apropiadas. Alrededor del núcleo de modelo puede haber un número variable de capas, pero siempre debe cumplirse que las interfaces estén más cerca que las clases que las utilizan. Con esto ya tenemos creado el core lógico de nuestra aplicación, que no tiene absolutamente ningún detalle de infraestructura.

Por último, en la capa más exterior es donde estarán todos los detalles de comunicación con el exterior (tanto de interfaz con el usuario como el almacenamiento) y los tests de integración.

La arquitectura hexagonal es un estilo de arquitectura de software que mueve el foco de un programador desde un plano más conceptual hacia la distinción entre el interior y el exterior del software. La parte interior son los casos prácticos y el modelo domain está construido sobre ello. La parte exterior es UI, base de datos, mensajería, etc. La conexión entre el interior y el exterior es mediante puertos, y su implementación equivalente se conocen como adaptadores. Por esta razón, este estilo de arquitectura hexagonal se conoce habitualmente como Puertos y Adaptadores. La arquitectura hexagonal promueve la separación de asuntos mediante la encapsulación de la lógica en diferentes capas de la aplicación.

Clean Architecture es un nombre popularizado por [Robert Cecil Martin](https://en.wikipedia.org/wiki/Robert_C._Martin), conocido como “Uncle Bob” que se basa en la premisa de estructurar el código en capas contiguas, es decir, que solo tienen comunicación con las capas que están inmediatamente a sus lados. Los niveles de los que se compone Clean Architecture son los siguientes:

UI: la interfaz de usuario propiamente dicha (html, xml, etc).

Presenters: clases que se subscriben a los eventos generados por la interfaz y que responden en consecuencia, también realizan el pintado de la información en la interfaz.

Use Cases: evaluación de reglas de negocio y tomas de decisiones.

Entities: modelo de datos de la aplicación, comunicación con servicios web, cache de datos.

Clean Code, o Código Limpio, es una filosofía de desarrollo de software que consiste en aplicar técnicas simples que facilitan la escritura y lectura de un código, volviéndolo más fácil de entender.

https://www.adictosaltrabajo.com/2019/07/02/capas-cebollas-y-colmenas-arquitecturas-en-el-backend/

https://apiumhub.com/es/tech-blog-barcelona/arquitectura-hexagonal/

<https://www2.deloitte.com/es/es/pages/technology/articles/clean-architecture.html>

<https://www.hostgator.mx/blog/clean-code-codigo-limpio/>