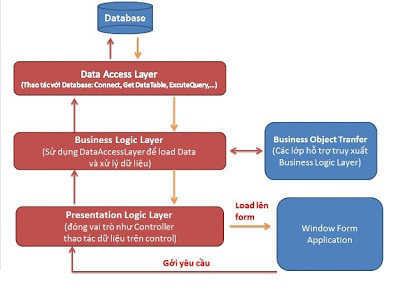
**Mô hình 3 Tier và 3 Layer**

[](https://phuongnguyenth.files.wordpress.com/2014/03/3tier.jpg)

**1.  Mô hình 3 Tier**

3-tiers là một kiến trúc kiểu **Client/Server** mà trong đó giao diện người dùng (**UI-User Interface**), các quy tắc xử lý(**BR-business rule** hay **BL-business logic**), và việc lưu trữ dữ liệu được phát triển như những module độc lập, và hầu hết là được duy trì trên các nền tảng độc lập, và mô hình 3 tầng (3-tiers) được coi là một kiến trúc phần mềm và là một mẫu thiết kế.

3-Tiers có tính vật lý (physical): là mô hình client-server (mỗi tier có thể đặt chung 1 nơi hoặc nhiều nơi, kết nối với nhau qua Web services, WCF, Remoting…).

Như vậy, ta có thể mô hình này phân tách ứng dụng ra làm 3 module riêng biệt, bao gồm:

– **Presentation tier:** bao gồm các thành phần phần xử lý giao diện **Graphic User Interface (GUI)**. Được dùng để giao tiếp với người dùng, nhiệm vụ chính là hiển thị dữ liệu và nhận dữ liệu từ người dùng.  
–  **Business tier:** gồm các thành phần **Business Logic Layer (BLL)**, **Data Access Layer (DAL)** và **Data Tranfer Object (DTO)**. Được dùng để cung cấp các chức năng của phần mềm  
–  **Data tier:** lưu trữ dữ liệu, là các hệ quản trị CSDL như **MS SQL Server**, **Oracle**,**SQLite**, **MS Access**, **XML files**, **text files**,…Dùng để lưu trữ dữ liệu, cho phép lớp**Business Logic** có thể tìm kiếm, trích xuất, cập nhật…

**2. Ưu điểm và nhược điểm mô hình 3 Tier**

Ưu điểm:

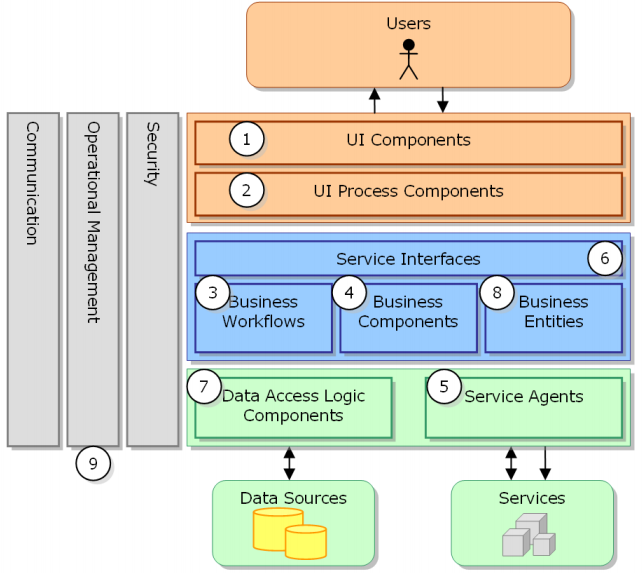
– Dễ dàng mở rộng, thay đổi quy mô của hệ thống: Khi cần tải lớn, người quản trị có thể dễ dàng thêm các máy chủ vào nhóm, hoặc lấy bớt ra trong trường hợp ngược lại.

Nhược điểm:

– Việc truyền dữ liệu giữa các tầng sẽ chậm hơn vì phải truyền giữa các tiến trình khác nhau (IPC), dữ liệu cần phải được đóng gói -> truyền đi -> mở gói trước khi có thể dùng được.

– Việc phát triển ứng dụng phức tạp hơn.

**3. Mô hình 3 Layer**

[](https://phuongnguyenth.files.wordpress.com/2014/03/3layer.png)

Mô hình 3-Layers có tính logic (mỗi layer có 1 công việc) và là 1 thành phần của 3-Tiers. được cấu thành từ: **Presentation Layers**, **Business Layers**, và **Data Layers**. Các lớp này sẽ giao tiếp với nhau thông qua các dịch vụ (**Services**) mà mỗi lớp cung cấp để tạo nên ứng dụng, lớp này cũng không cần biết bên trong lớp kia làm gì mà chỉ cần biết lớp kia cung cấp dịch vụ gì cho mình và sử dụng nó mà thôi. Gồm 3 thành phần :

– **Graphic User Interface (GUI)**: Thành phần giao diện, là các form của chương trình tương tác với người sử dụng.  
–  **Business Logic Layer (BLL)**: Xử lý các nghiệp vụ của chương trình như tính toán, xử lý hợp lệ và toàn vẹn về mặt dữ liệu.  
–  **Data Access Layer (DAL)**: Tầng giao tiếp với các hệ quản trị CSDL

**4. Lý do dùng mô hình 3 Tier**

        Khi dùng từ layer, chúng ta nói tới việc phân chia ứng dụng thành các thành phần một cách logic theo chức năng hoặc theo vai trò, điều này giúp phần mềm của bạn có cấu trúc sáng sủa, dễ dùng lại, từ đó giúp việc phát triển và bảo trì dễ dàng hơn. Các layer khác nhau khi được thực thi vẫn có thể nằm trong cùng một vùng bộ nhớ của một process, và hiển nhiên việc giao tiếp giữa 2 layer có thể không phải là giao tiếp giữa 2 process, đồng nghĩa với việc chúng không liên quan tới mô hình**Client/Server.**

Trái lại, tier liên quan đến cách phân chia một cách vật lý các thành phần trên các máy tính khác nhau.

Điều làm nhiều người nhầm lẫn giữa layer và tier là chúng có cùng cách phân chia (presentation, business, data), tuy nhiên trên thực tế chúng khác nhau. Vì cách phân chia như trên nên 1 tier có thể chứa nhiều hơn 1 layer.

**5. Lớp Data Access Layer (DAL) thuộc tầng nào?**

**Lớp Business Logic.**

Trái với nhiều người nghĩ, cứ cái gì có chữ Data thì nó phải thuộc lớp 3, tuy nhiên vì**DAL** chỉ đóng vai trò truy vấn, chứ bản thân nó không cung cấp dữ liệu, và nó vẫn phải được thực thi bởi các Business Object, vậy nên trong đa số trường hợp nó sẽ nằm trong lớp 2 (một số thiết kế tách nó riêng thành 1 tier).

Nên nhớ rằng việc tách riêng ra một **DAL** giúp bạn có một thiết kế tốt hơn, nhưng không phải là bắt buộc. Và việc tự tạo một DAL với việc dùng chung một tập các lớp truy xuất dữ liệu được cung cấp bởi một công nghệ/công cụ có sẵn như **LINQ to SQL**,**NHibernate** hay **Entity Framework** không có gì khác nhau về kiến trúc hệ thống.

Có lẽ vì sự tồn tại của **DAL** mà rất nhiều người hiểu nhầm giữa 3-tiers và 3-layers.