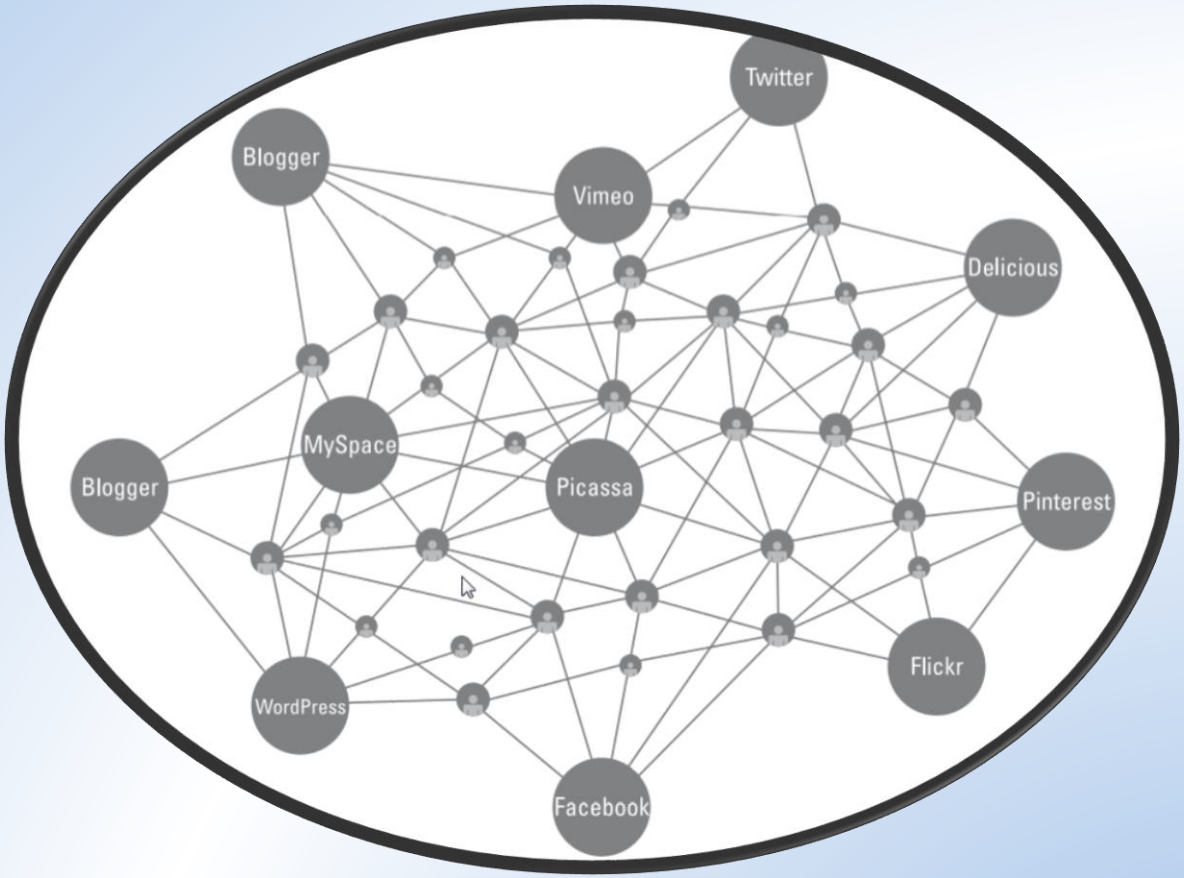


APP ARCHITECTURE OVERVIEW

Presenter: PhuongNQK

Goal

- Give an overview of currently prominent app architectures in relation to each other

Old lessons

- What is a software app?
  - A system designed to automate specific tasks in a logical manner to satisfy a set of requirements
- Basic design principles
  - Interface > Implementation
  - Modularization
- Design views
  - Conceptual -> Logical -> Physical

What is app architecture?

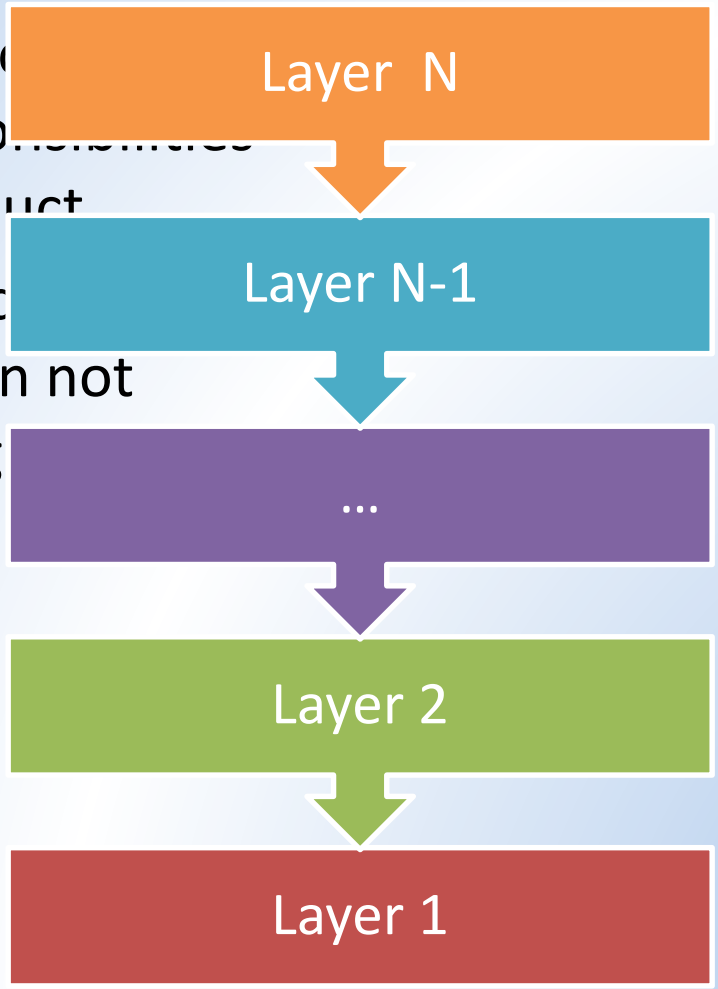
- Organizational design of an entire software application, including all sub-components and external applications interchanges
- Goals:
  - Complete the necessary business processes as defined in the system requirements
  - Support future growth

What will we look at?

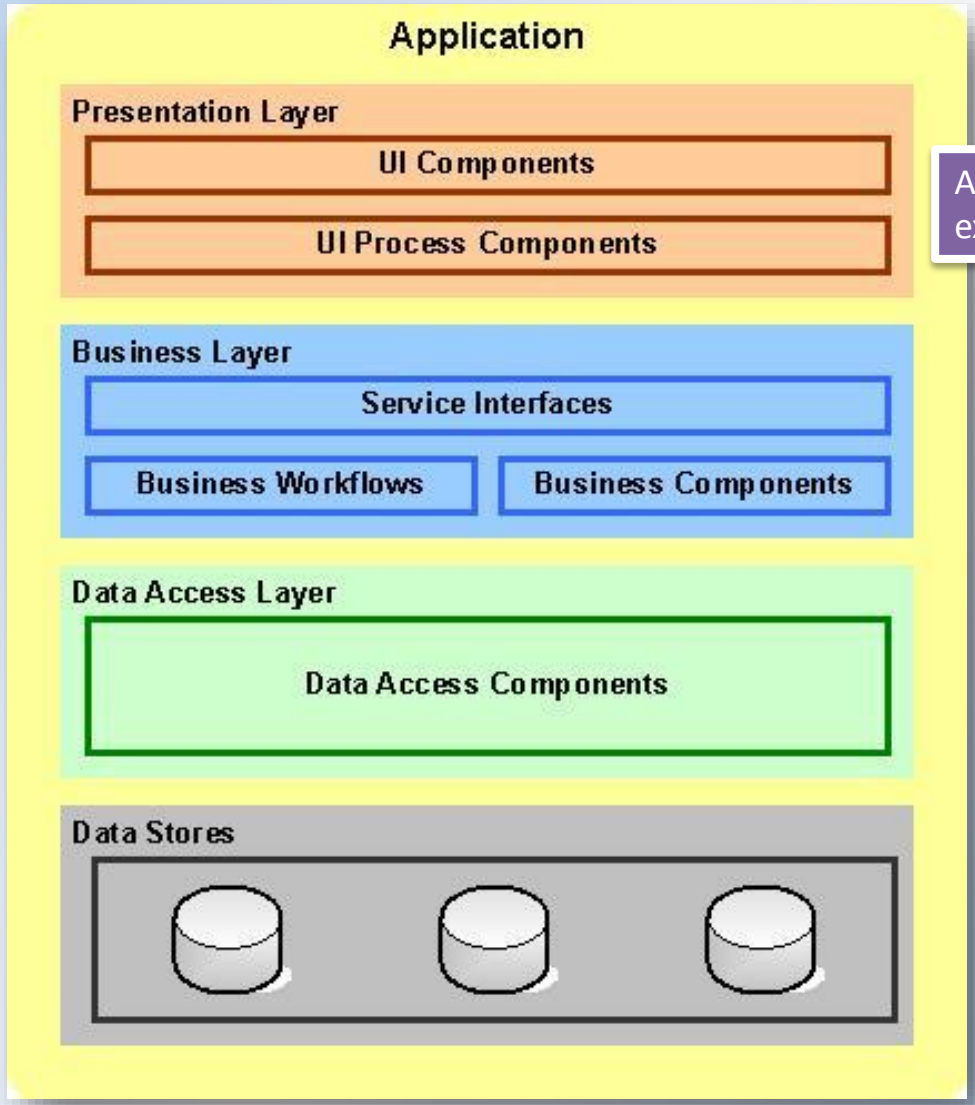
- Layered architecture
- DDD (**D**omain-**D**riven **D**esign)
- Clean architecture
- SOA (**S**ervice-**O**riented **A**rchitecture)
- Cloud-based architecture

Layered architecture

- Use n layers for all
  the different respo
  of a software prod
- A layer can depend
  lower layer, but can not
  depend on any hig
- Layer vs. tier

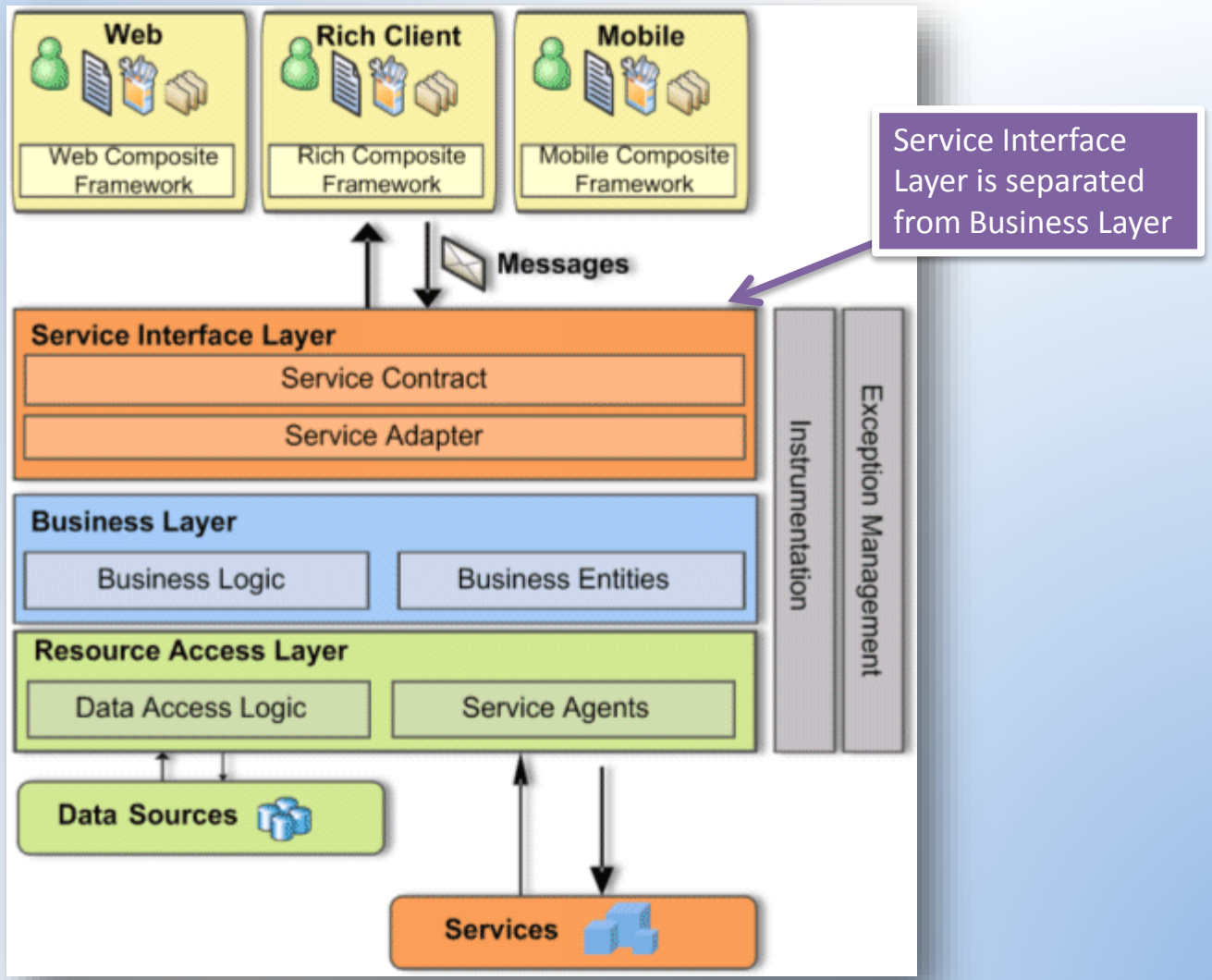


Traditional examples

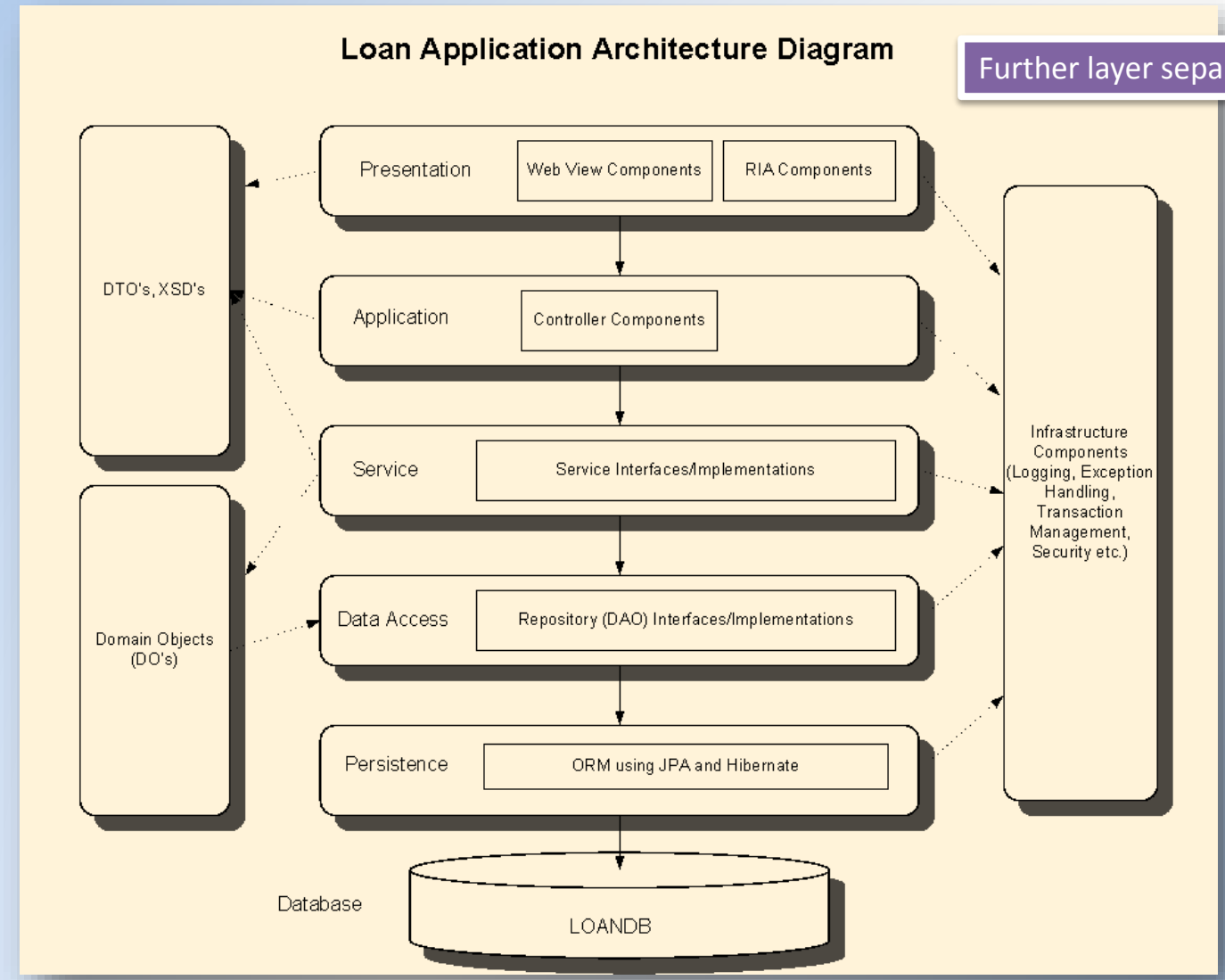


A simple, typical example

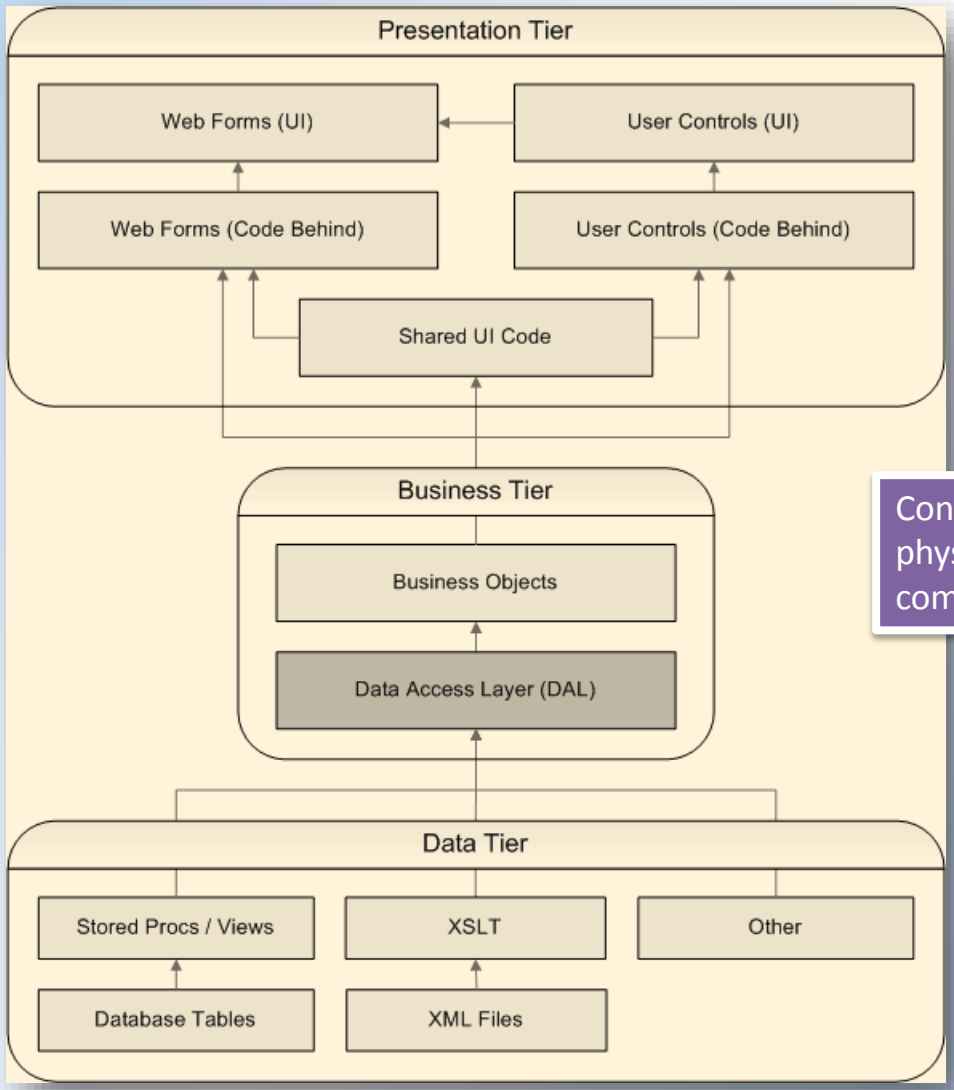
Traditional examples



Traditional examples

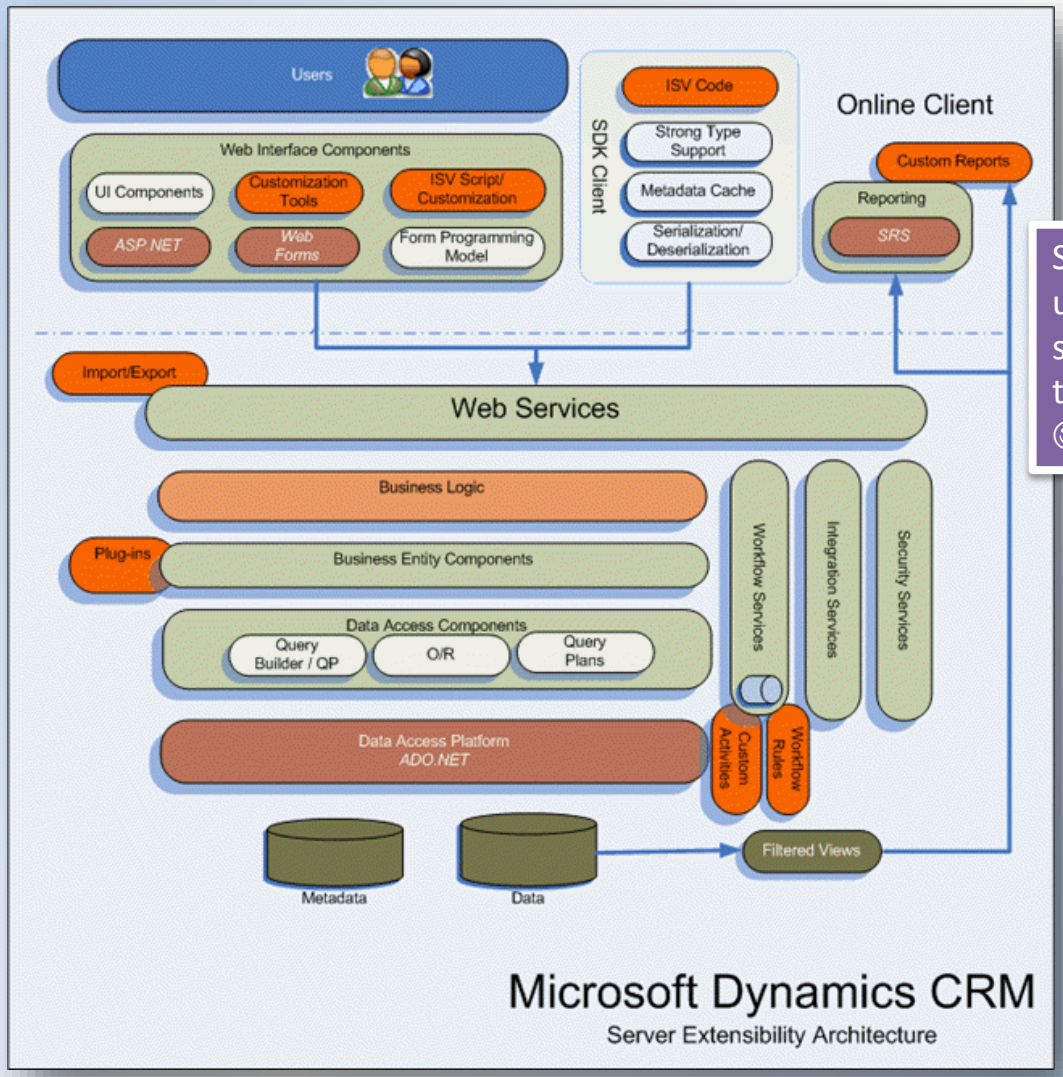


Traditional examples



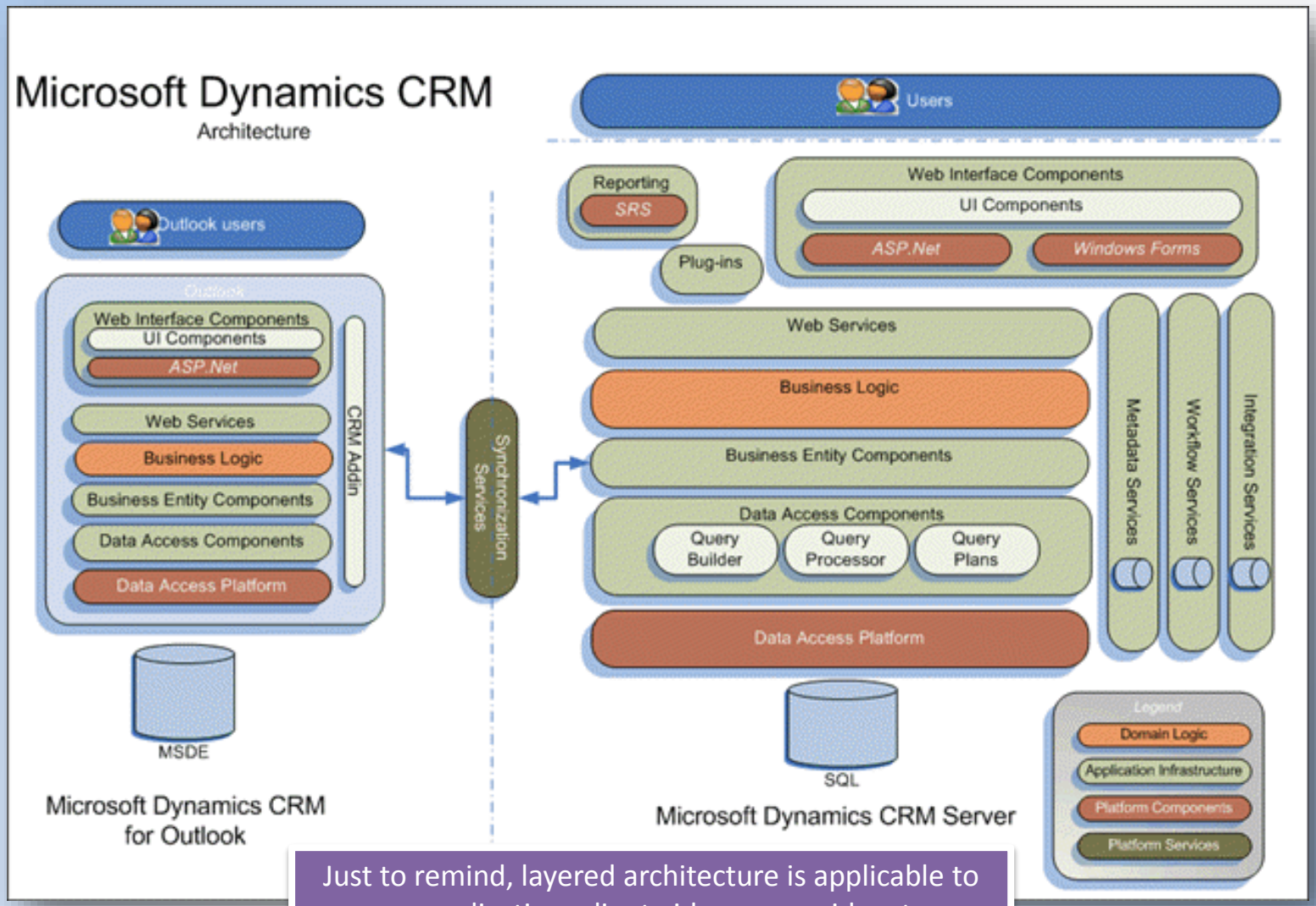
Conceptual and physical views combined

Traditional examples



So easy to understand something like this now, right ☺?

Traditional examples



Just to remind, layered architecture is applicable to any application: client-side, server-side, etc.

Q & A

- What's wrong with the examples?
  - They are similar, but not standardized.
- How can we standardize them?
  - DDD.

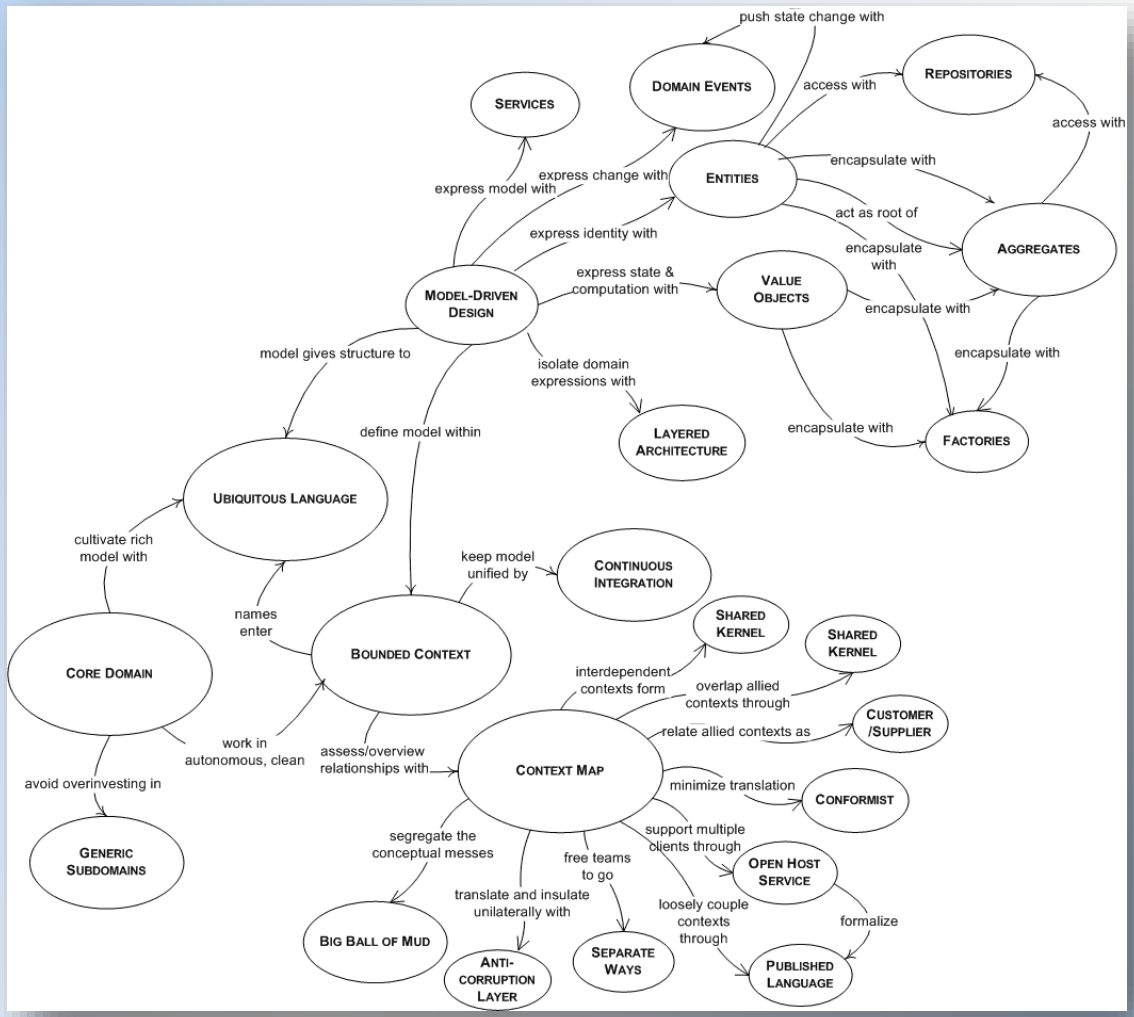
What is Domain Driven Design?

- A standardization of the best implementations of layered architecture
- An approach to software development for complex needs by connecting the implementation to an evolving model

DDD premises

- Primary focus = Domain layer
- Base complex designs on a model of the domain
- Initiate a creative collaboration between technical and domain experts to iteratively refine a conceptual model that addresses particular domain problems

DDD Vocabulary

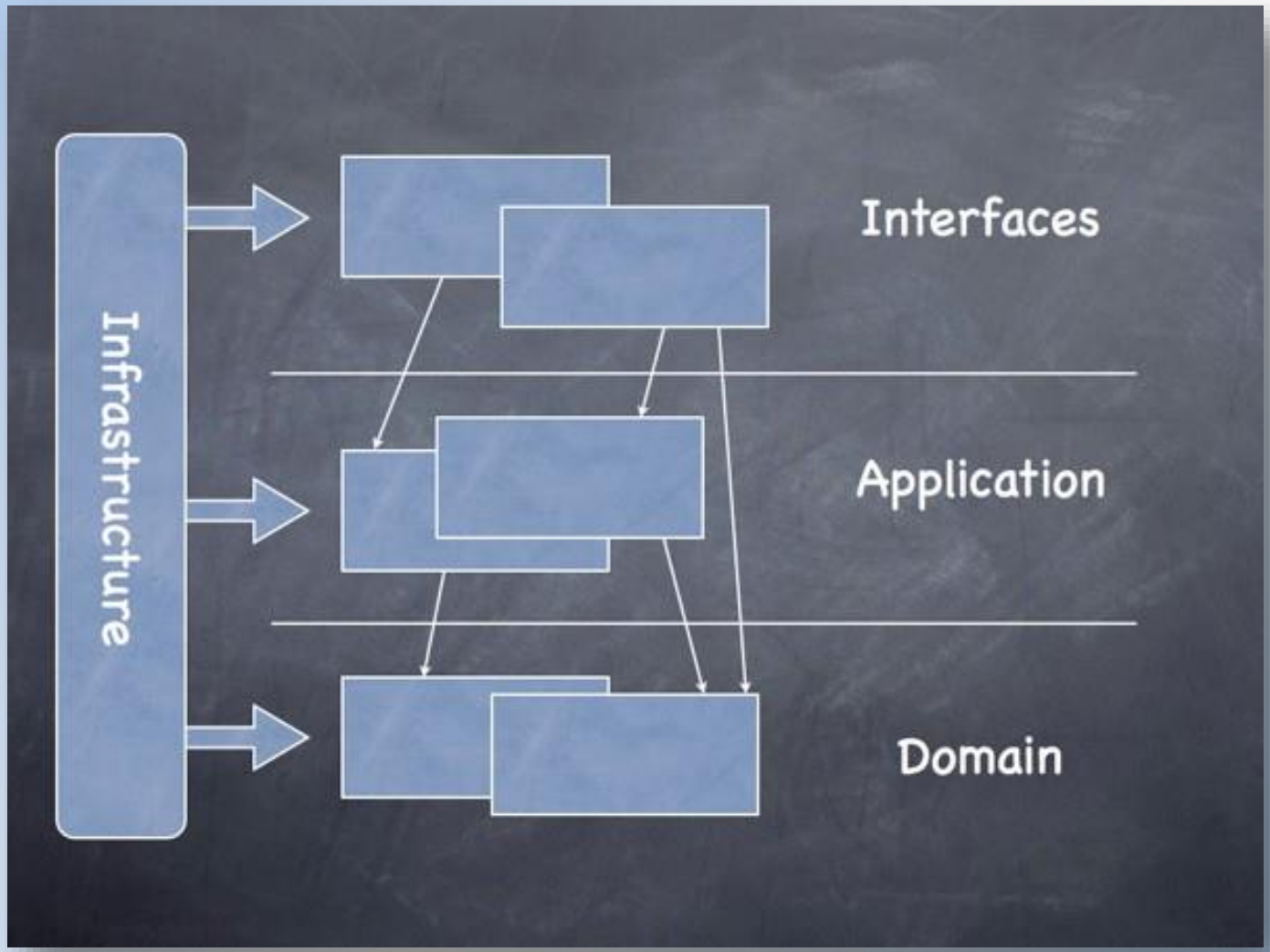


Visit here: http://en.wikipedia.org/wiki/Domain-driven_design

DDD – Typical layers

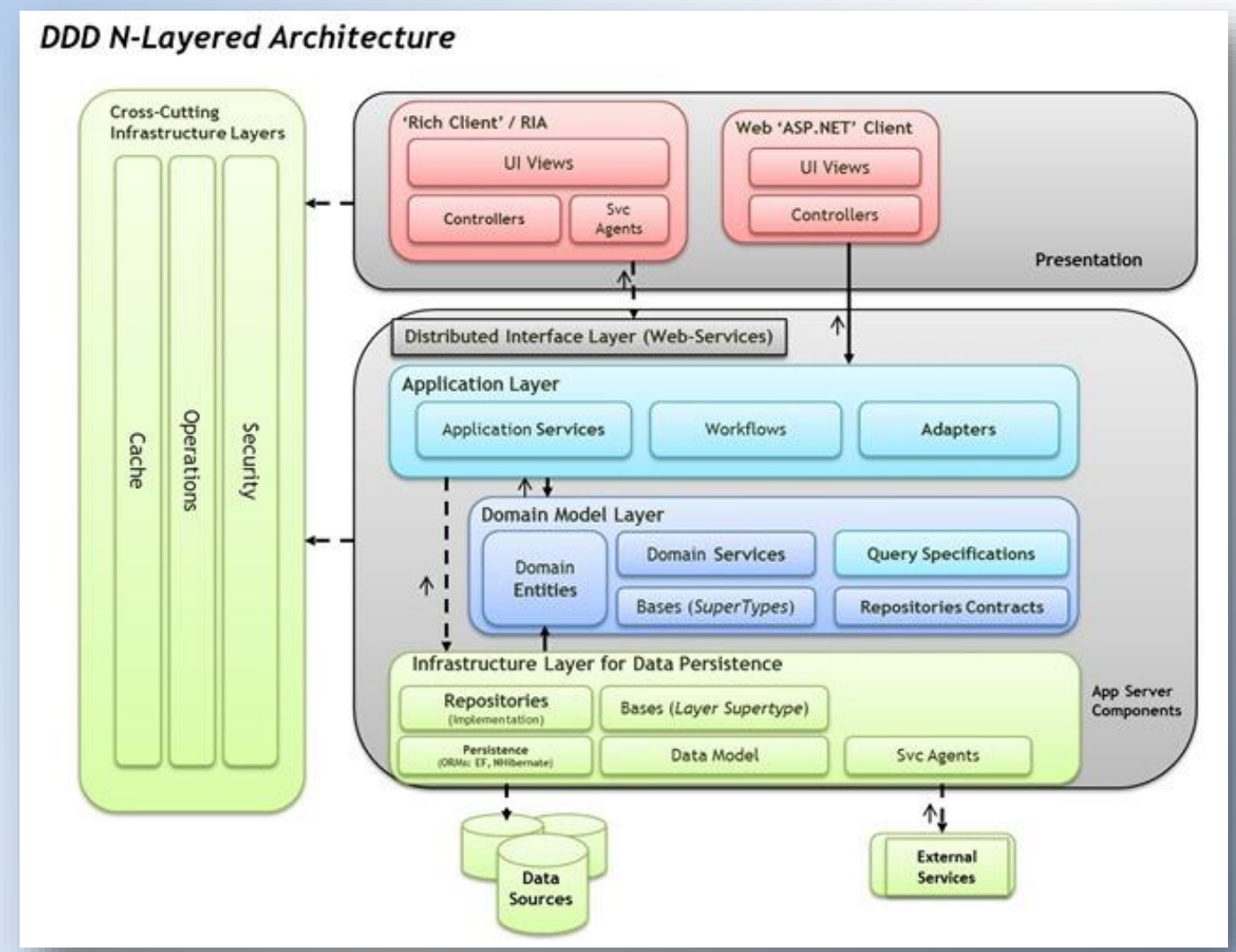
User Interface Layer	Accepts user commands and presents information back to the user
Application Layer	Manages transactions, translates DTOs, coordinates application activities, creates and accesses domain objects
Domain Layer	Contains the state and behavior of the domain
Infrastructure Layer	Supports all other layers, includes repositories, adapters, frameworks etc.

DDD – Typical layers

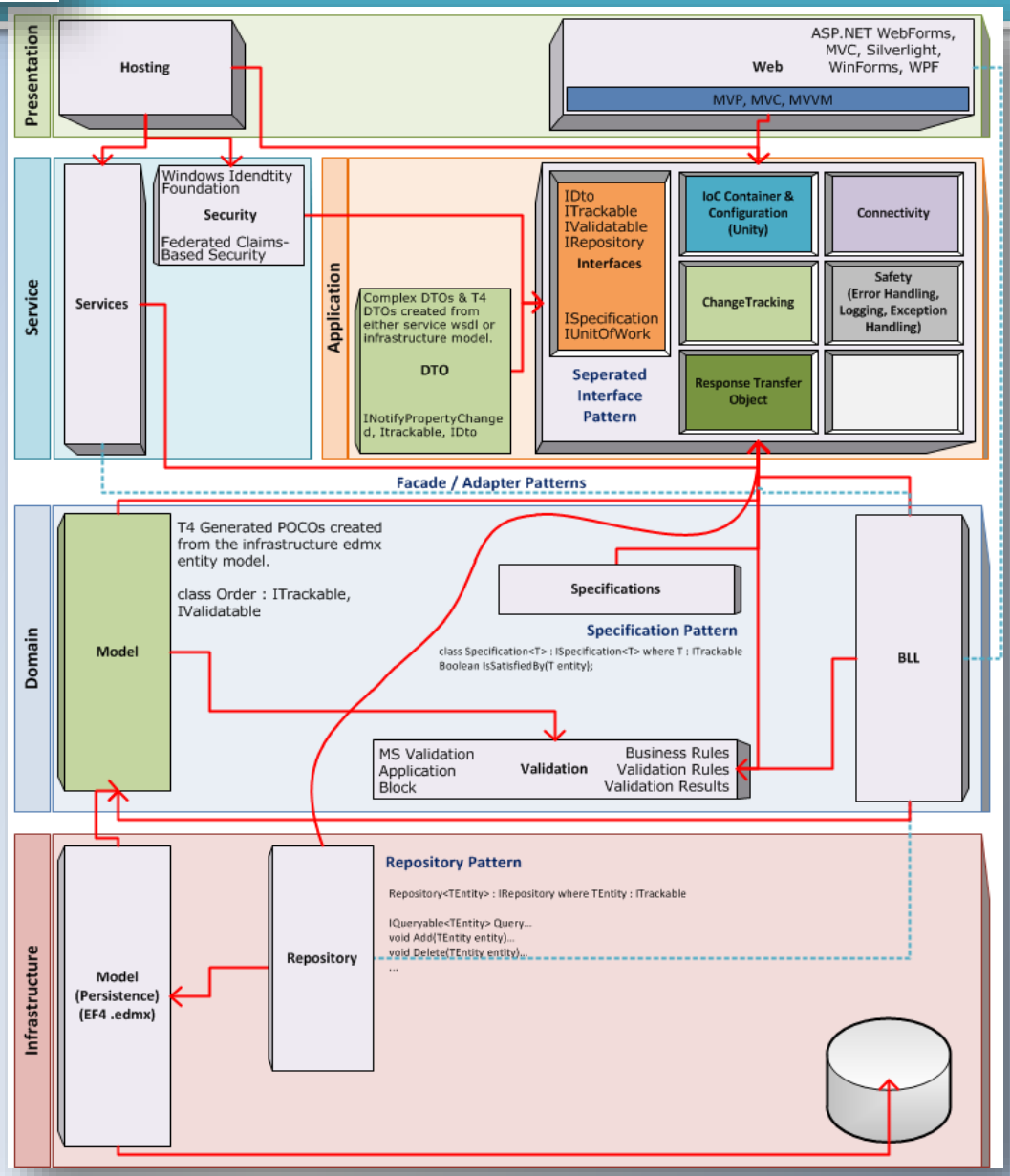


Visit here: http://dddsample.sourceforge.net/architecture.html

DDD N-layer architecture



A bit deeper dive



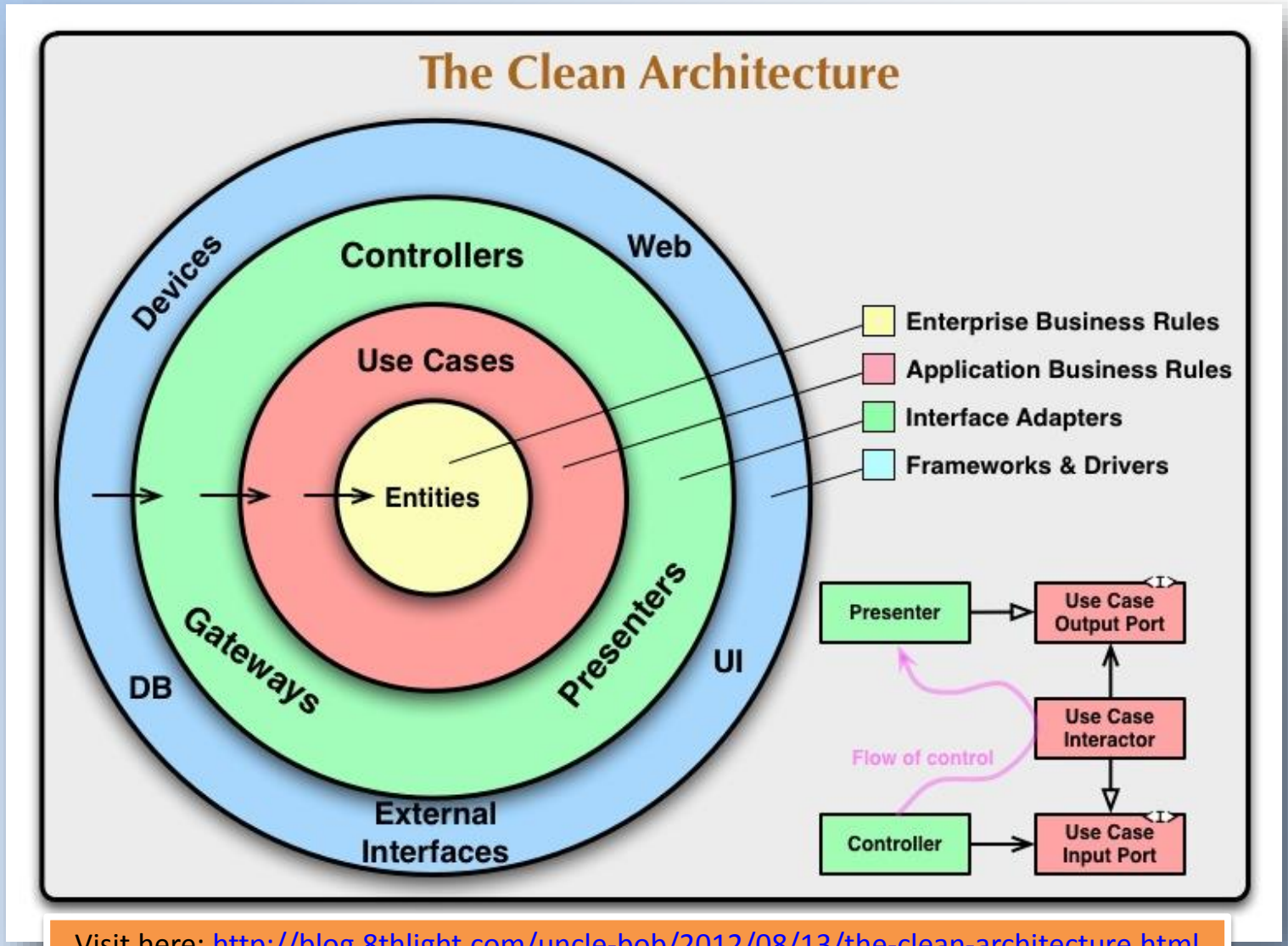
Q & A

- What's wrong with that implementation?
  - Domain layer depends on Infrastructure layer.
- Why is such dependency wrong?
  - Because real-world domains do not.
- How to remove that dependency?
  - Clean architecture

What is clean architecture?

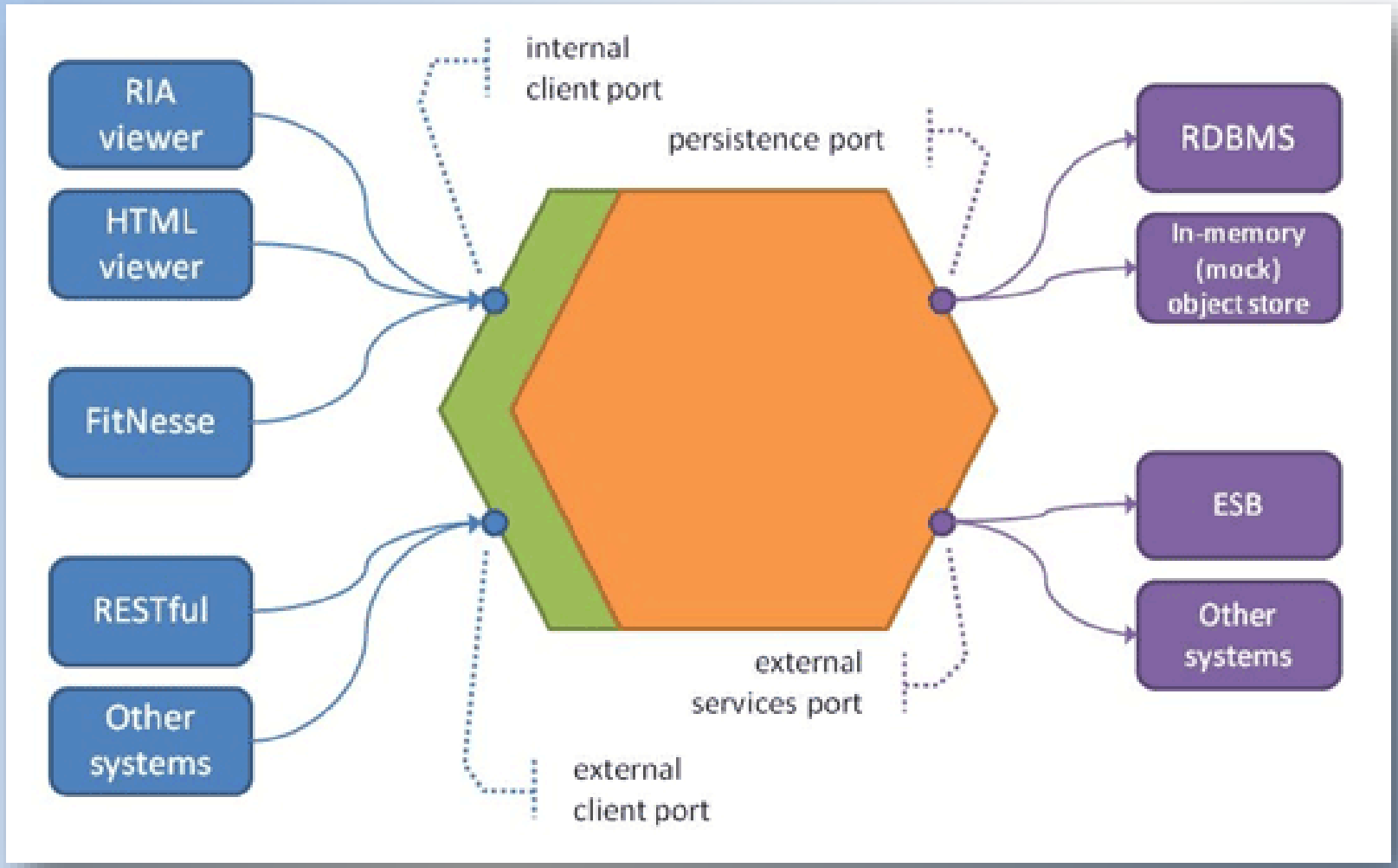
- Aim to produce systems:
  - Testable
  - Independent of
    - Frameworks
    - UI
    - Database
    - Any External Agency
- Dependency rule:
  - Source code dependencies can only point inwards

Clean architecture



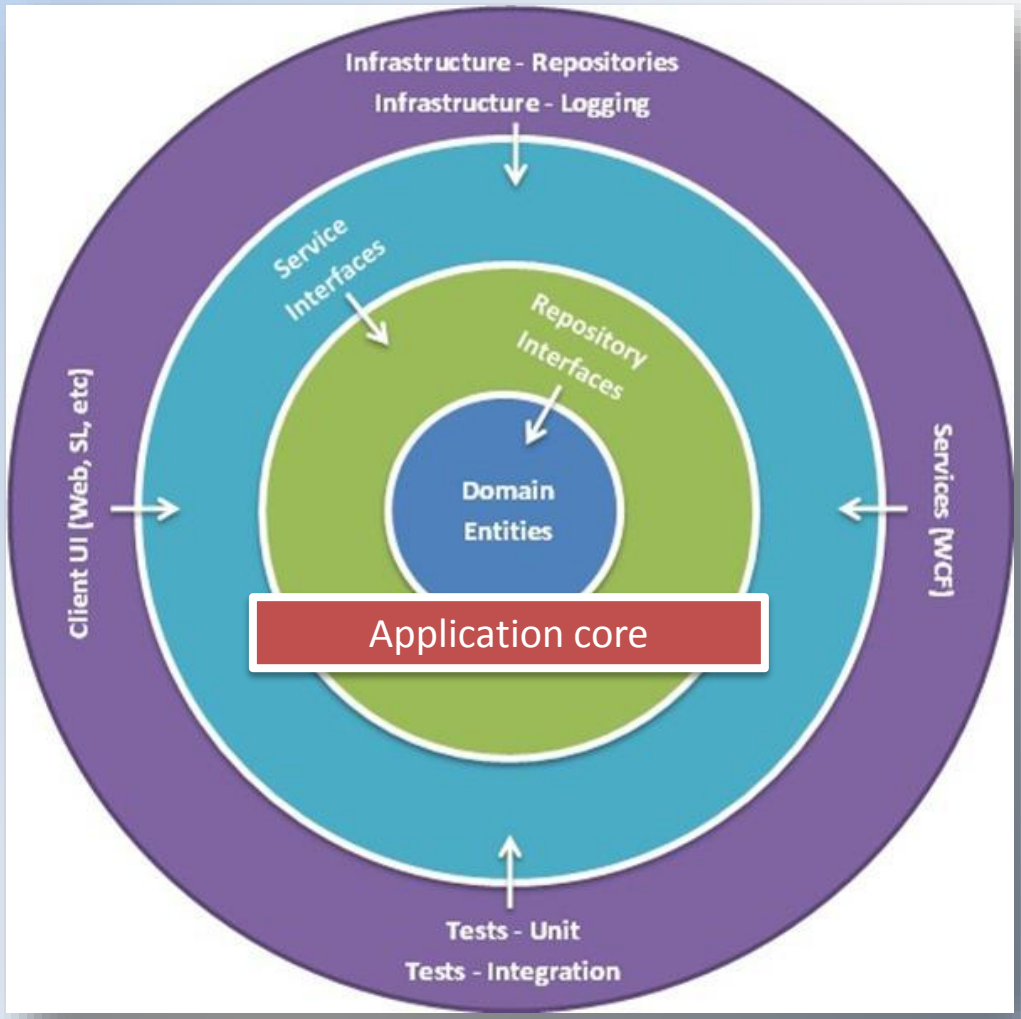
Visit here: http://blog.8thlight.com/uncle-bob/2012/08/13/the-clean-architecture.html

Examples



Hexagonal architecture: http://alistair.cockburn.us/Hexagonal+architecture

Onion architecture

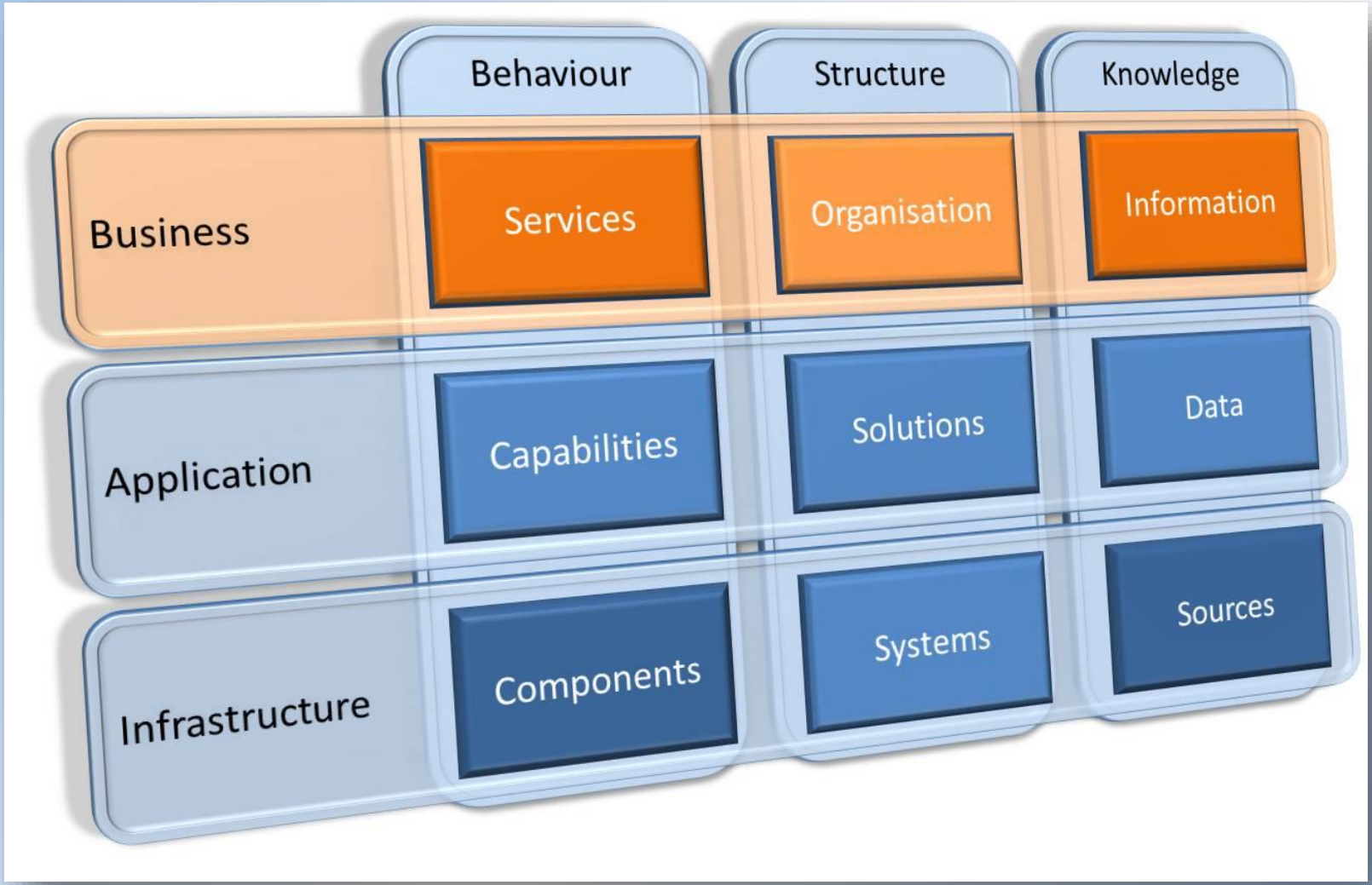


Onion architecture: http://jeffreypalermo.com/blog/the-onion-architecture-part-1/

Q & A

- What's wrong with all aforementioned stuffs?
  - They have not focused on delivering sets of functionality independently as well as flexibly integrating with other apps.
- But they can, using components / modules.
  - Not as well as SOA services.
- How better?
  - They help to maximize the business values. You can deliver smaller parts at cheaper costs, and you can deliver more by better integrating with other apps.

From business view



Enterprise Architecture: http://theenterprisingarchitect.blogspot.com/

Needs for SOA

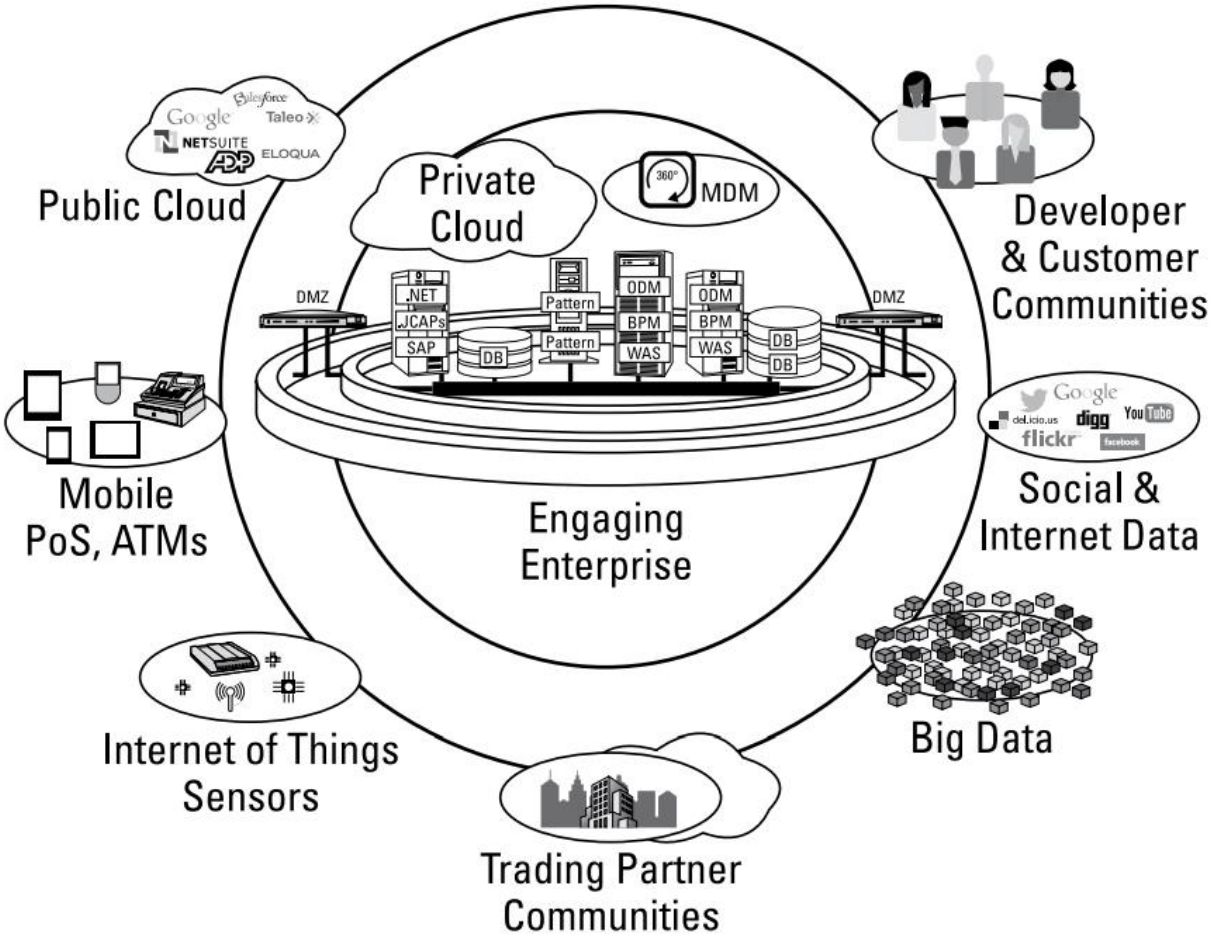


Figure 1-2: An engaging enterprise is interaction-centric.

SOA principles

- Service orientation at the core
- Process integrity at Internet scale
- Integration with enterprise capibilities and backend systems
- A basis in industry standards
- Leveraging and extending open-source techs
- Providing the platform for a growing ecosystem

SOA platform

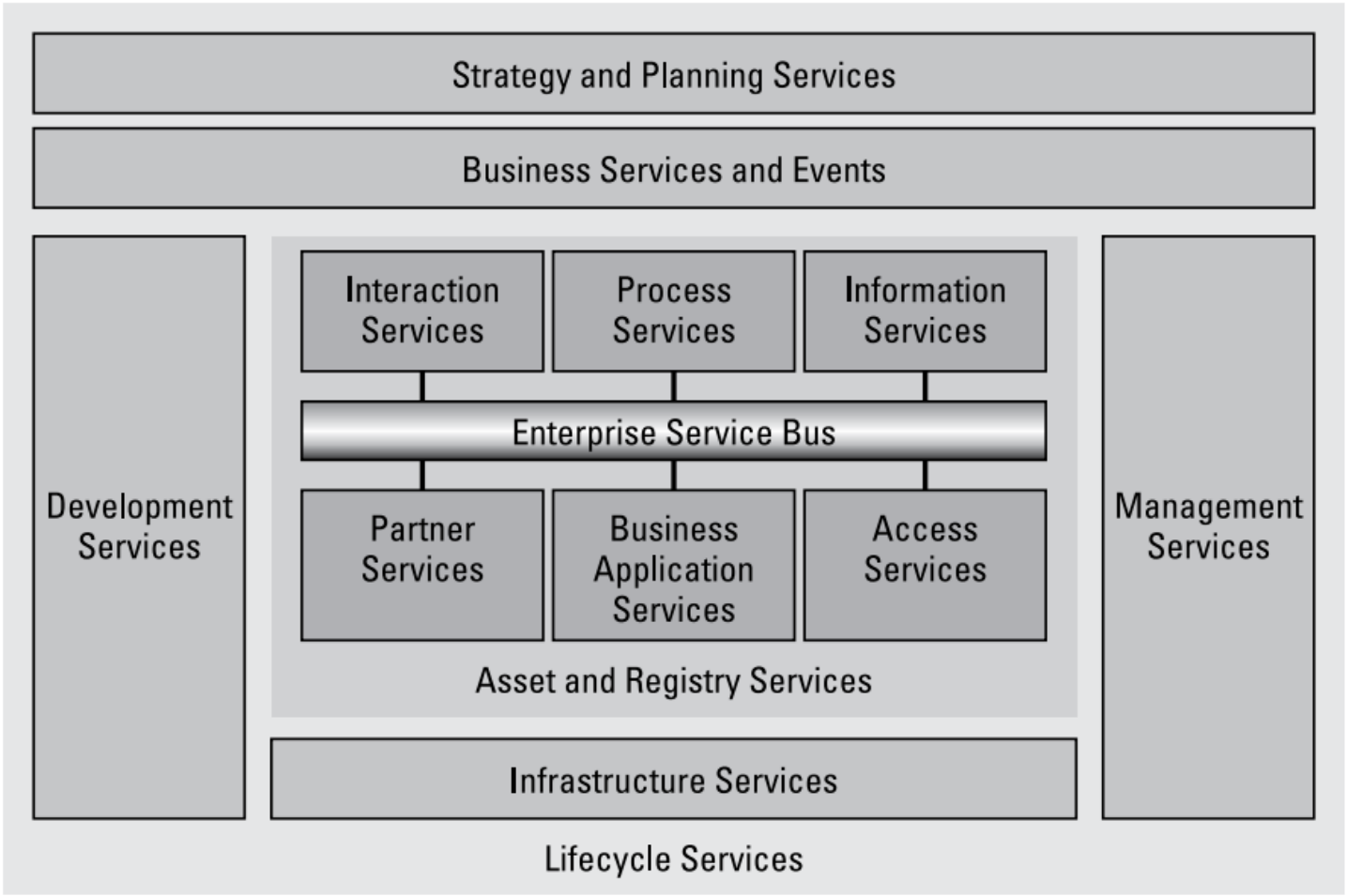
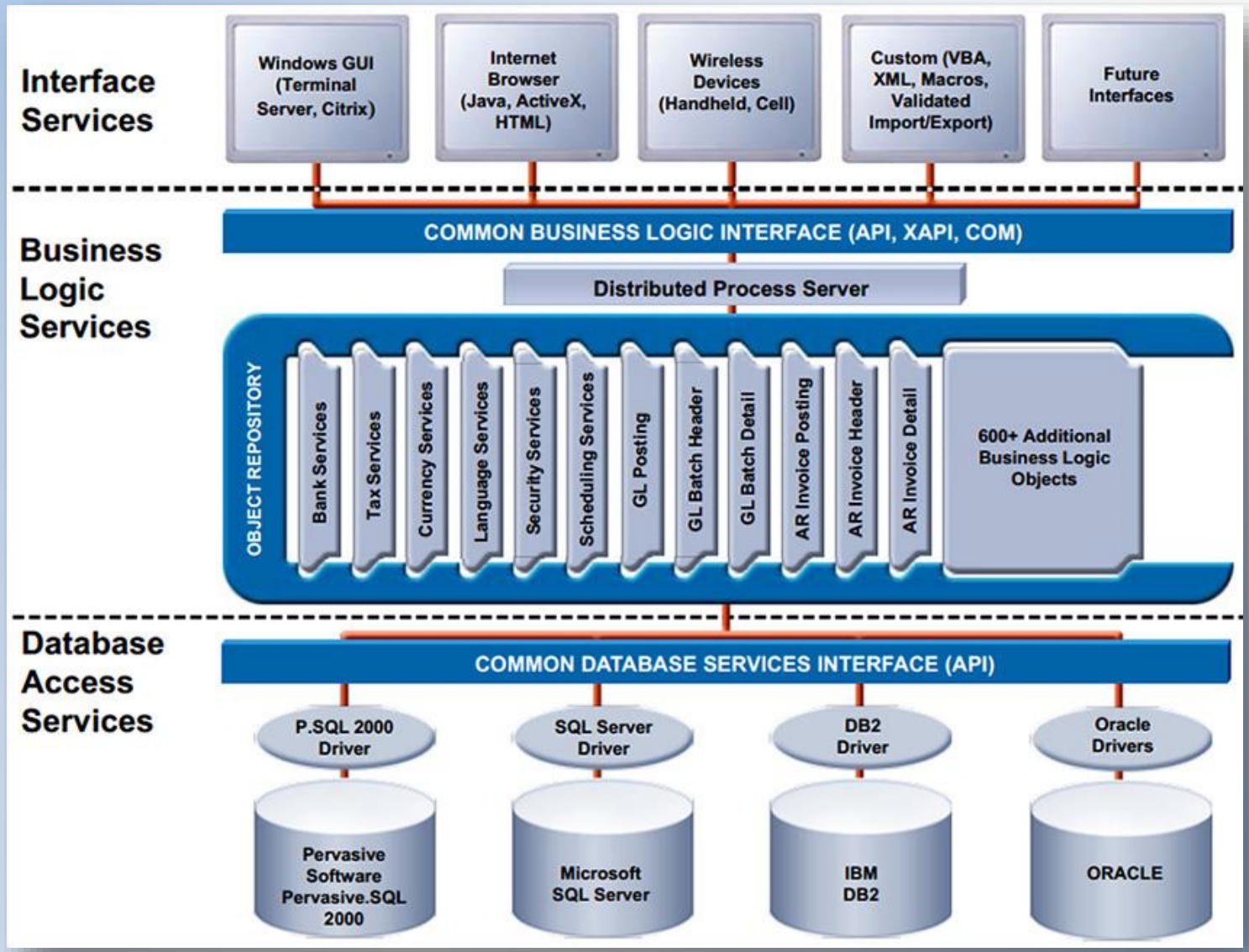


Figure 2-1: The IBM SOA Reference Model defines a good SOA platform.

Example


Interface Services
Windows GUI (Terminal Server, Citrix)
Internet Browser (Java, ActiveX, HTML)
Wireless Devices (Handheld, Cell)
Custom (VBA, XML, Macros, Validated Import/Export)
Future Interfaces
Business Logic Services
COMMON BUSINESS LOGIC INTERFACE (API, XAPI, COM)
Distributed Process Server
OBJECT REPOSITORY
Bank Services
Tax Services
Currency Services
Language Services
Security Services
Scheduling Services
GL Posting
GL Batch Header
GL Batch Detail
AR Invoice Posting
AR Invoice Header
AR Invoice Detail
600+ Additional Business Logic Objects
Database Access Services
COMMON DATABASE SERVICES INTERFACE (API)
P.SQL 2000 Driver
SQL Server Driver
DB2 Driver
Oracle Drivers
Pervasive Software Pervasive.SQL 2000
Microsoft SQL Server
IBM DB2
ORACLE

Q & A

- What's wrong with SOA?
  - It's just one side of a coin
- What coin?
  - Cloud computing
- What's the other side?
  - Dynamic infrastructure
- Oh, my god! What are they all about?

Sample need: Socializing with SOA

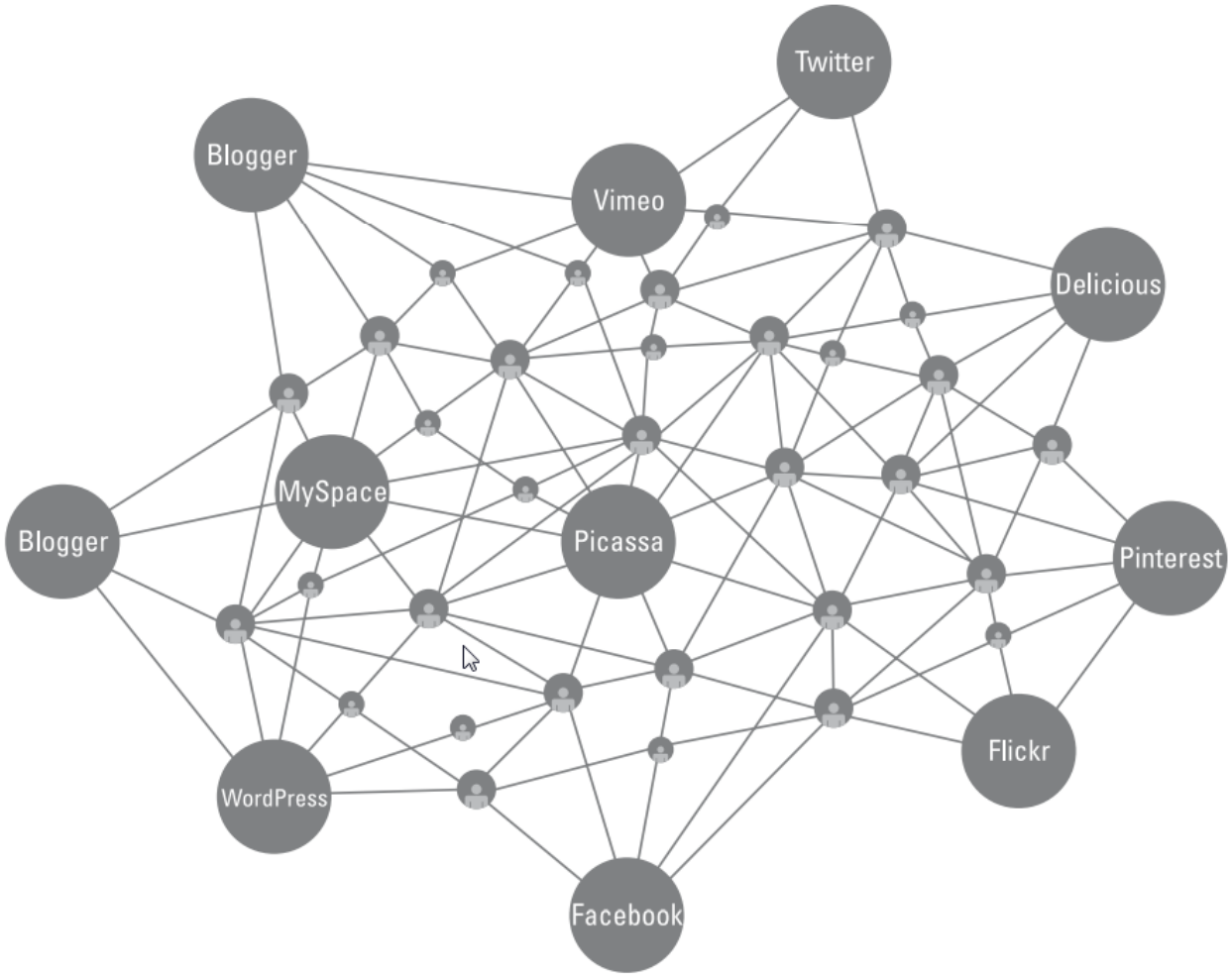
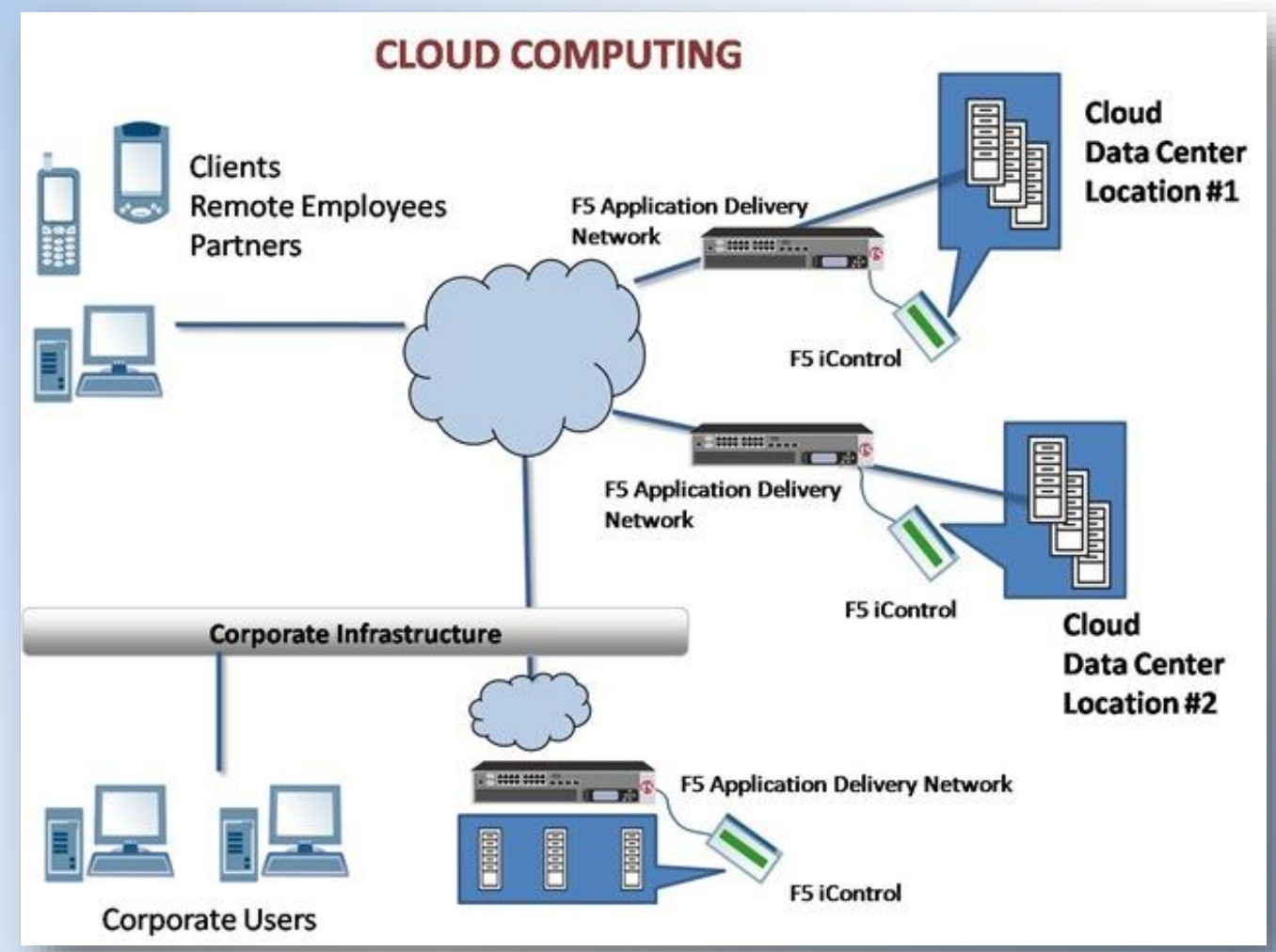


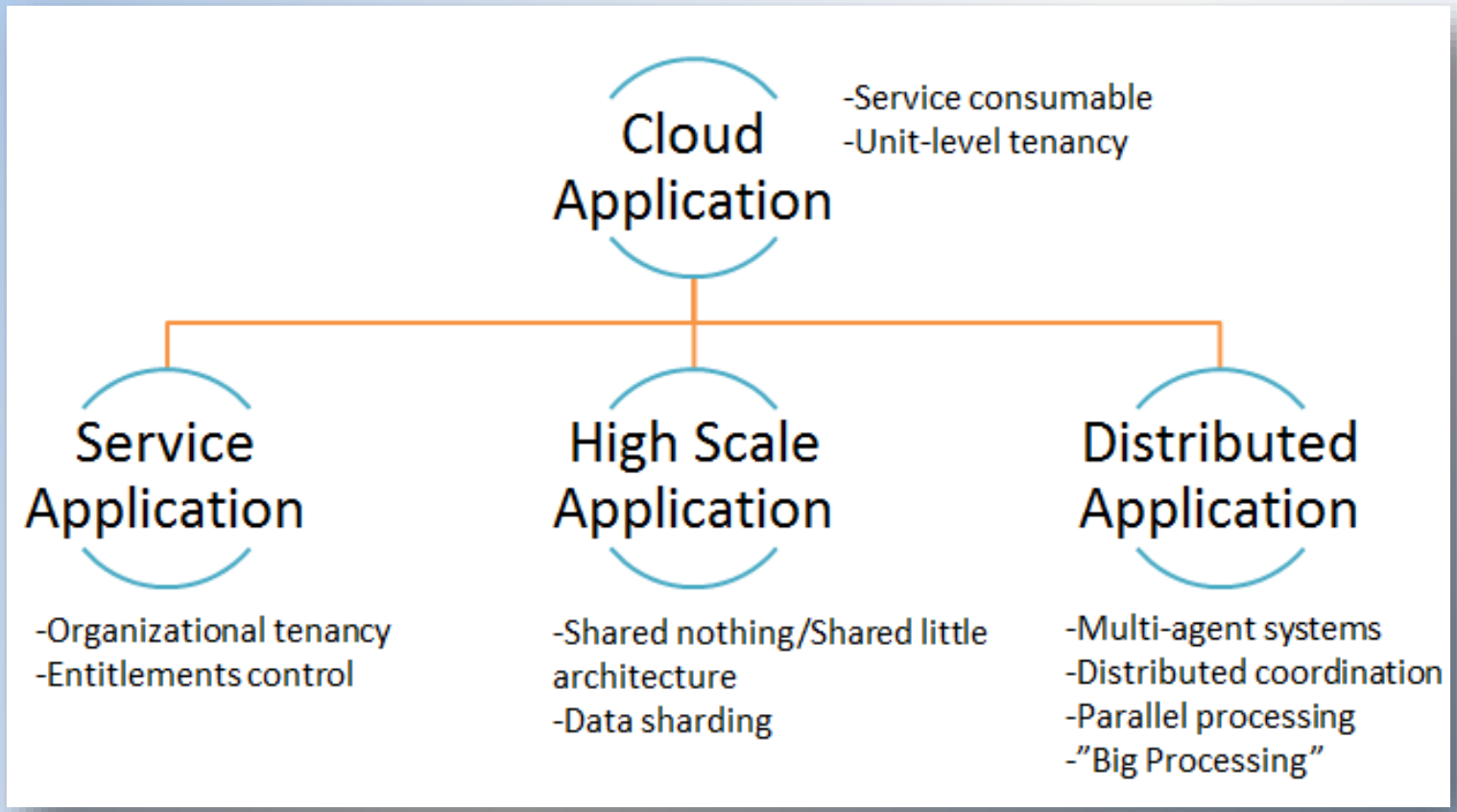
Figure 4-1: Interacting through the social graph.

Cloud computing



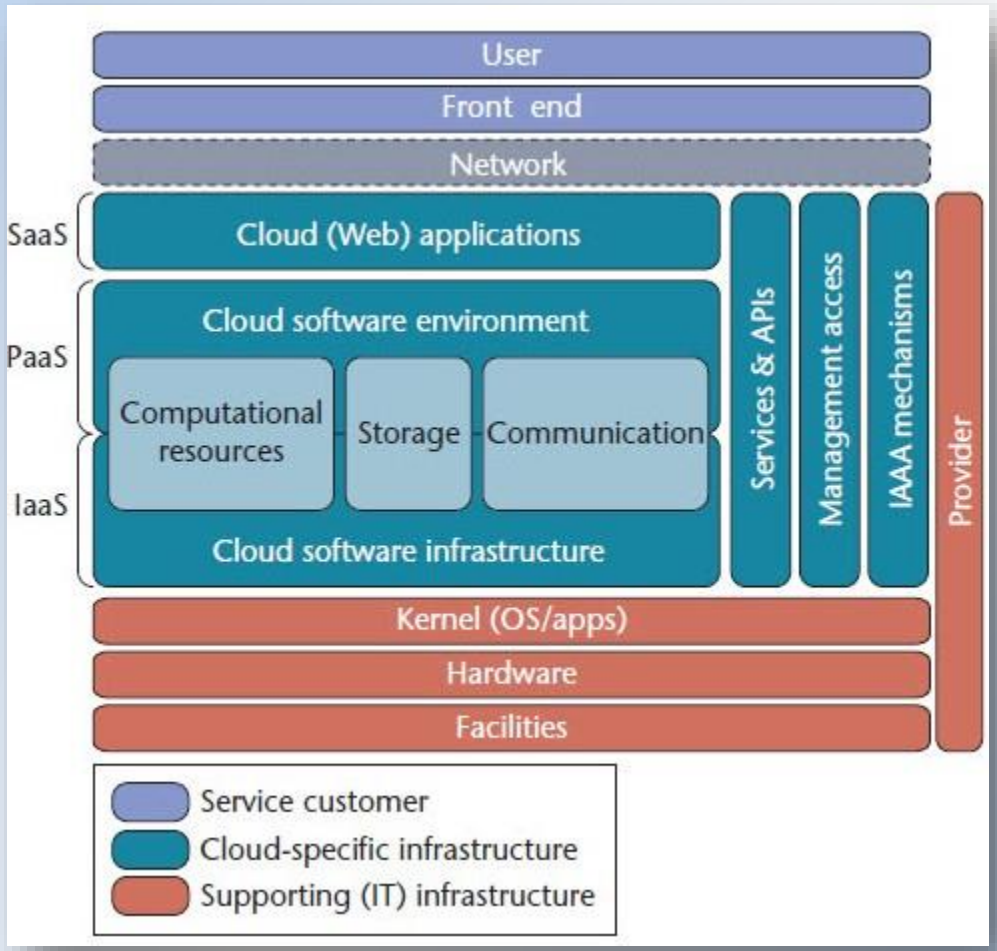
We already have Internet as the backbone

Cloud application



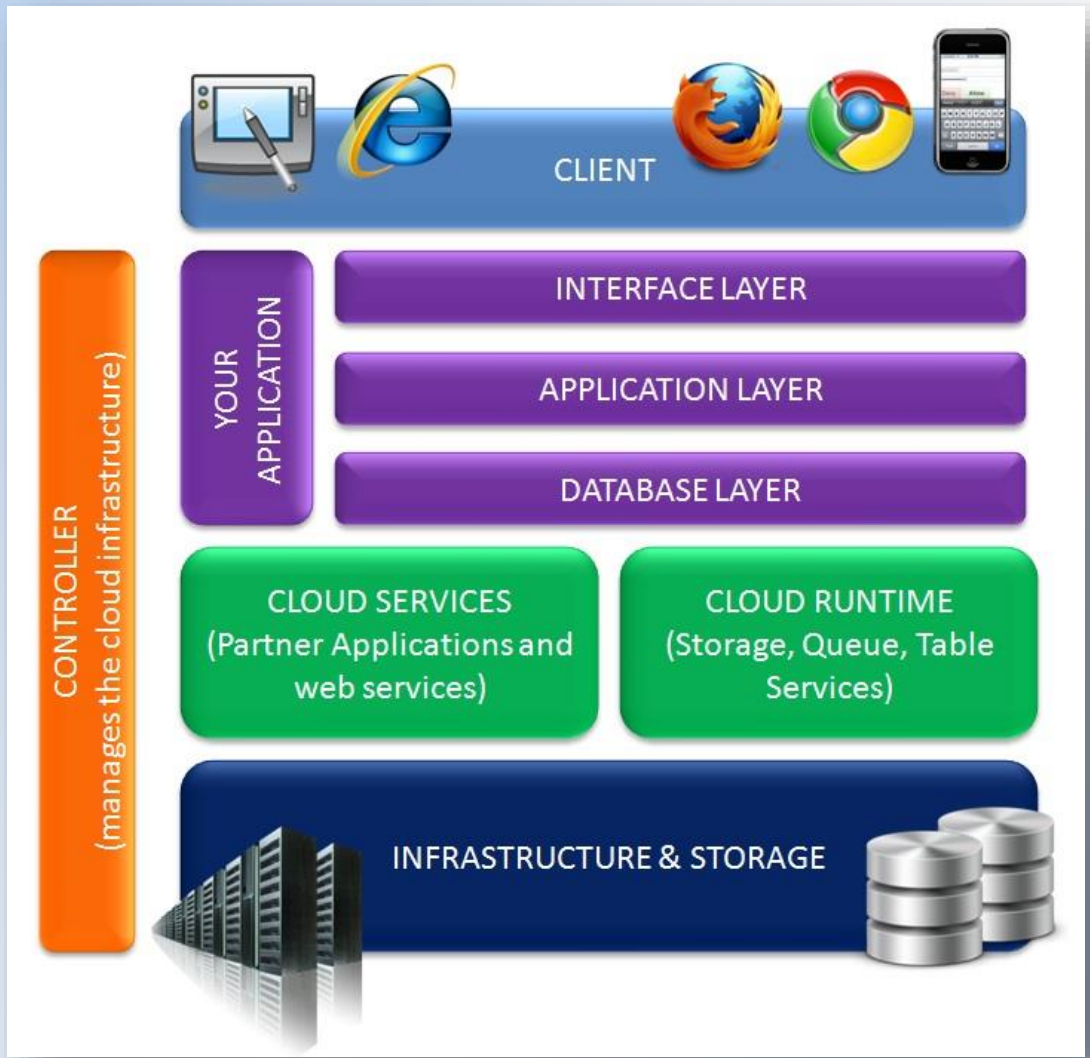
What remain are our applications need to be cloud-ready
http://apprenda.com/blog/answer-this-what-is-a-cloud-application-to-you/

Dynamic infrastructure



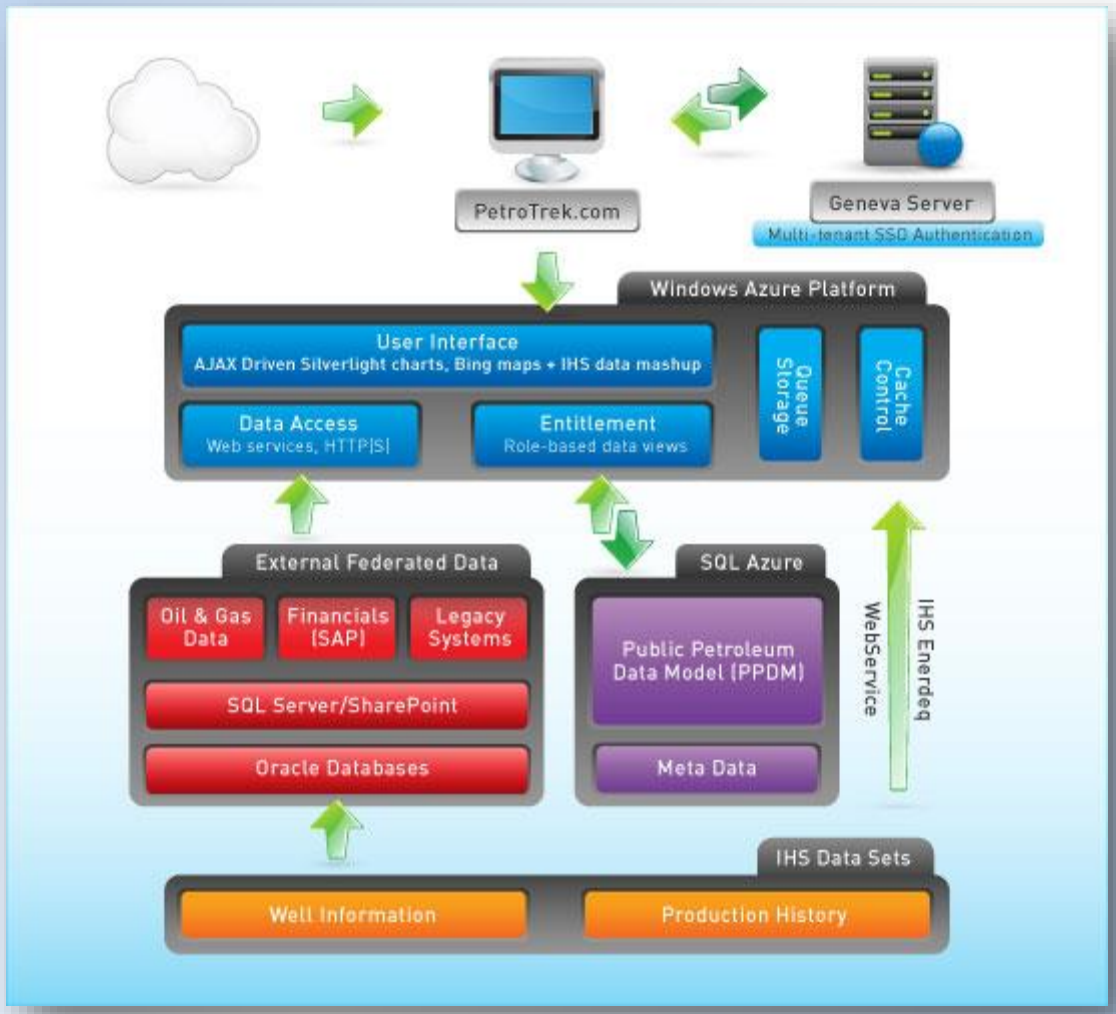
… and dynamic infrastructure has to be available via Internet
http://en.wikipedia.org/wiki/Dynamic_infrastructure

Cloud-based architecture is simple



Then our app architecture is as simple as we used to see

Example with Azure platform



Q & A

- What's wrong with the cloud?
  - Well, it is the current and future trend of software development. The cloud has changed medicine, real estate, small business, enterprise IT and a whole lot more. Engaging businesses will stick to the cloud. More here.
- What's after the cloud?
  - I've answered so far. It's your turn now (tip in references) ☺.

As a summary

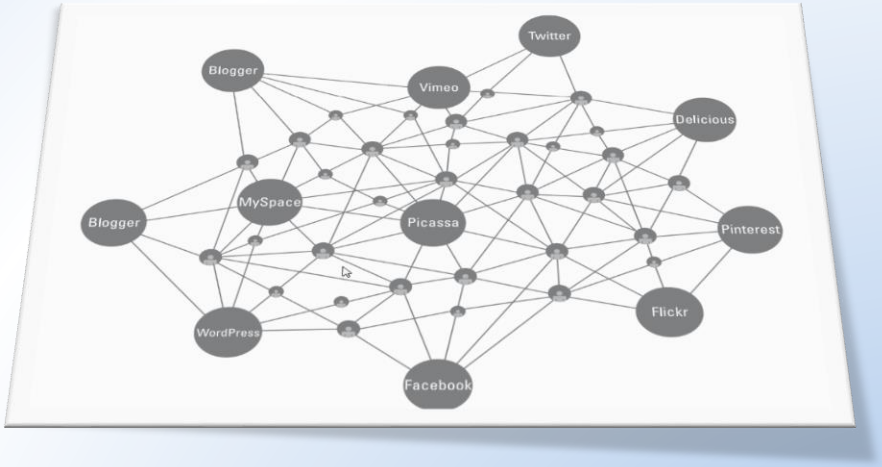
- **Layered Architecture** solves biz requirements
- **DDD** focuses on the Domain layer, thus increases biz semantics
- **Clean Architecture** adds more technical value by making the app core technical-independent
- **SOA** maximizes biz values by delivering functionality in industry-compliant units
- **Cloud-based Architecture** maximizes technical values all over the Internet to deliver SOA services
- ... (You know what **the next** will do ☺?)

Q & A

References

- **Design views:**
  - http://www.1keydata.com/datawarehousing/data-modeling-levels.html
  - http://it.toolbox.com/blogs/bridging-gaps/systems-architecture-fundamentals-conceptual-logical-physical-designs-11352
- **Layered architecture:**
  - http://archfirst.org/books/layered-architecture
  - http://en.wikipedia.org/wiki/Multilayered_architecture
- **DDD (Domain-Driven Design):**
  - http://en.wikipedia.org/wiki/Domain-driven_design
  - http://archfirst.org/books/domain-driven-design
  - http://dddsample.sourceforge.net/architecture.html
  - http://www.infoq.com/articles/ddd-in-practice
  - http://blogs.msdn.com/b/marblogging/archive/2011/05/23/domain-drive-design-n-layered-net-4-0-architecture-guide.aspx
- **Clean architecture:**
  - http://alistair.cockburn.us/Hexagonal+architecture
  - http://blog.8thlight.com/uncle-bob/2012/08/13/the-clean-architecture.html
- **Enterprise Architect:**
  - http://theenterprisingarchitect.blogspot.com/
- **SOA:**
  - http://en.wikipedia.org/wiki/Service-oriented_architecture
  - http://searchsoa.techtarget.com/definition/service-oriented-architecture
  - **SOA Design Principles for Dummies, IBM Limited Edition**, *Claus T.Jensen,* ***A Wiley Brand***
- **Cloud computing:**
  - http://apprenda.com/blog/answer-this-what-is-a-cloud-application-to-you/
  - http://searchcloudcomputing.techtarget.com/definition/Software-as-a-Service
  - http://searchcloudcomputing.techtarget.com/definition/Platform-as-a-Service-PaaS
  - http://searchcloudcomputing.techtarget.com/definition/Infrastructure-as-a-Service-IaaS
  - http://en.wikipedia.org/wiki/Dynamic_infrastructure
  - http://www.onlinetech.com/resources/wiki/data-centers/after-the-cloud-what-next-mobile-technology-in-data-centers

THANKS FOR COMING. SEE YA!



For more, please visit: http://phuonglamcs.com/relax/presentations/