

# **CleanArchitecture**

---

카카오톡 #탭 적용기

#탭 개발자 Latte

# **Developer**

# **Application**

# **Android Application**

# **iOS Application**

# **Server Application**

# **[Platform] Application**

# **Todo Application**

# **Calendar Application**

# **UseCase1 Application**

# What is the focus?

---

**Platform**

**UseCase**

# **To do Application**

**Android?**

To do Application

**iOS?**

To do Application

**To do** Application

**UseCase1** Application

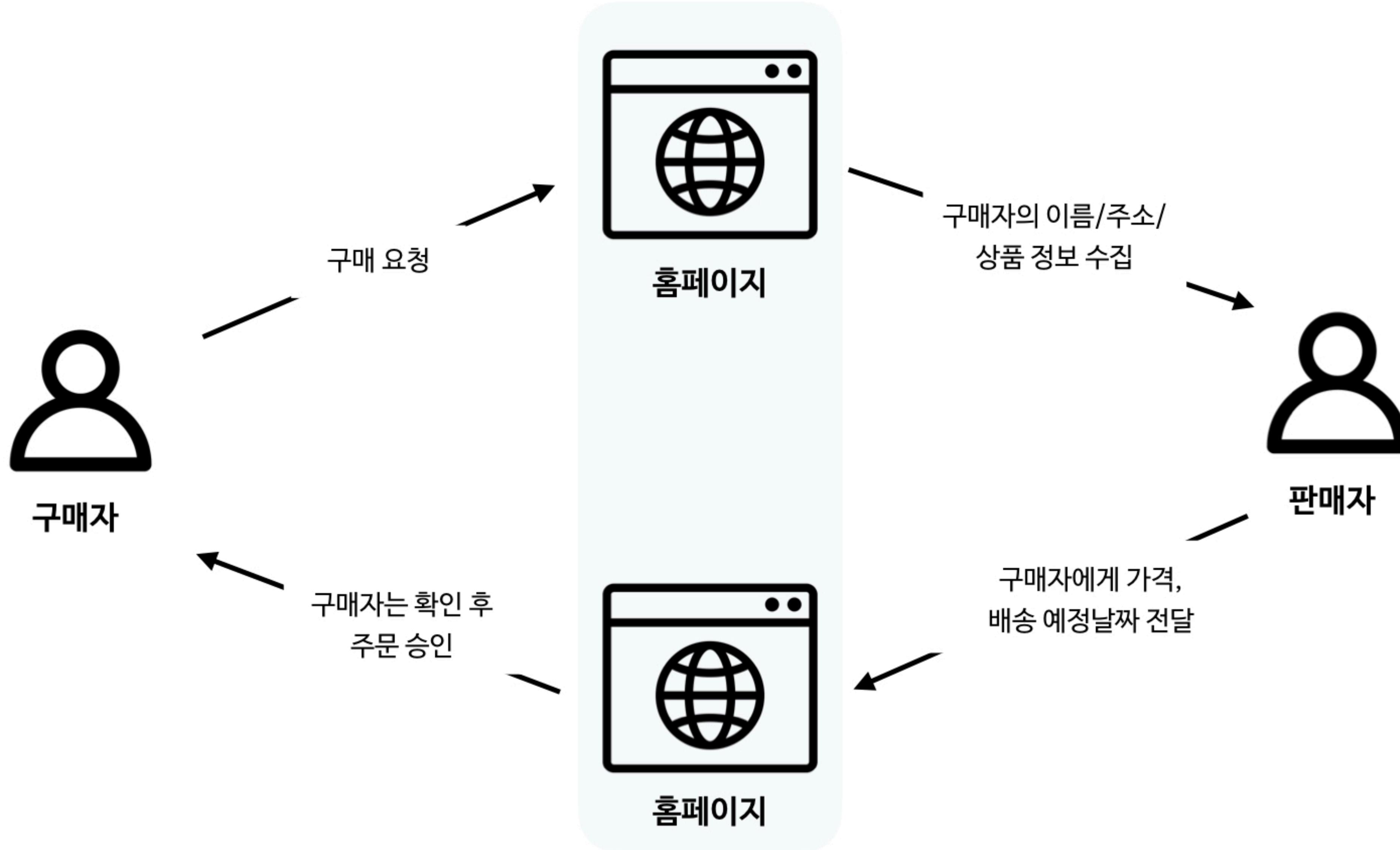
**[ UseCase ]**

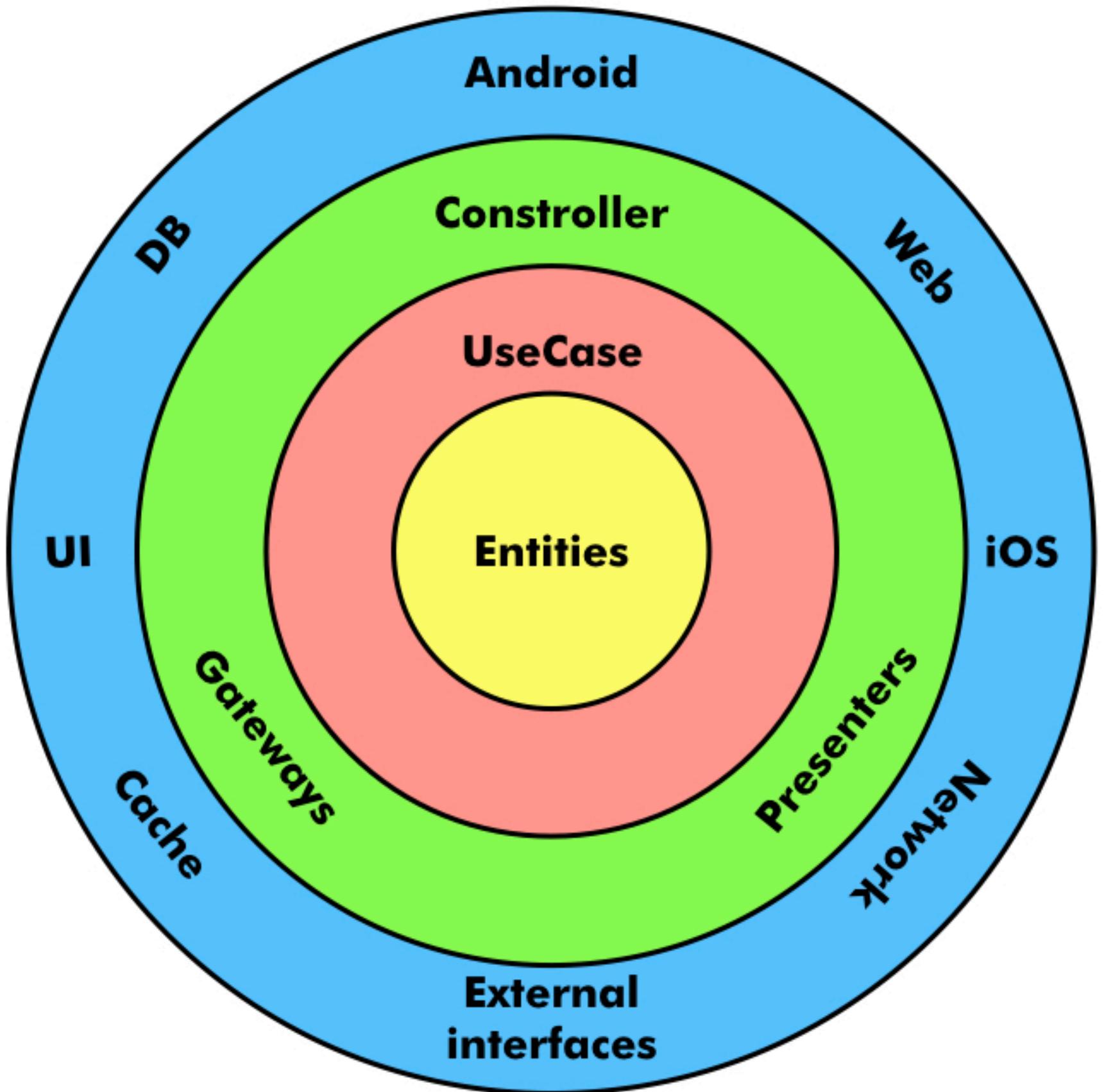
Is my **UseCase1**  
Bound to Platform?

**Architecture**

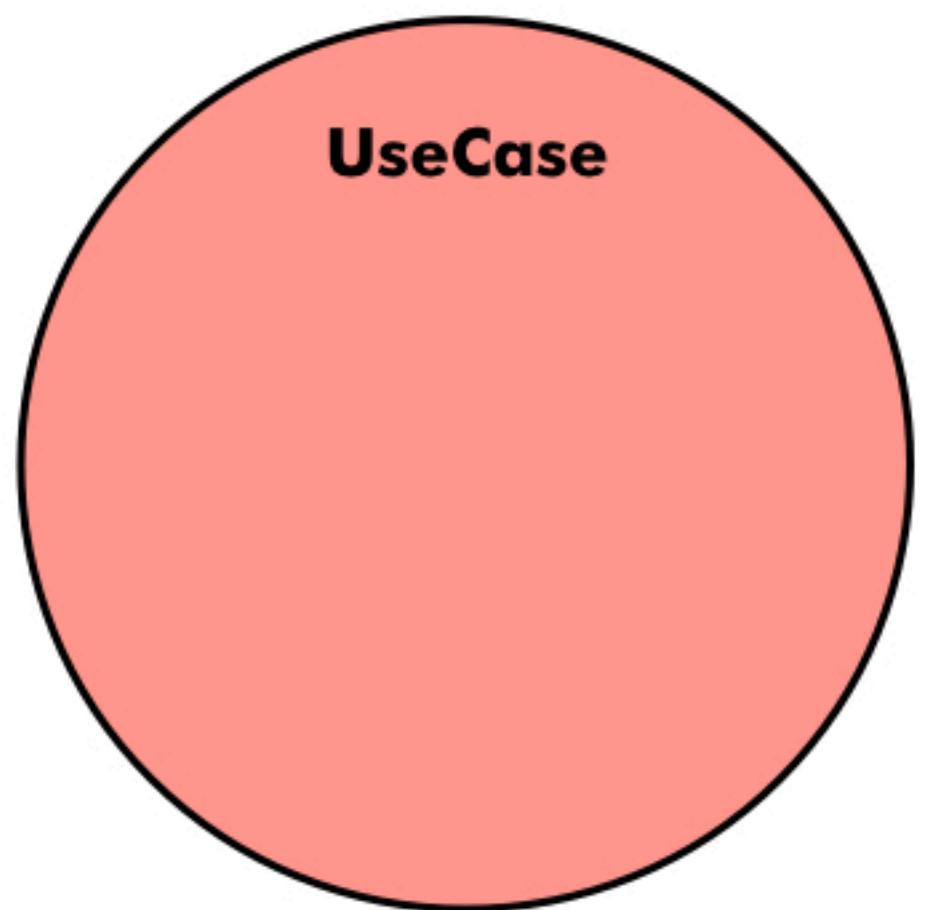
**"Architecture is about intent"**

- Uncle Bob



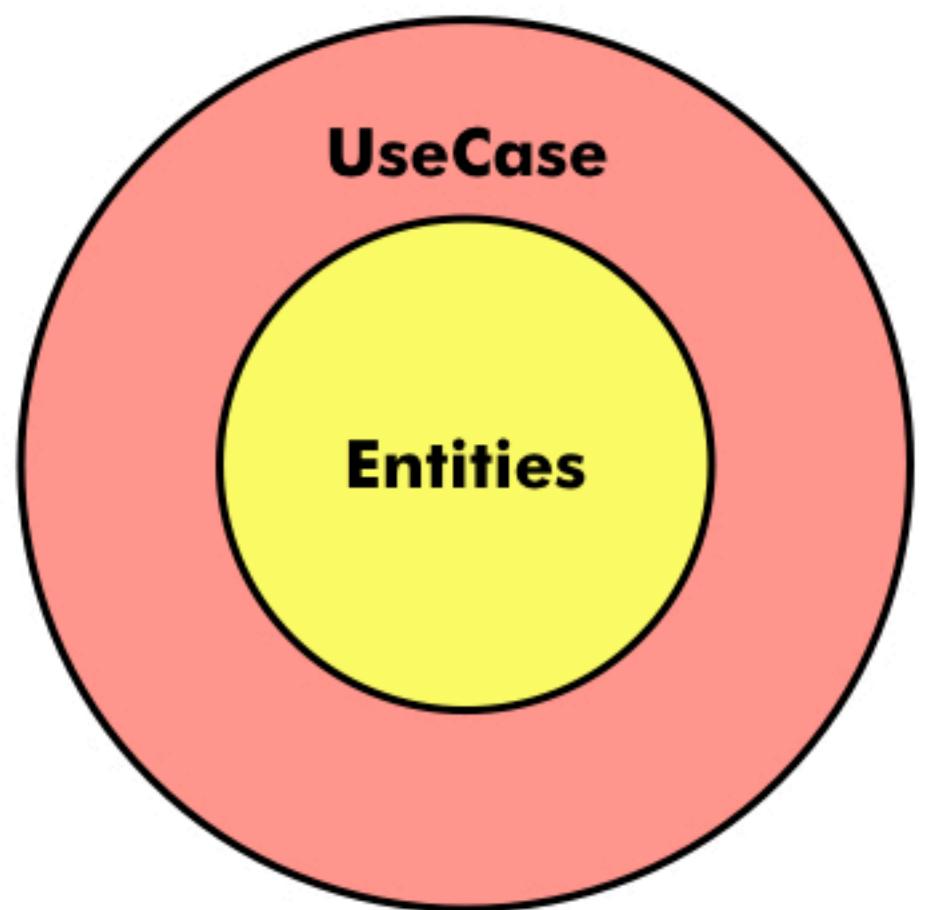


# The Clean Architecture

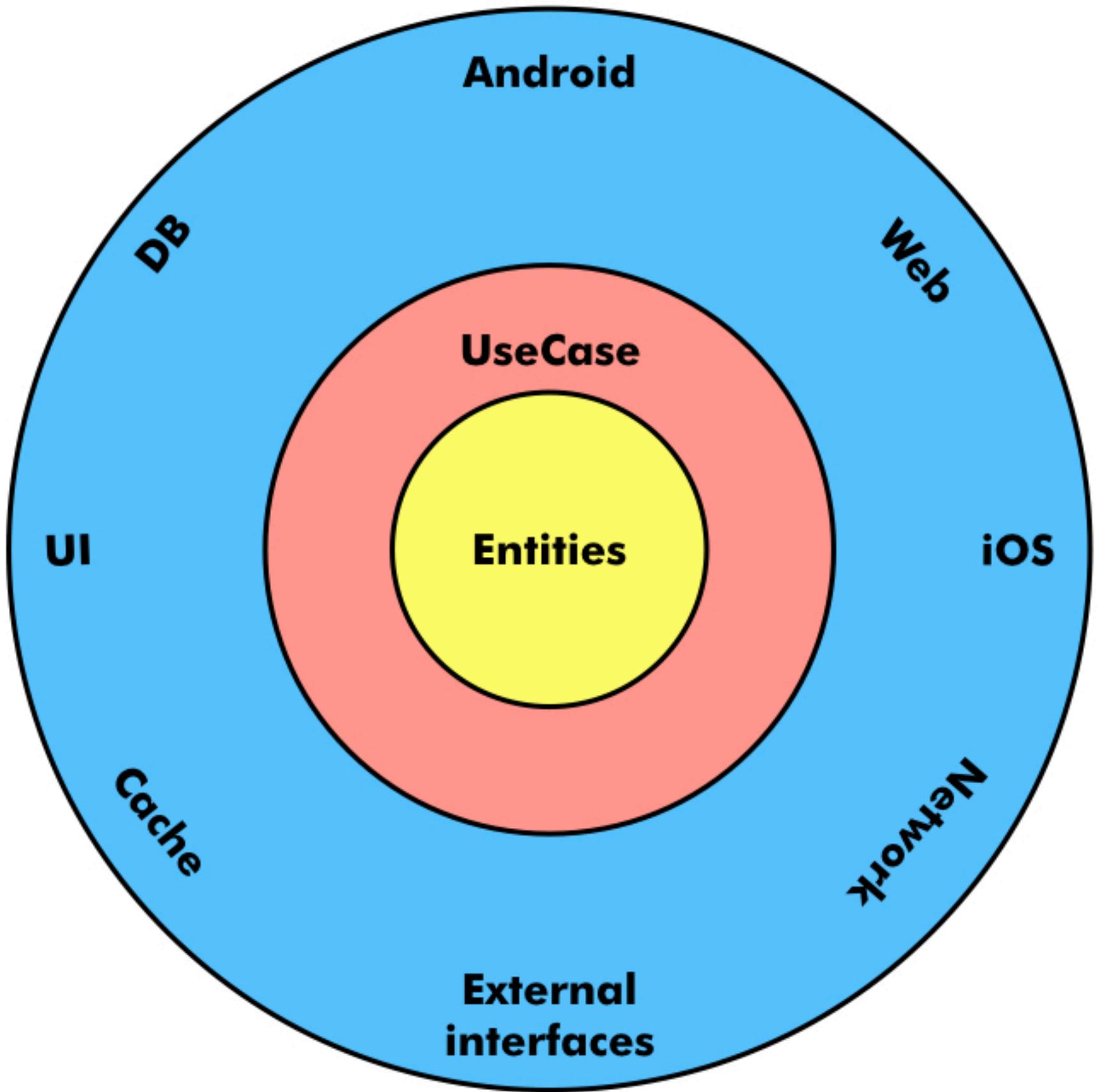


**UseCase**

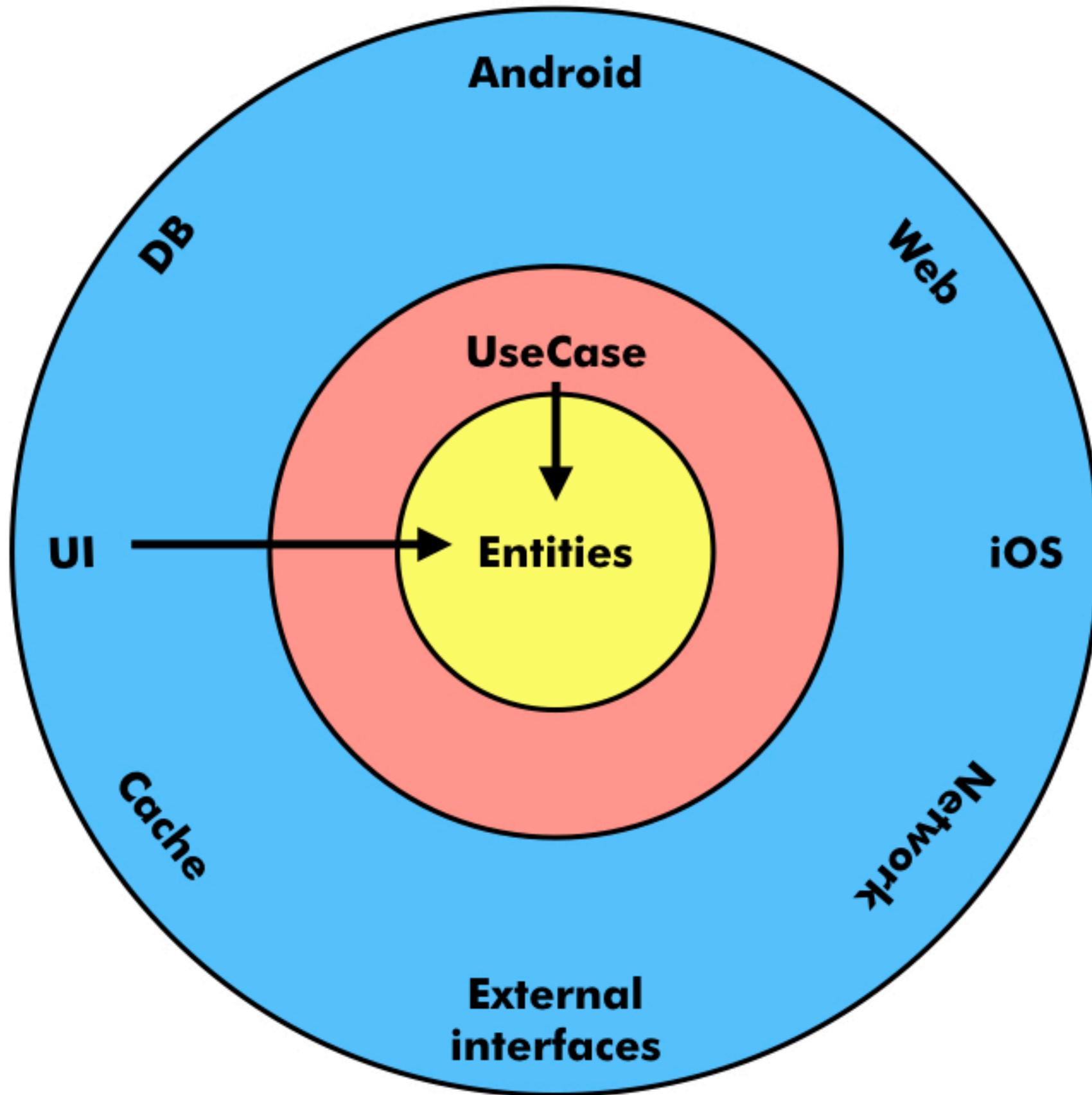
**Use Case**



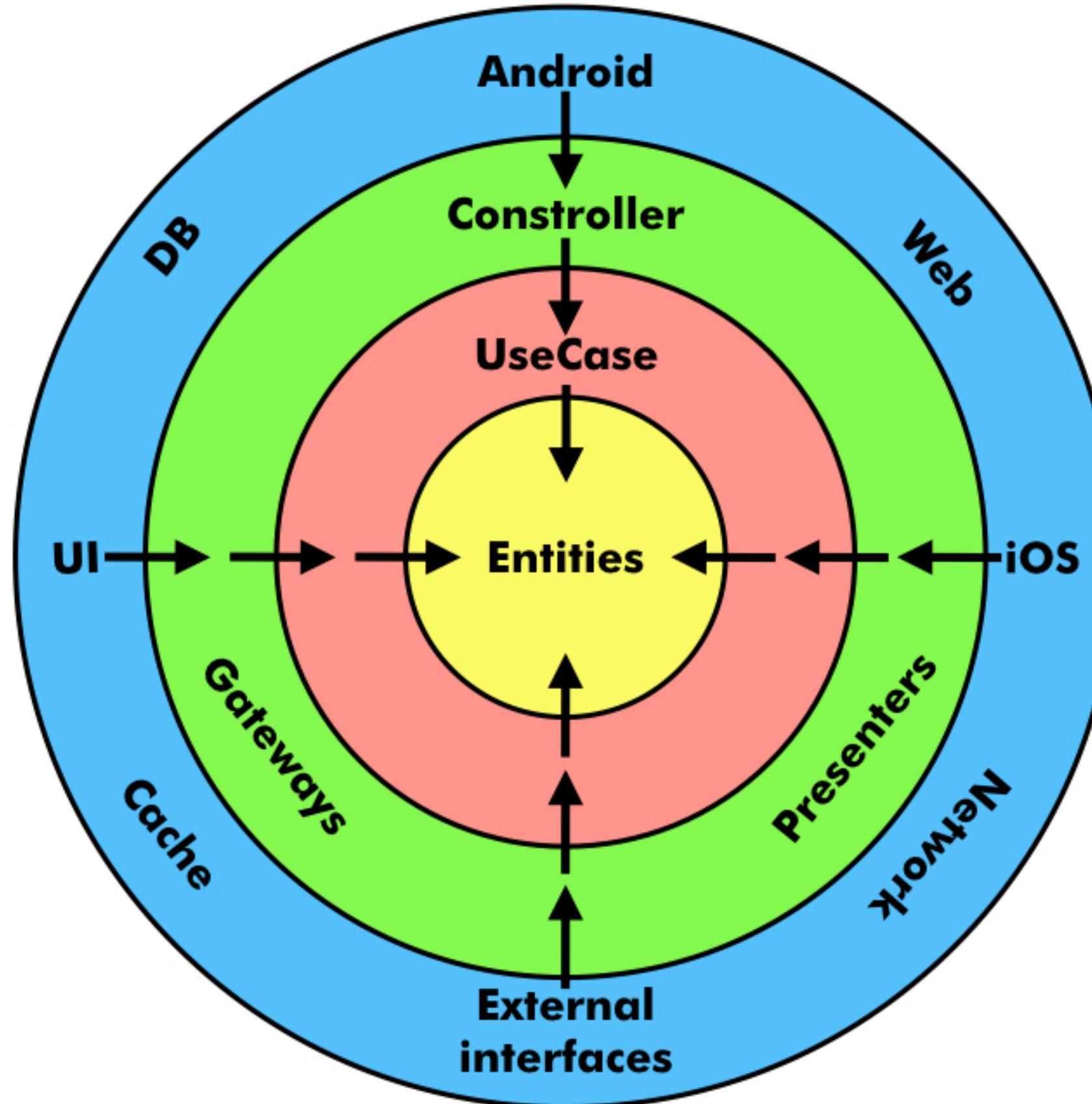
**Entity**  
Business Object



**Entity**  
{DB, API, ... }



**Entity**  
{DB, API, ... }



# The Dependency Rule

**Done.**

**So what?**

**Next!**

# Why?

#탭의 구조 및 문제점

MVVM

MVVM

**VM**

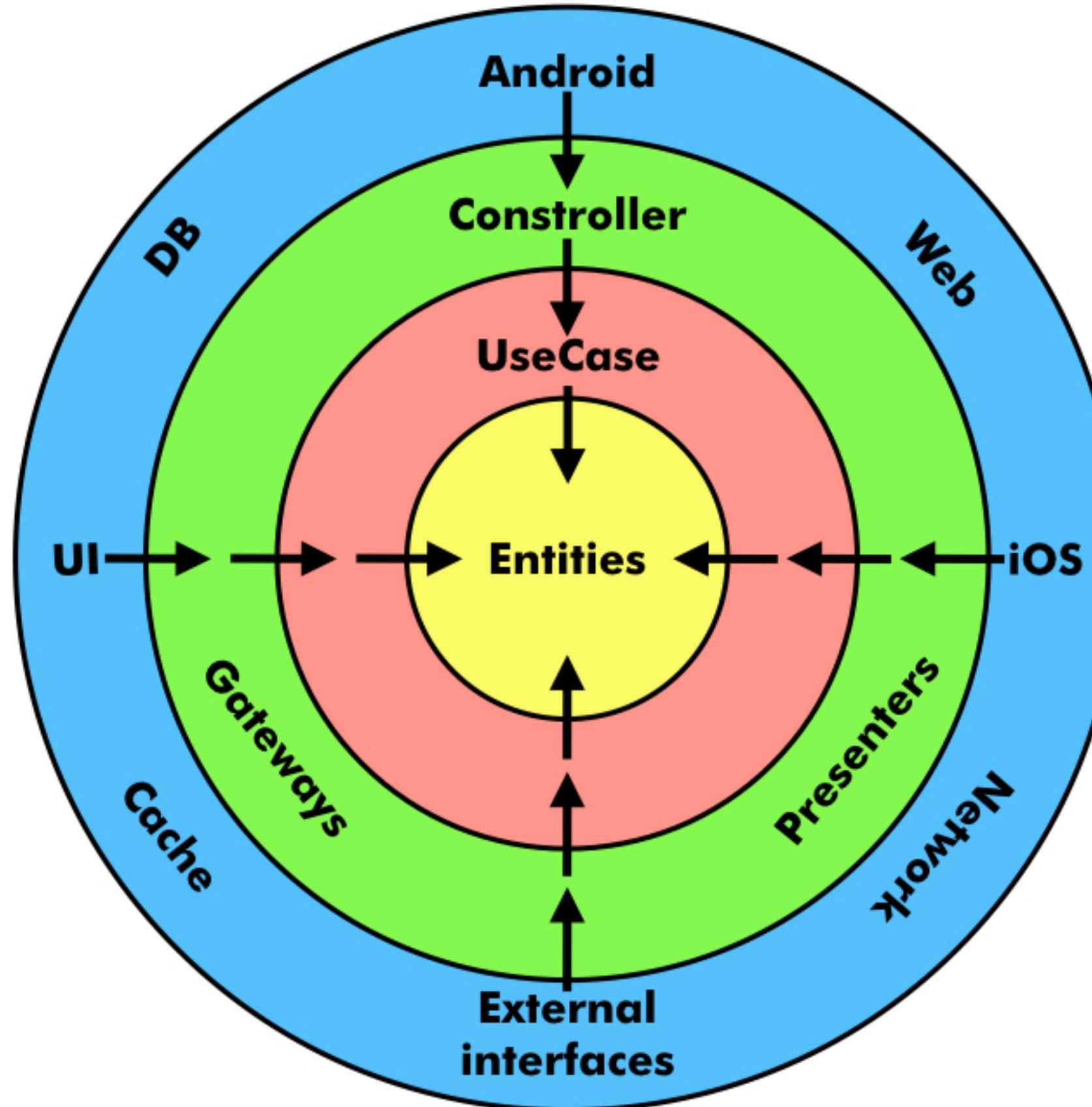
Presenter / Domain / Data

Presenter / Domain / Data

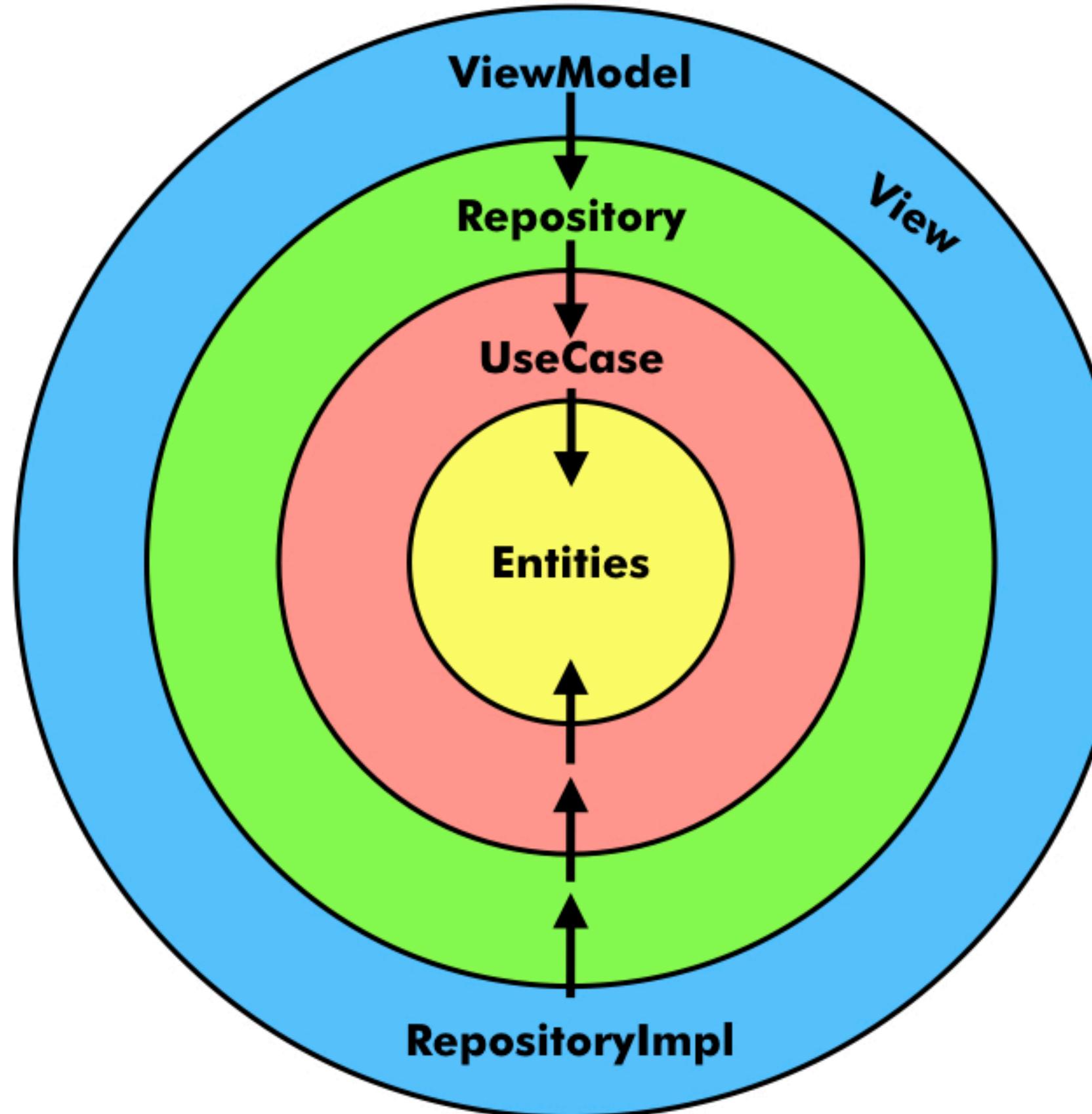
**View + ViewModel**

Presenter / Domain / Data  
**Repository + UseCase**

Presenter / Domain / Data  
**RepositoryImpl**  
+ **DataSource**



# The Dependency Rule



# The Dependency Rule

**Show me the code!**

~~Cheat Enabled~~

# 카카오톡

## #탭 코드

The screenshot shows the Android Studio interface with the code editor open to the `View.java` file. The code is part of the `View` class and handles mapping view-relative coordinates to screen-relative coordinates. It includes logic for handling scroll offsets and parent view matrices.

```
    7728     outRect.set(position.left, position.top, position.right, position.bottom);
    7729     outRect.set(Math.round(position.left), Math.round(position.top),
    7730         Math.round(position.right), Math.round(position.bottom));
    7731 }
    7732 /**
    7733 * Map a rectangle from view-relative coordinates to screen-relative coordinates
    7734 *
    7735 * @param rect The rectangle to be mapped
    7736 * @param clipToParent Whether to clip child bounds to the parent ones.
    7737 * @hide
    7738 */
    7739 public void mapRectFromViewToScreenCoords(RectF rect, boolean clipToParent) {
    7740     if (!hasIdentityMatrix()) {
    7741         getMatrix().mapRect(rect);
    7742     }
    7743
    7744     rect.offset(mLeft, mTop);
    7745
    7746     ViewParent parent = mParent;
    7747     while (parent instanceof View) {
    7748         View parentView = (View) parent;
    7749
    7750         rect.offset(-parentView.mScrollX, -parentView.mScrollY);
    7751
    7752         if (clipToParent) {
    7753             rect.left = Math.max(rect.left, 0);
    7754             rect.top = Math.max(rect.top, 0);
    7755             rect.right = Math.min(rect.right, parentView.getWidth());
    7756             rect.bottom = Math.min(rect.bottom, parentView.getHeight());
    7757         }
    7758
    7759         if (!parentView.hasIdentityMatrix()) {
    7760             parentView.getMatrix().mapRect(rect);
    7761         }
    7762
    7763         rect.offset(parentView.mLeft, parentView.mTop);
    7764
    7765         parent = parentView.mParent;
    7766     }
    7767 }
```

# 기자오톡

## #탭 코드

### 예제 코드

기자오톡  
#탭 코드  
예제 코드

방송 #영화 쇼핑 뉴스 웹소설 ⌛ ↗

지금 딱 추천



얼굴 밀으로 입으로 히어로 연기를 입으로만 했다  
하는 배우 디즈니+ 애심작, '팔콘과 원  
터솔져'를 둘러싼 궁금증 3  
테일러콘텐츠 맥스무비



헉! 카라 출신 강지영의 충격적인 일본 근황  
필더무비 살인의 추억, 설국열차, 마더 등 꼭 봐야할 봉준호 전...  
얼루어코리아



더보기 >

실시간 예매순위



# 검색어를 입력해주세요



#김군 감상평 #봉준호 훈장검토

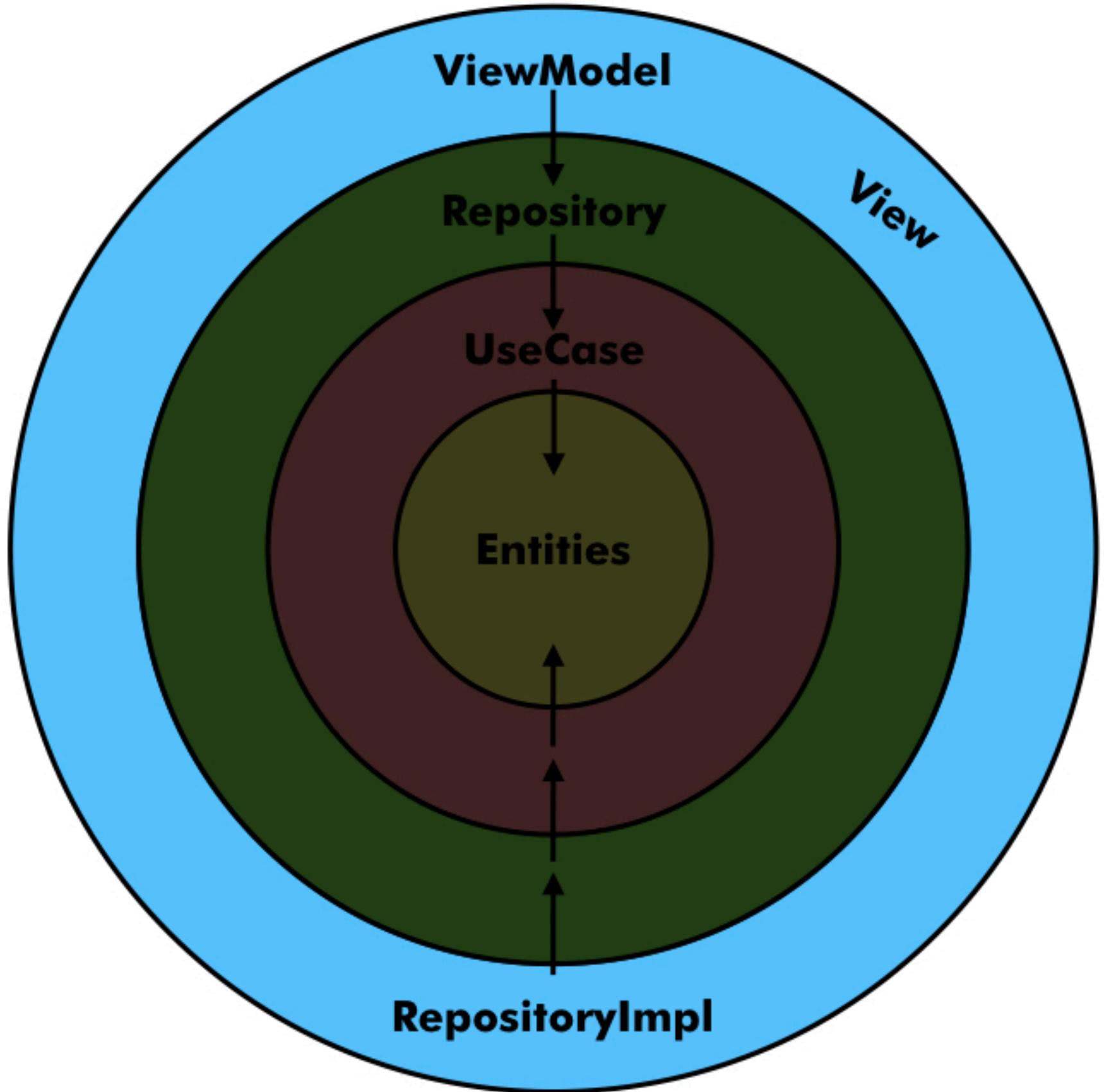
#맨인블랙 비하인드 #기생충 미국개봉



...

# 예제 코드

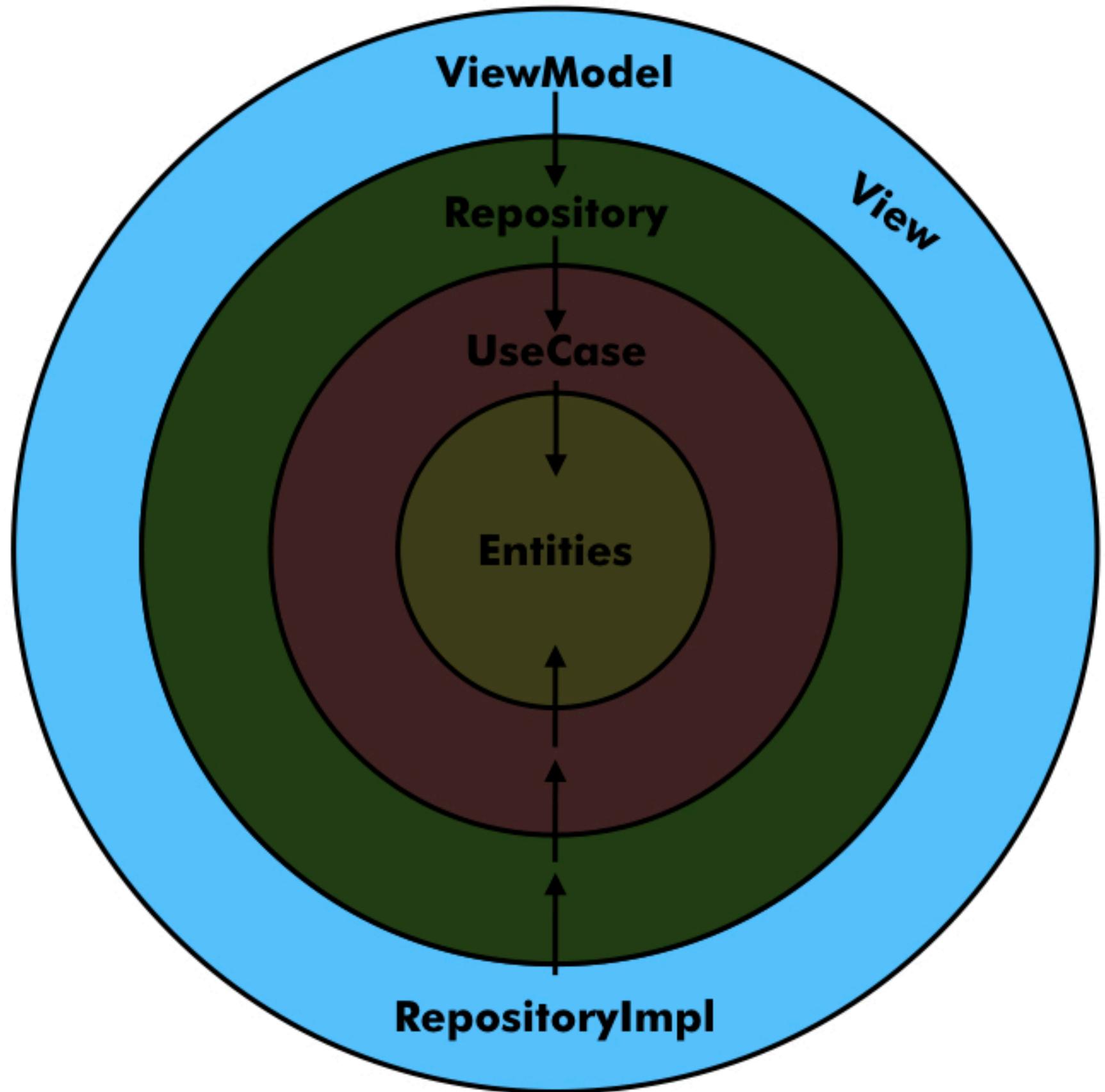
With Uncle Bob



```
swipeRefreshLayout = view.findViewById<SwipeRefreshLayout>(R.id.swipe_refresh_layout);
setOnRefreshListener {
    viewModel?.onSwipeRefresh()
}
}
```

# 예제 코드

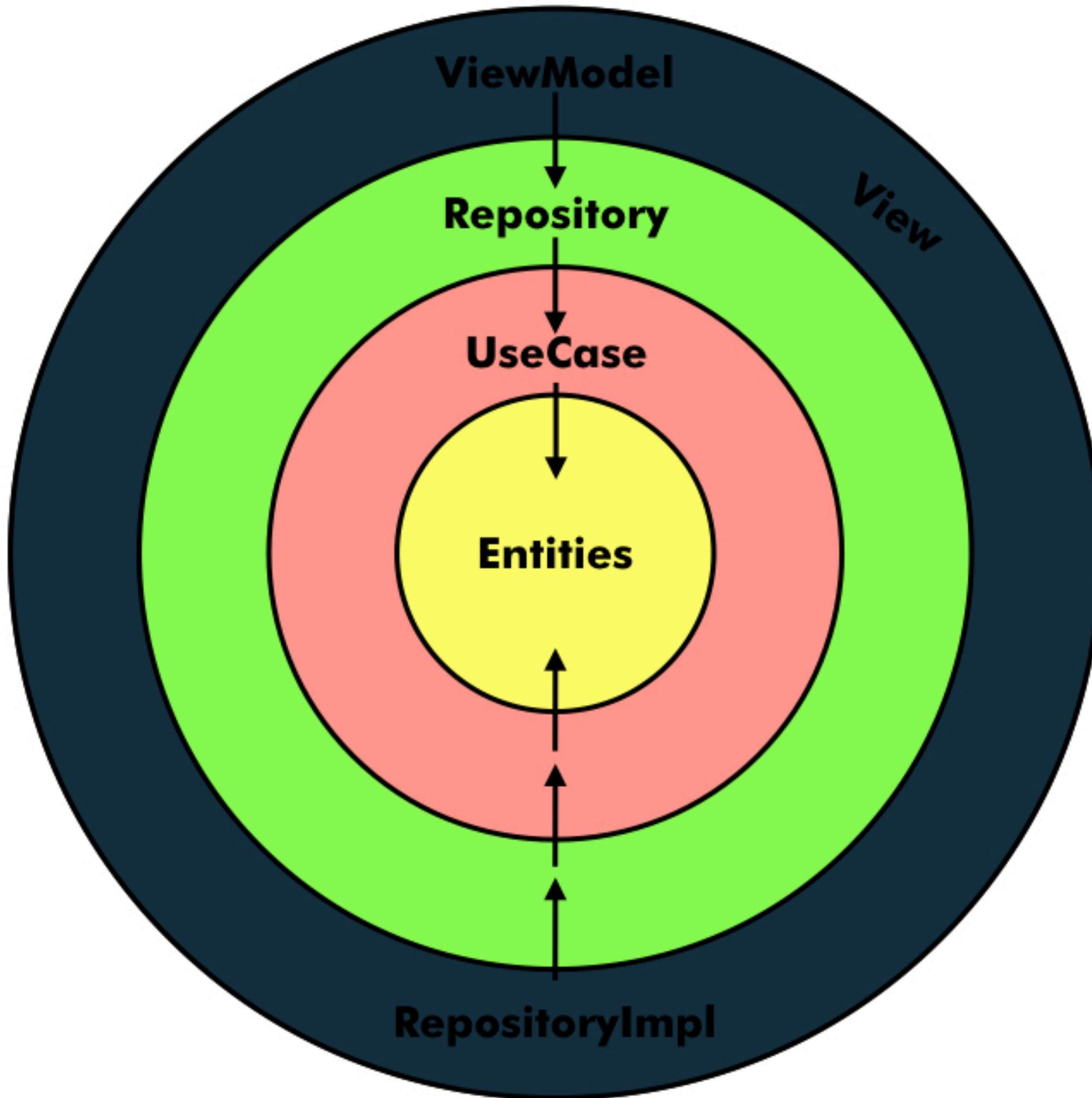
With Uncle Bob



```
fun onSwipeRefresh() {  
    refreshUseCase(position, coll, onSuccess, onFailure)  
}
```

# 예제 코드

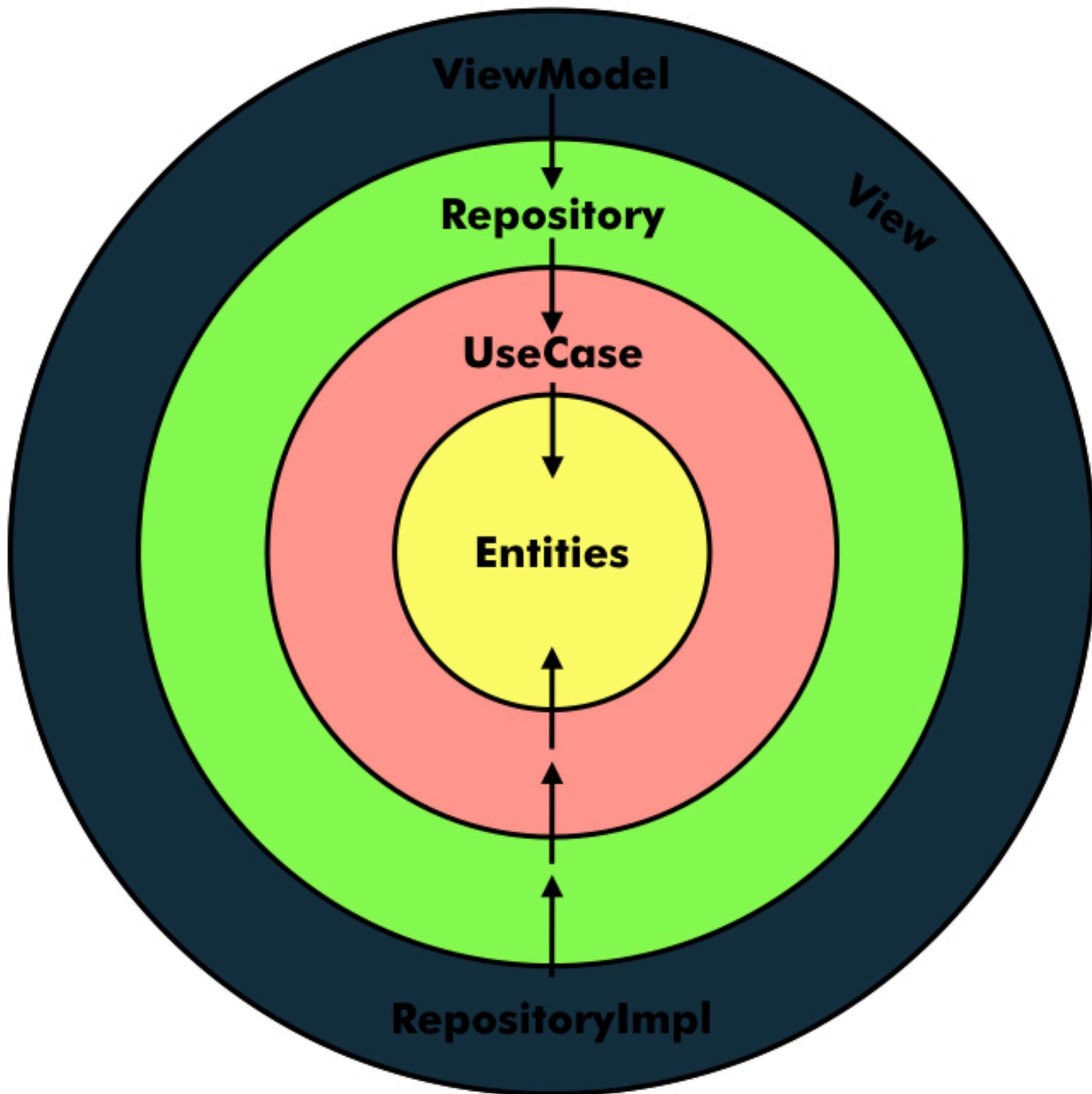
With Uncle Bob



```
class RefreshUseCase(private val repository: Repository) : UseCase() {  
    operator fun invoke(position: Int,  
                      coll: Item,  
                      scheduler: Scheduler,  
                      success: (coll: Item) -> Unit,  
                      error: (throwable: Throwable) -> Unit) {  
        disposable?.dispose()  
        disposable = repository.refresh(position, coll)  
            .observeOn(scheduler)  
            .subscribe(success, error)  
    }  
}
```

# 예제 코드

With Uncle Bob



```
override fun refresh(position: Int, item: Item): Single<Item> {  
    return networkDataSource.getData(headers, parameters)  
        .observeOn(Schedulers.computation())  
}
```

# **Strengths**

## Strengths

- 1. Independent of X**
- 2. Testable**
- 3. Separation of Concerns**

# **Weakness**

Weakness

- 1. learning curve**
- 2. Increase complexity**

**"A good architecture is an  
architecture that allows major  
decisions to be delayed!"**

**"To delay those decisions as long as possible so that you have the most information with which to make them."**

Uncle Bob

**Thank you**

# **Q&A**