

# Interoperability: Examples from MSC's Architectural Directions



# Architecture

- Architecture is never fully described by a single drawing or representation
- There are always multiple Aspects of an Architecture which needs to be described
- Take a house for example:  
*Plat, Layout Drawing, Framing Diagram, ...*
- The same is true with Systems Architecture



# Aspects of Systems Architecture

- Business Architecture
- Application Architecture
- Application Integration Architecture
- Service (Function) Architecture
- Execution Architecture
- Administrative Architecture
- Physical Architecture



# Business Architecture

- **Goal: Assure System Supports Business Functions Efficiently; the **Constitution****
  - **Structure of the Business Process**
    - Tasks with Information Consumption/Production
  - **Business Task to Application ID/Mapping**
    - Identify Major and “*Mini*” Apps needed for task
    - Data Consumption/Production
  - **Data Sharing**
    - Among Business Units, Tasks and External Enterprises (Customers/Partners/Vendors)



# Application Architecture

- **Strategy and structures for crafting point-of-use applications.**
- **Goals: Rapid Development of Production Quality Applications**
  - **Re-Use and Sharing of Production Quality Functions**
  - **Prepackaged, Reusable, GUI Widgets**

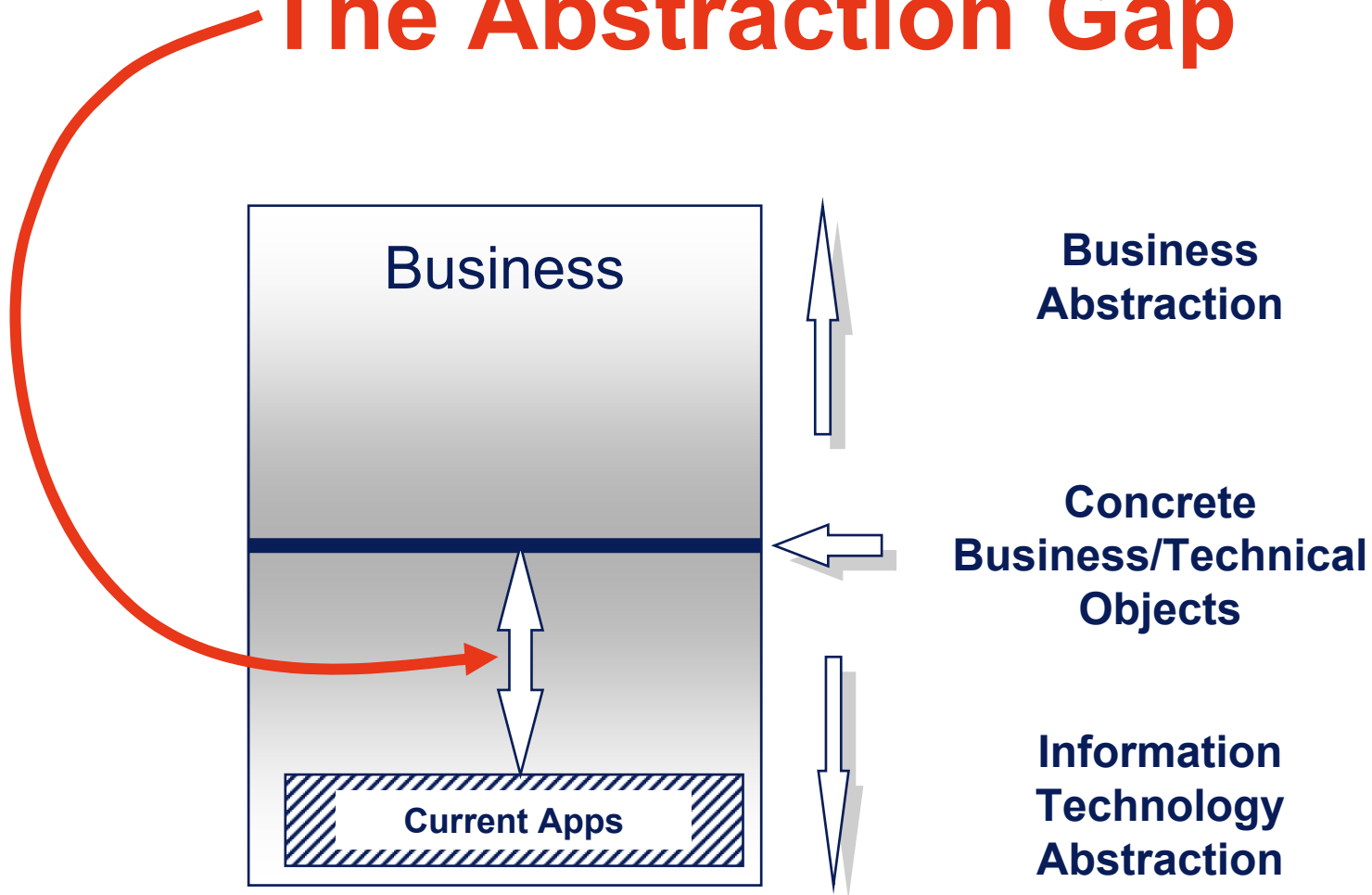


# Example Architectural Goals of an Enterprise Materials Database

- **Business**
  - **Provide Uniform Material Reference Across the Business Process**
- **Application Integration**
  - **Provide Access to Bonafide Material Properties Consistently across all Engineering Applications**



# The Abstraction Gap



# Bad Effects of Abstraction Gap

- **Business Process is Highly Dependent on Particular Applications**
- **Small Changes in the Business Process may Require Vast Changes in the Application that may be Expensive or Impossible**
- **The Cost of Changes in the Infrastructure are **not Proportional** to the Degree of Change in the Business Process**
- **The Application Holds the Business Process Hostage!**

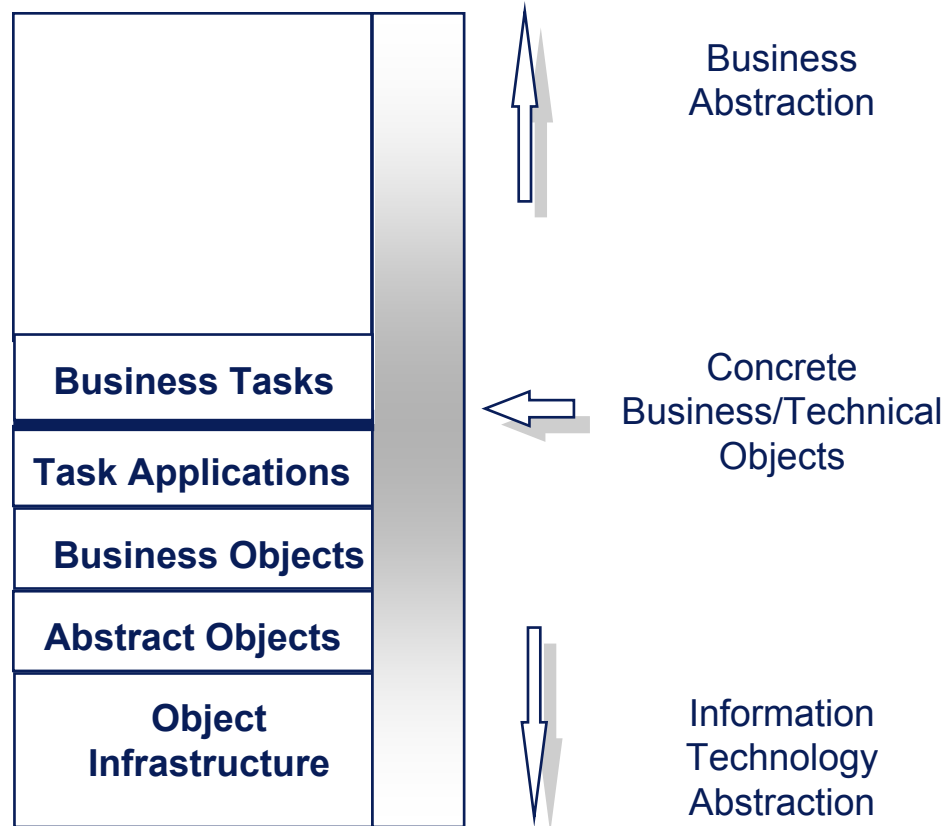


# Spanning the Abstraction Gap

- Object Technology permits the definition of large granularity objects with complex methods.
- Objects can be defined with a one to one correspondence with the business objects.
- Application programming can be done in terms of the business objects.
- Application programming does not require tedious, detailed, field-level programming.
- Reprogramming the infrastructure is proportional in effort to Re-Engineering the Business Process.



# Spanning the Abstraction Gap



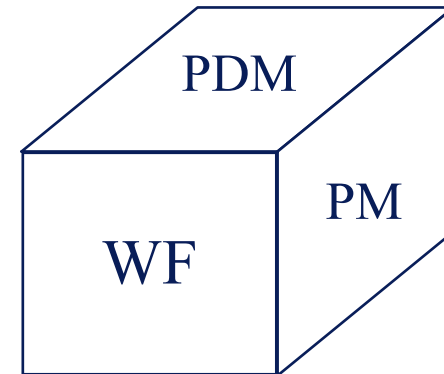
# Service (Functional) Architecture

- **Infrastructure Services for use by Applications**
- **Move the work out of the applications to the Services**
- **Applications no longer to contain unshareable business rules and algorithms.**
- **Applications responsible for presenting information in the context of the specific business task.**

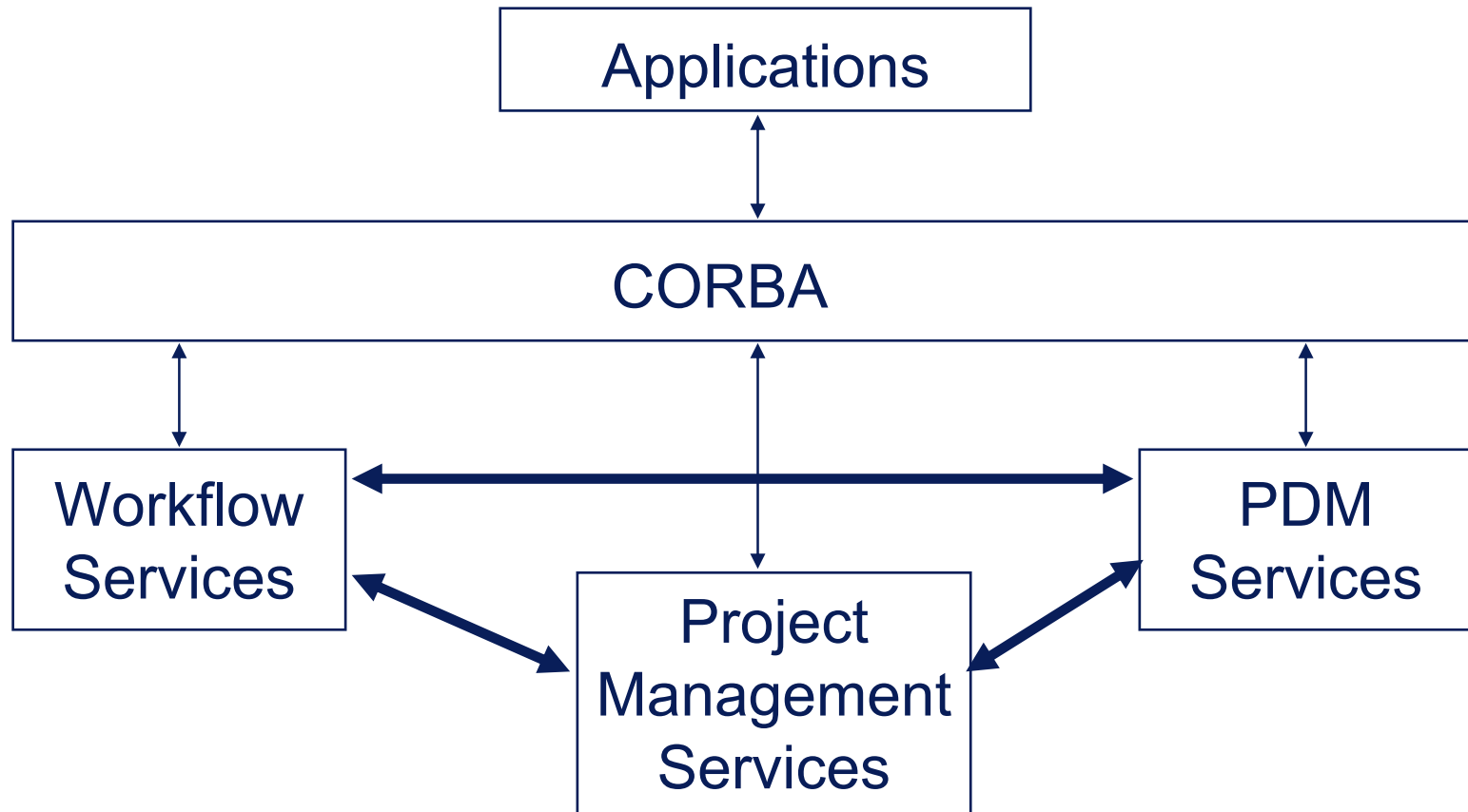


# Example of Service Architecture WF/PDM/PM Integration

- Vehicle for Collaboration with NCMS Project Endeavor (concept funding)
- Integration of
  - Workflow
  - Product Data Management
  - Project Management
- Integrated Object Views
- Task-Oriented Data Acquisition



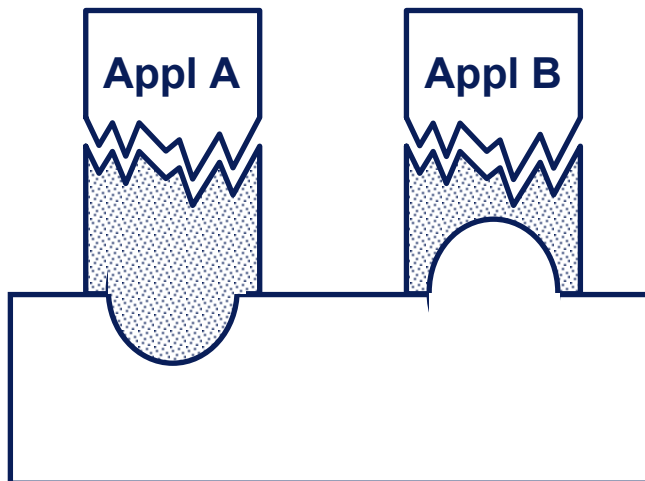
# Example of Service Architecture Integration via Infrastructure



# Application Integration Architecture

- **Goals: Facilitate the rapid assimilation of standalone applications into a cooperative interoperable system.**

- **Techniques for “standardizing” the development of “glue code**



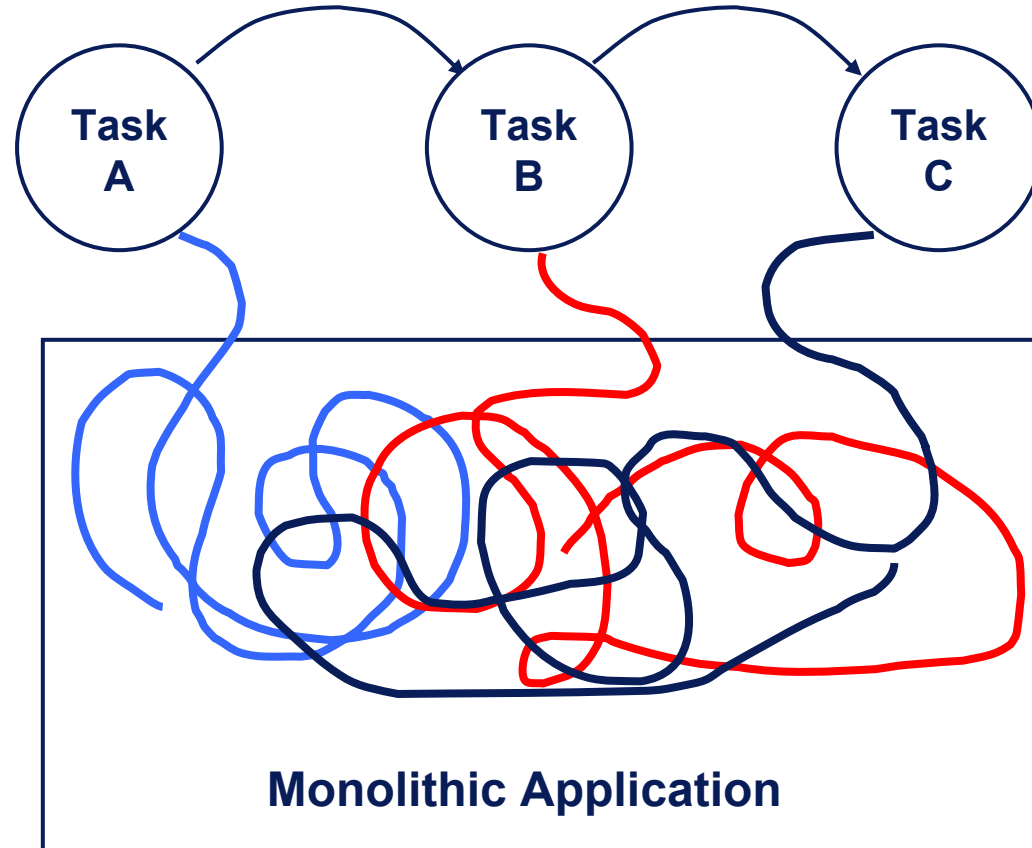
# The Monolithic Legacy

## using the Example of PDM (Product Data Management)

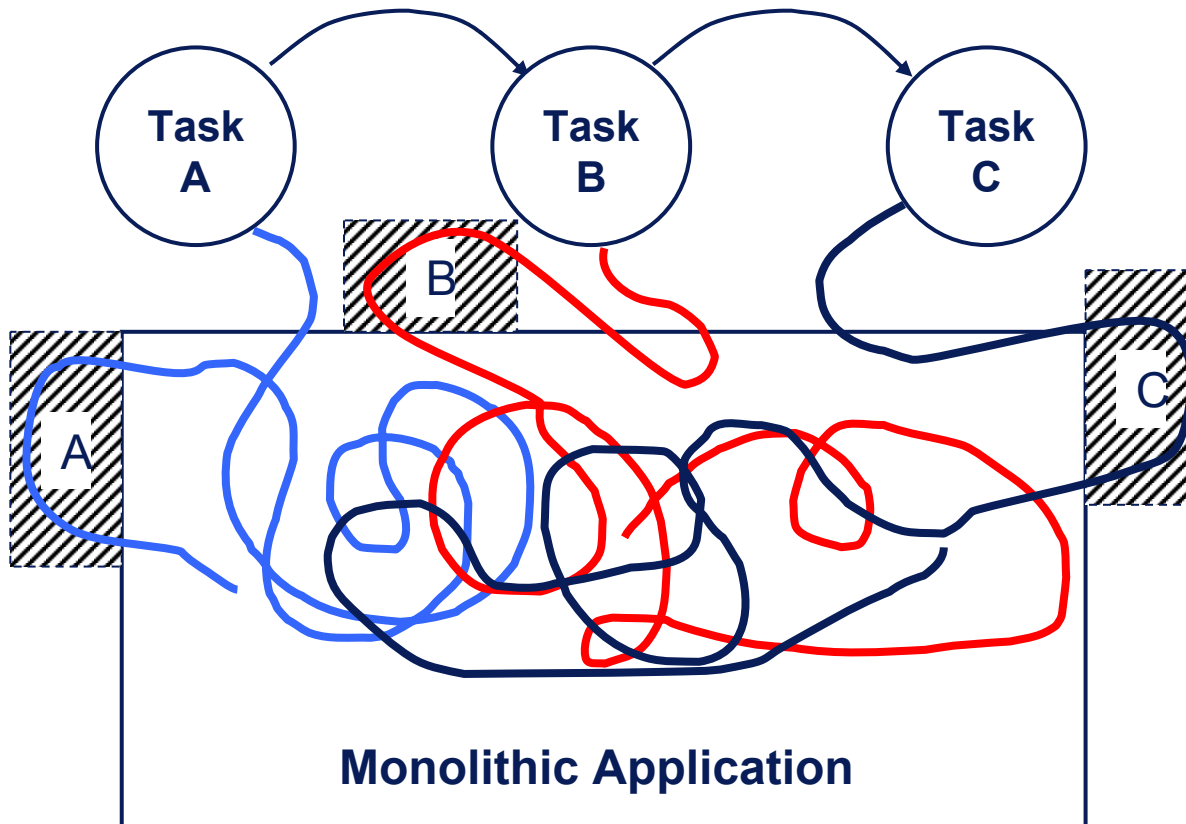
- **Artificial Boundaries**
  - **What is in a Product Data Management System?**
  - **What is not in a PDM?**
  - **Does a given function belong in PDM, Workflow, or ERP?  
Does it really matter?**
- **No Engineer wants to be an expert in PDM**
- **Need to make the PDM services oriented toward the Business, and available to all applications**
- **Need to make PDM happen transparently, as a side-effect of normal business (design, analysis,...)**



# Integration via Monolithic Applications

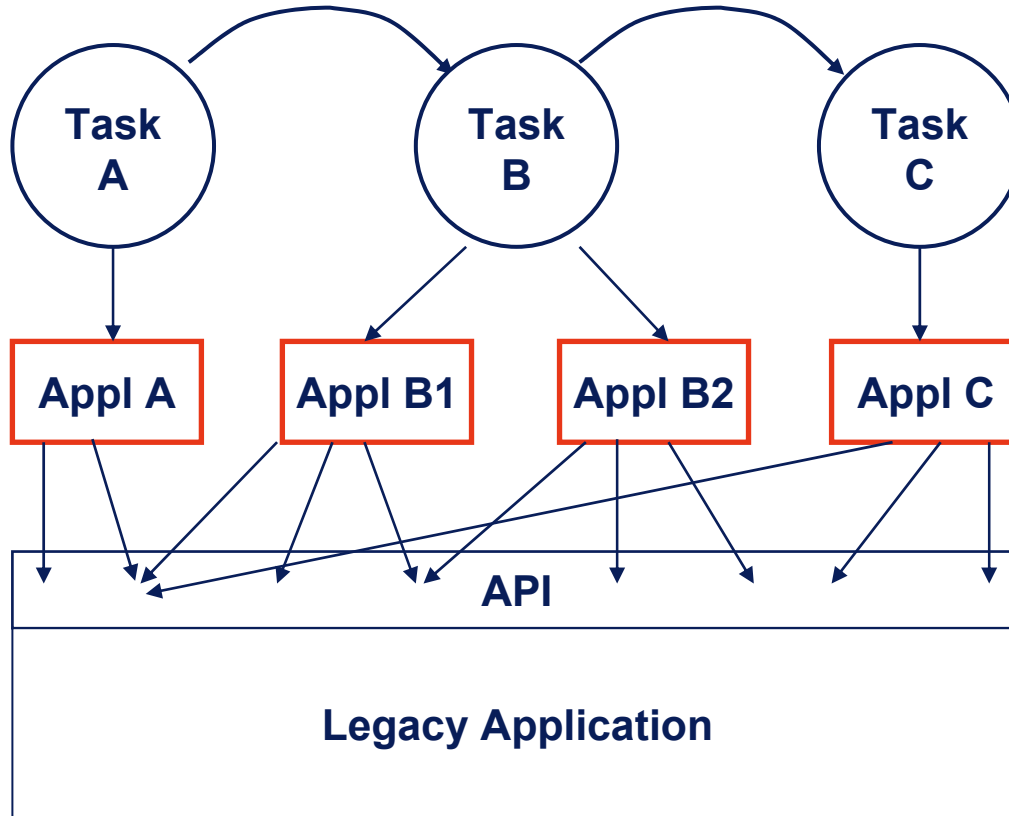


# Business Consequences of Monolithic Applications

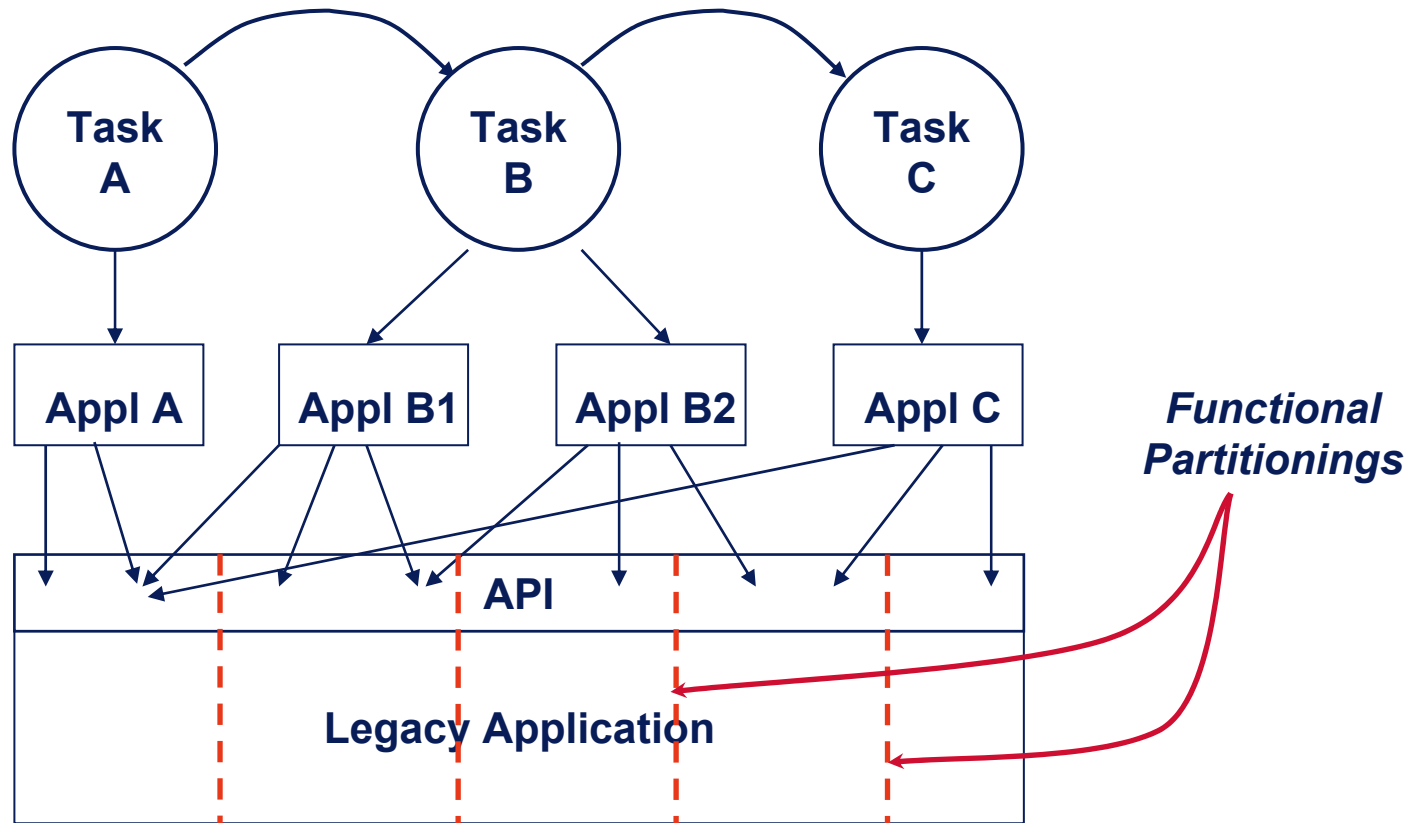


*Small Changes in Business Process can Necessitate need for Unanticipated Functionality*

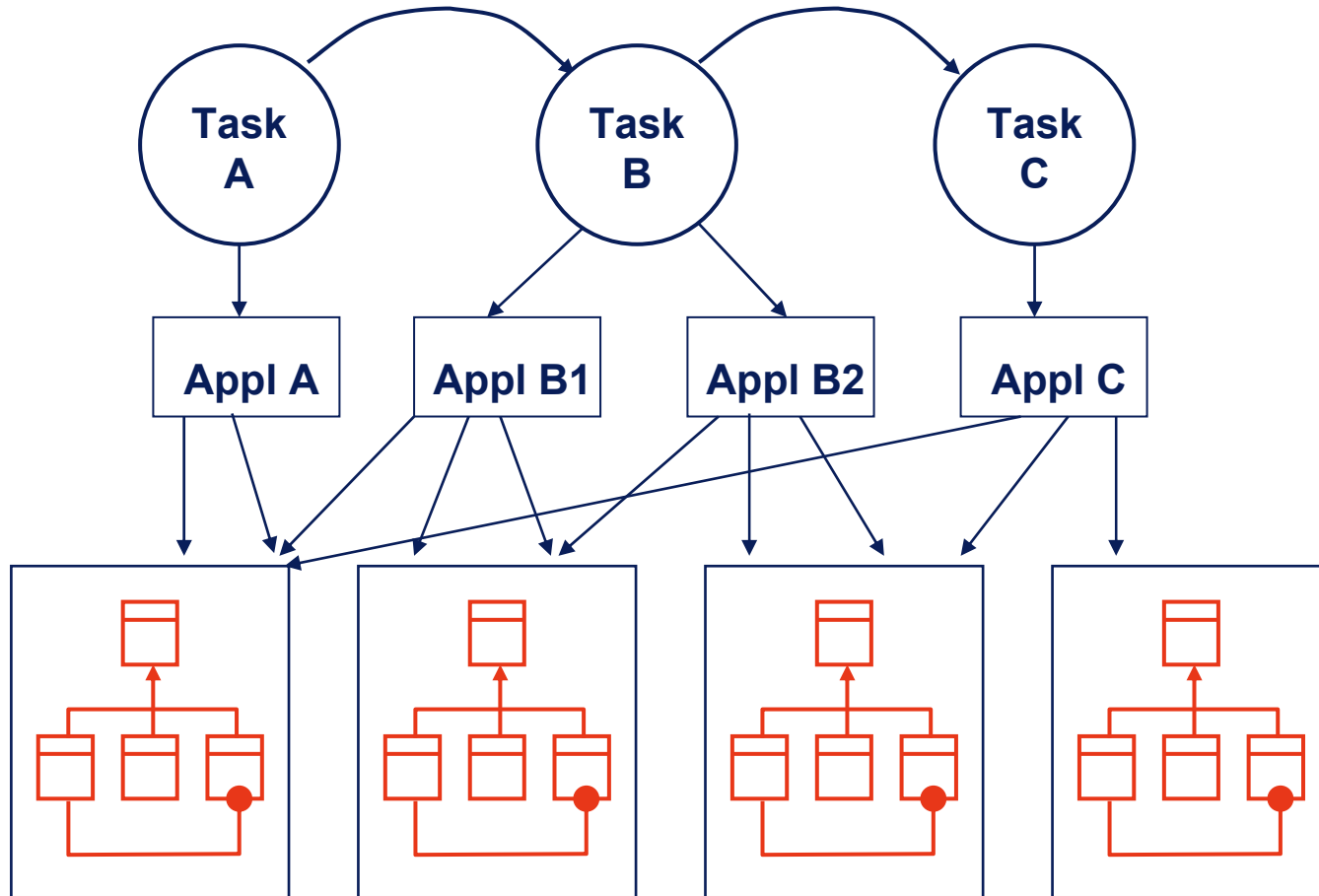
# Shift to Small Task Oriented Applications



# Shift to Business Oriented Infrastructure



# Principles of Functional Partitioning: Methods of Object Models



# OMG PDM Enablers

- **Product Data Management**
  - **What is It?**
  - **What's it Contain?**
- **Enablers**
  - **Part Structure**
  - **Document Management**
  - **Effectivity**
  - **Change Management**
  - **Etc...**



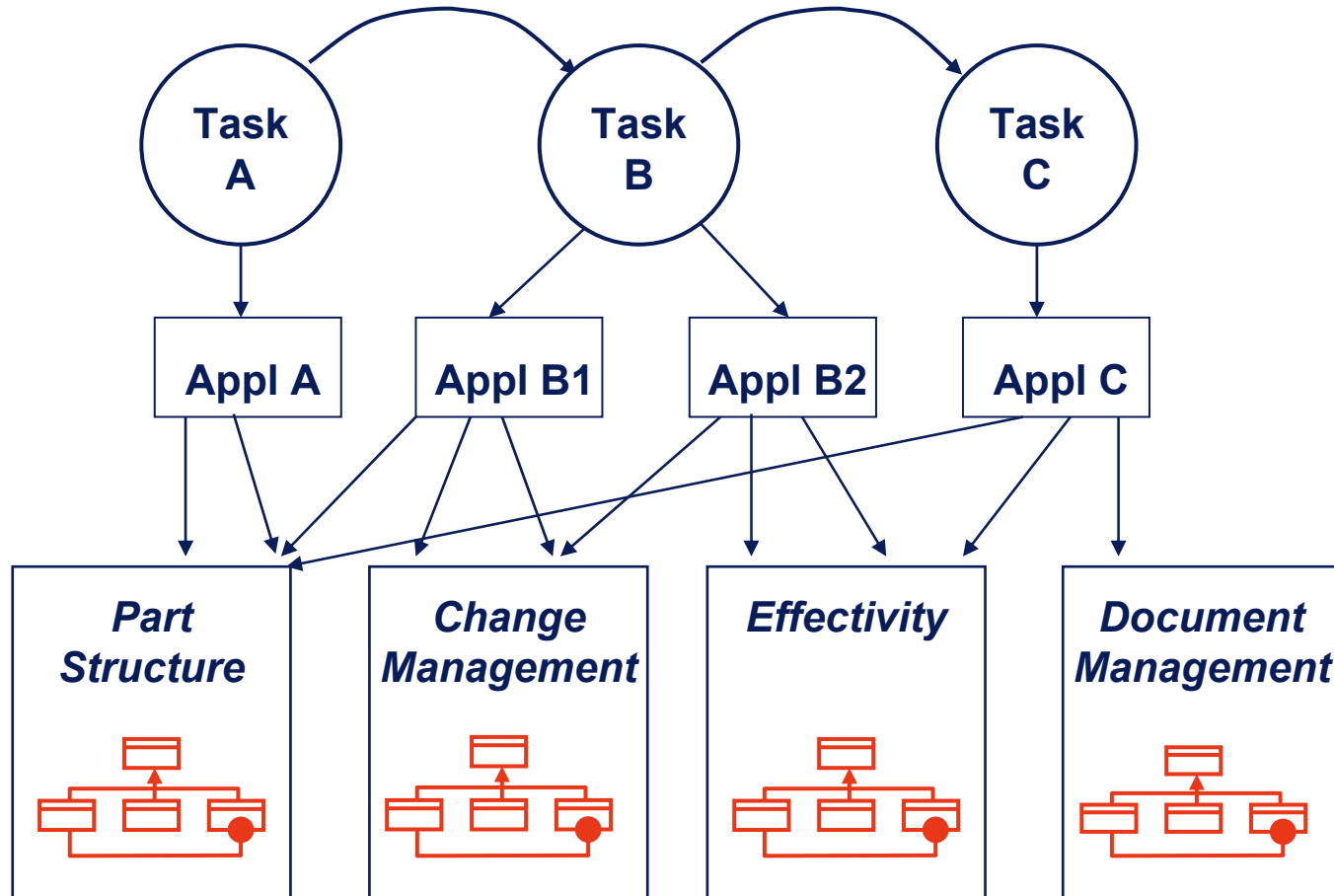
# Joint PDM Submission Team

- **MacNeal-Schwendler**
  - Independent Chair
  - Representing RRM
- **PDM Vendors**
  - Metaphase
  - IBM
  - Sherpa
  - Adra
  - Fujitsu
  - DEC
  - NIIP

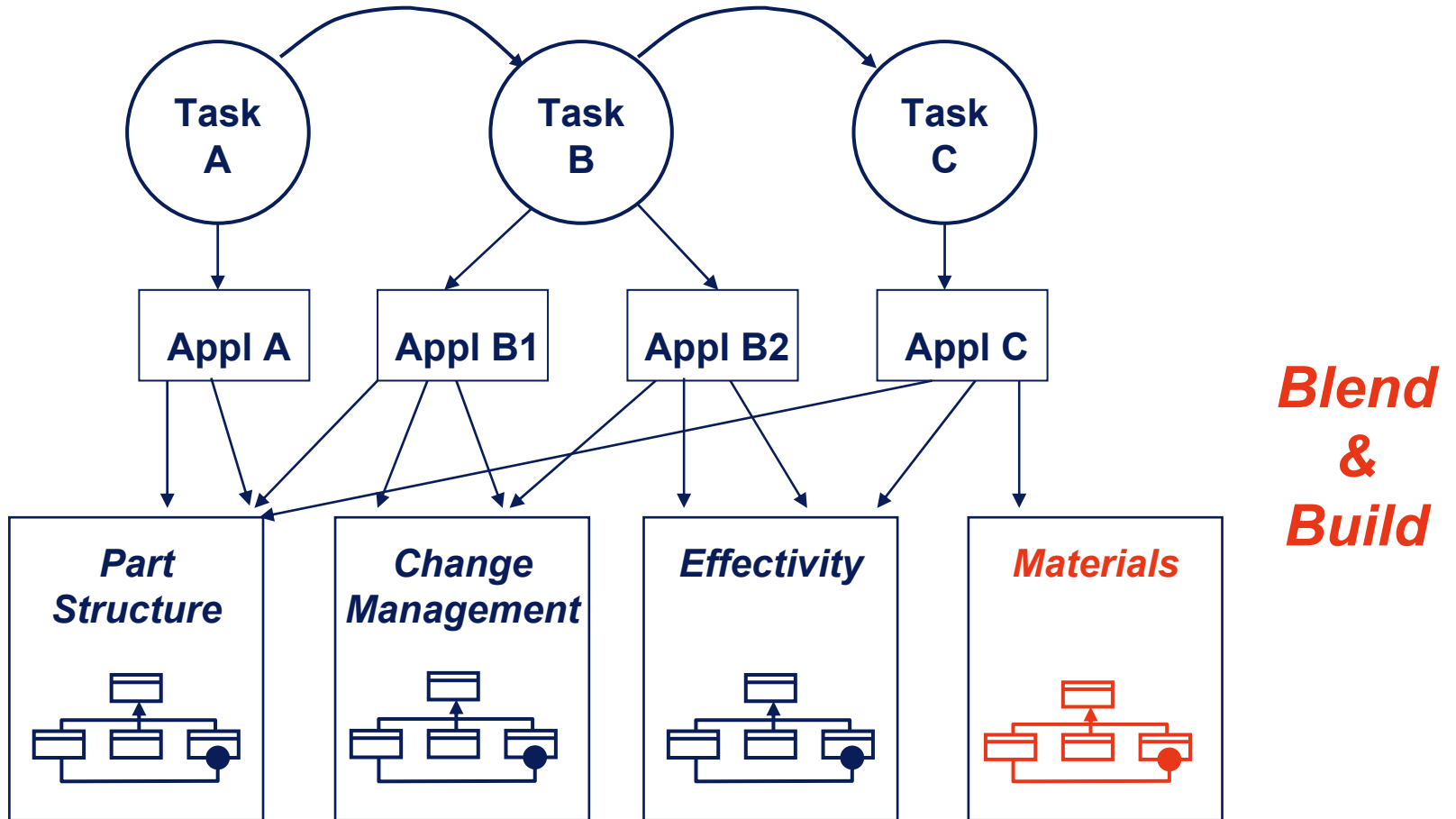
- **Goal:**
  - Provide Standard Service Interface to PDM Enablers
  - Implementable by all Participating Vendors
- **Approach:**
  - Define Object Model of Enablers and their Interdependencies
  - Derive IDL Interfaces from Object Model.



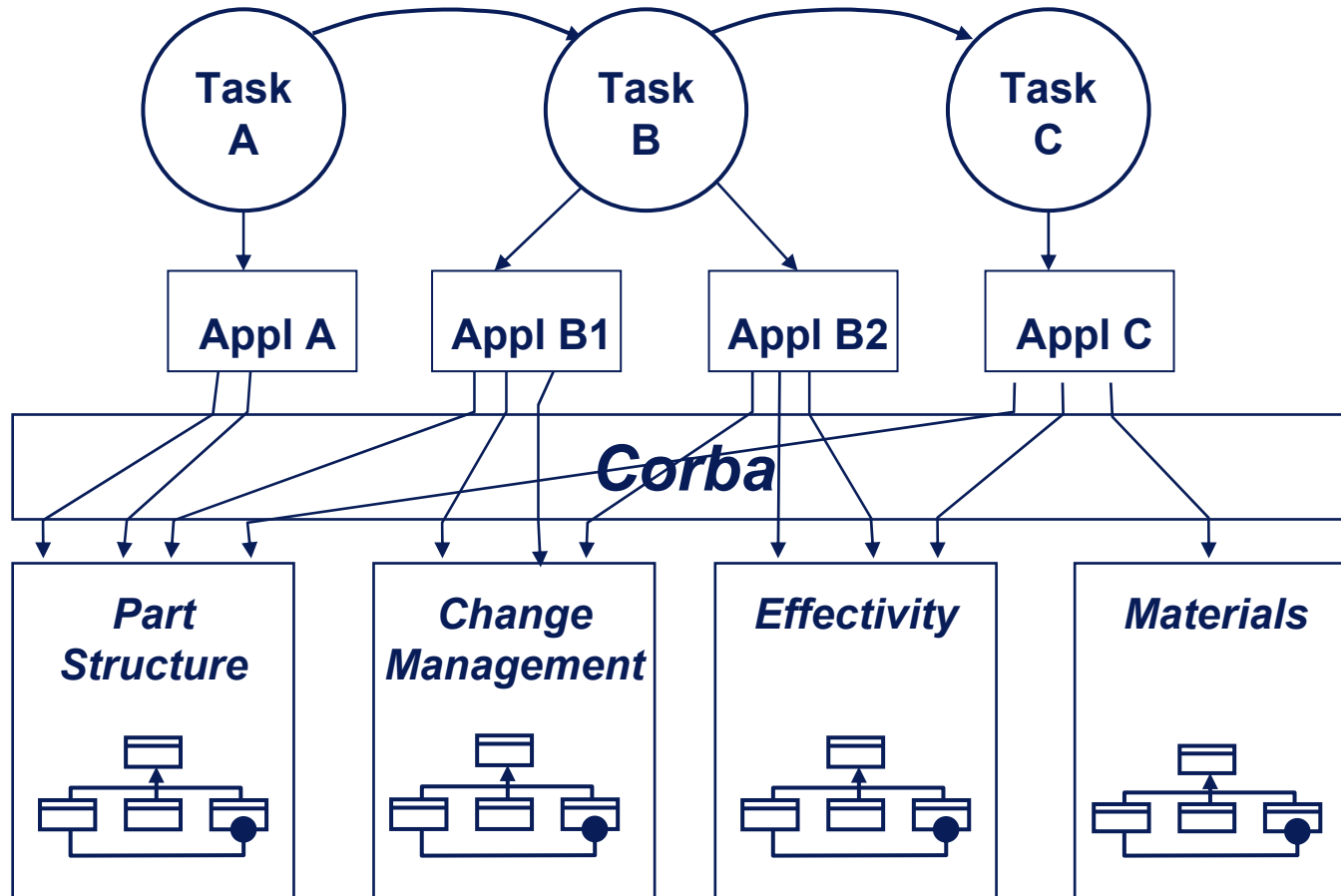
# The Case of PDM



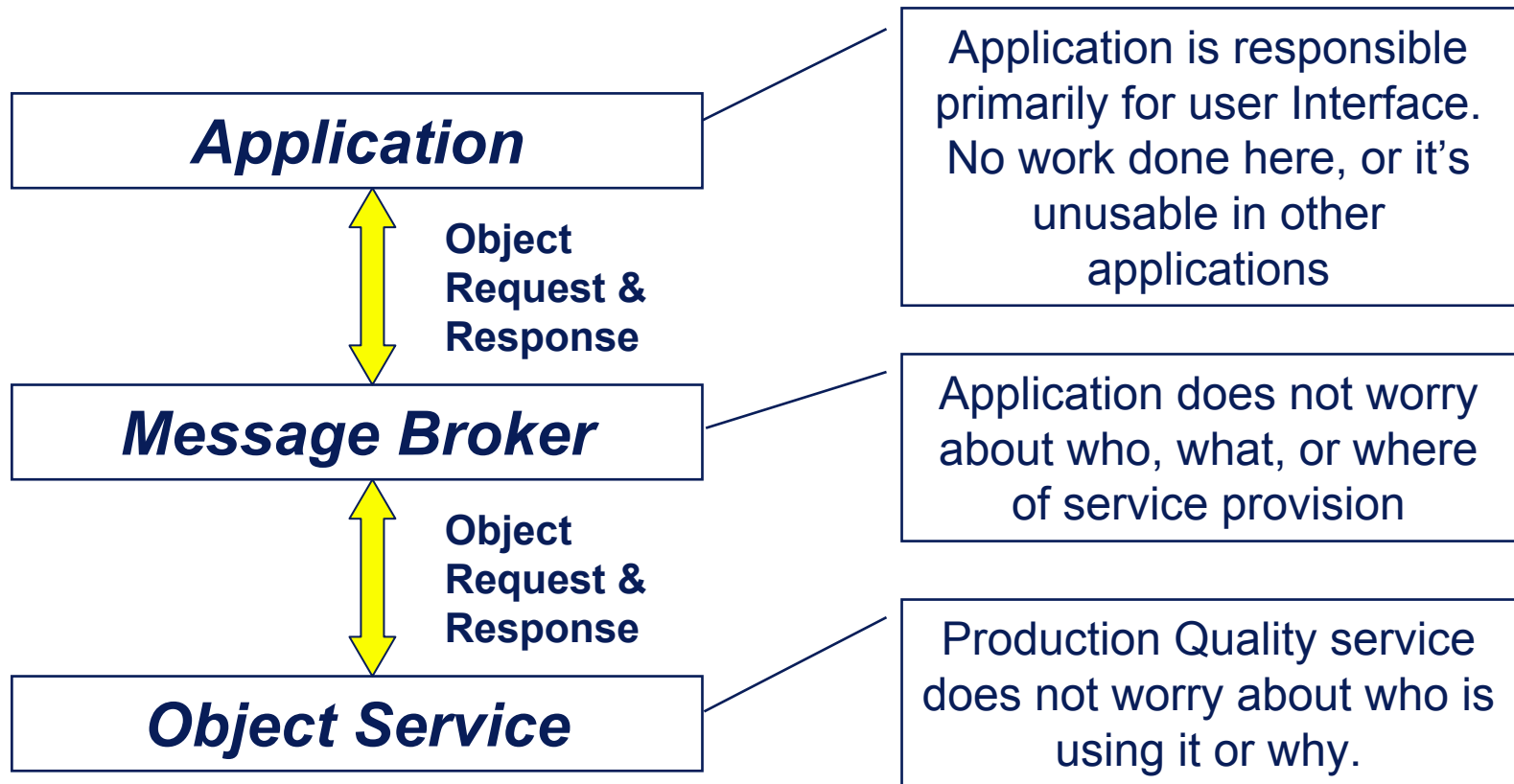
# The Case of Material Services



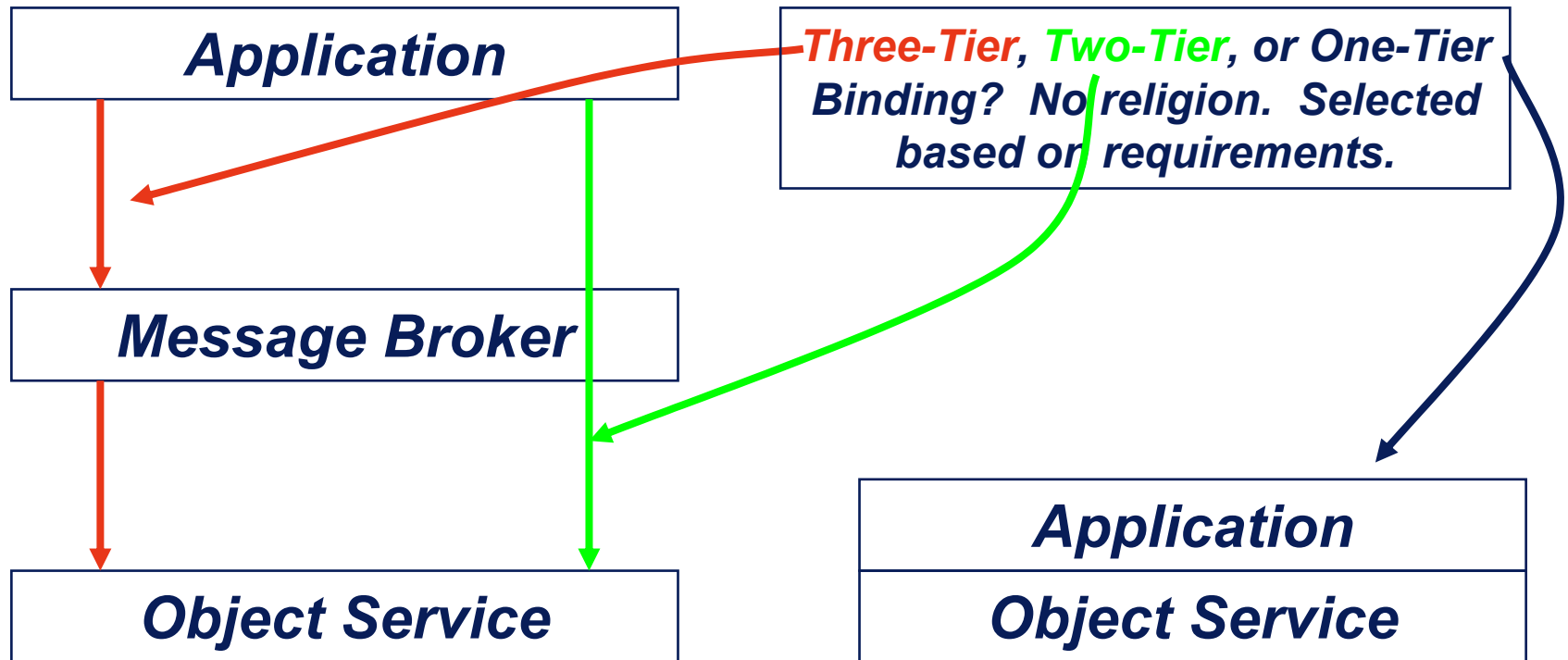
# Distributed Objects



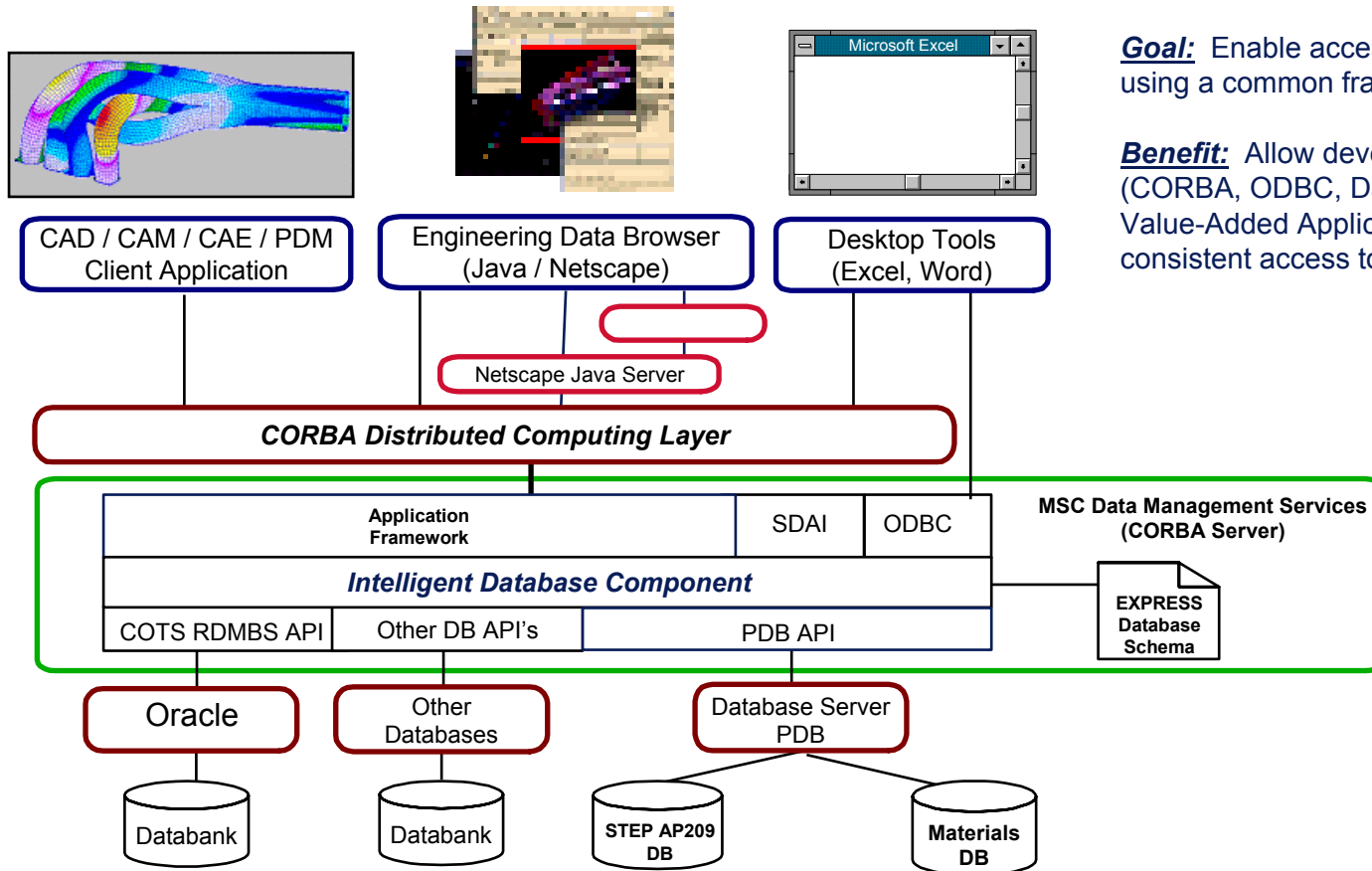
# Execution Architecture



# Execution Architecture



# MSC Data Management Architecture



**Goal:** Enable access to MSC databases using a common framework.

**Benefit:** Allow development of interfaces (CORBA, ODBC, Data Browsing Tools, Value-Added Applications) which provide consistent access to data.



# Enterprise Evolution

- Revolution is often advocated, but seldom practical in a large company.
- Legacy systems need to be accommodated while transitions to the future takes place.
- Technology and Business Processes evolve continuously...
- We need to prepare more for the journey than the destination. We won't be at any destination long, but will be on the journey forever.
- Blend & Build

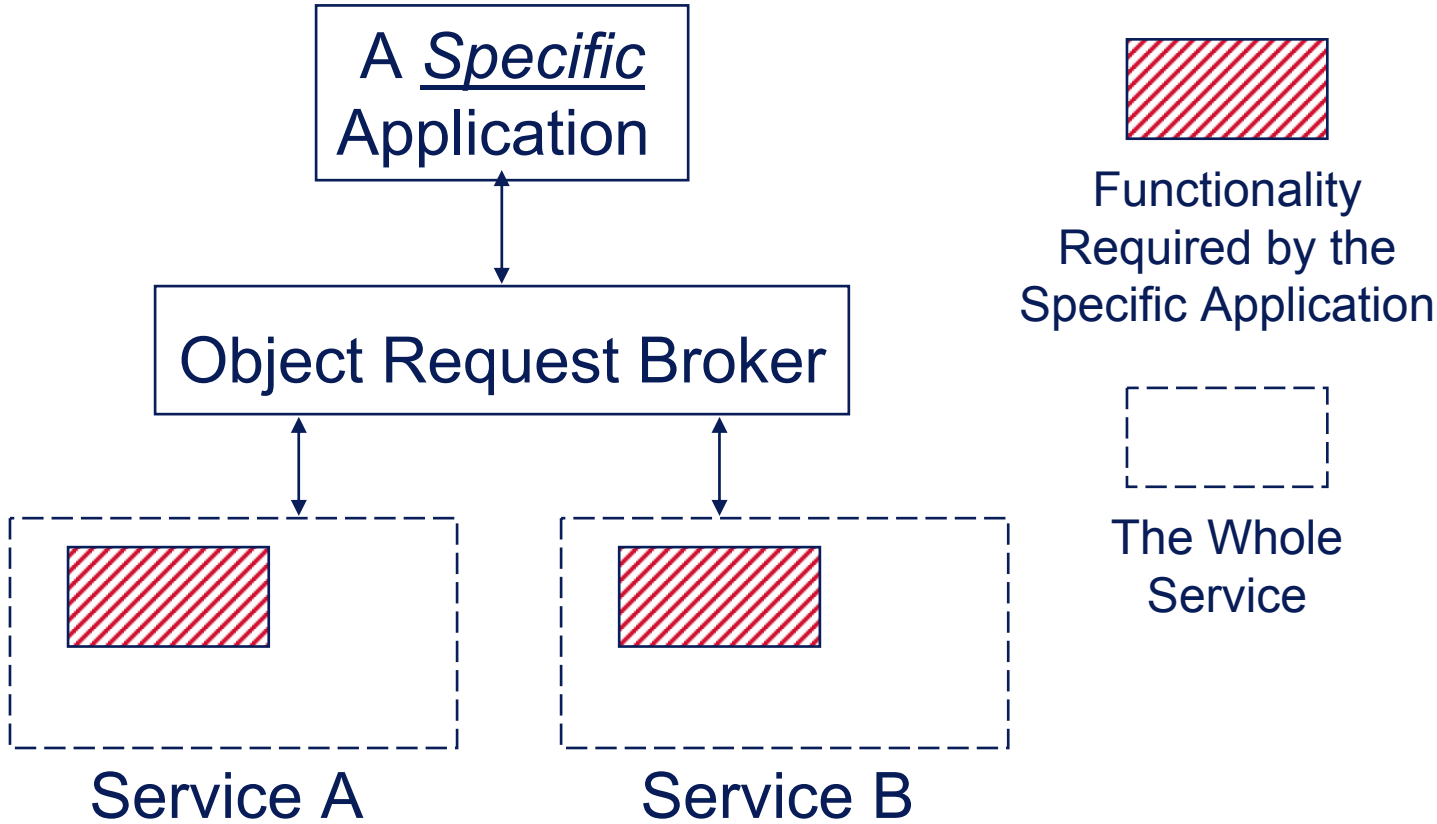


# Blend & Build

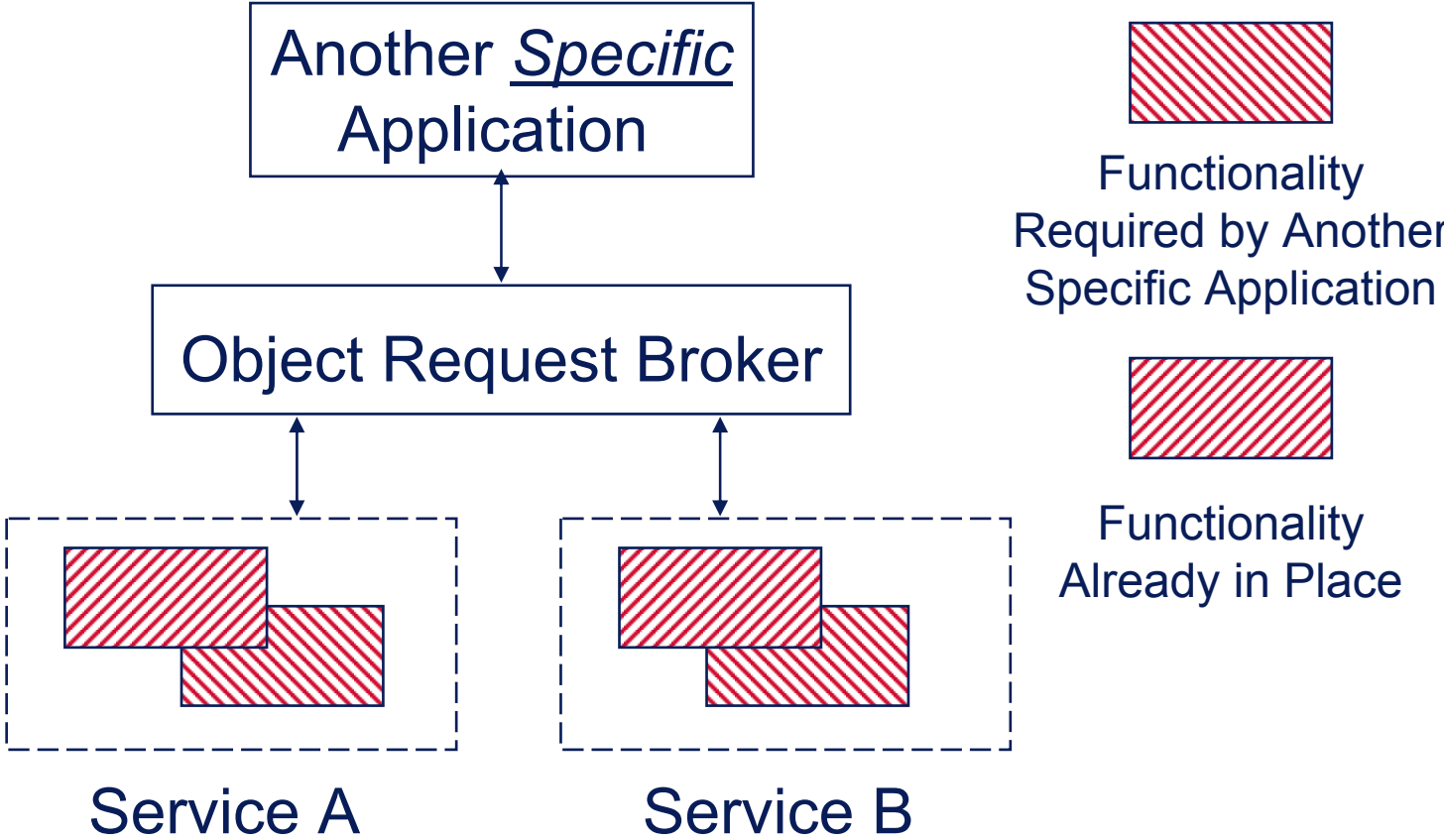
- We need to implement in small digestible chunks.
- Task oriented applications
- Integration through the infrastructure
- Incremental development of the infrastructure
- Reusability of existing infrastructure
- Evolution, not Revolution



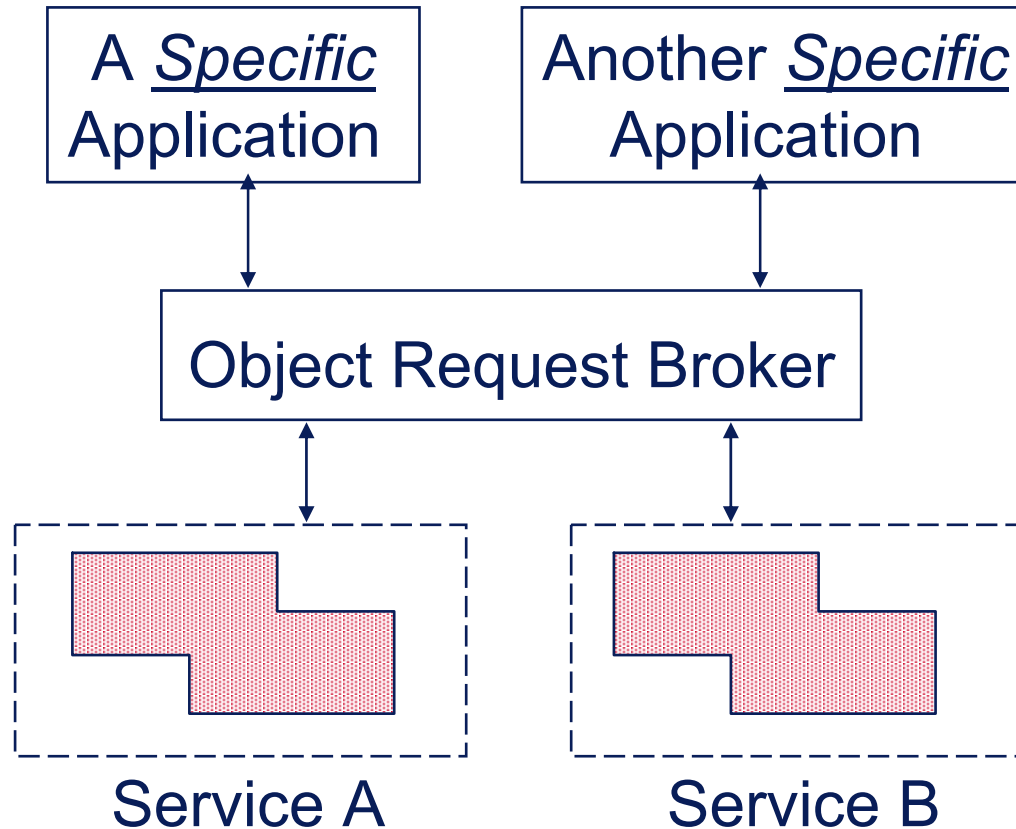
# Incremental Infrastructure - 1



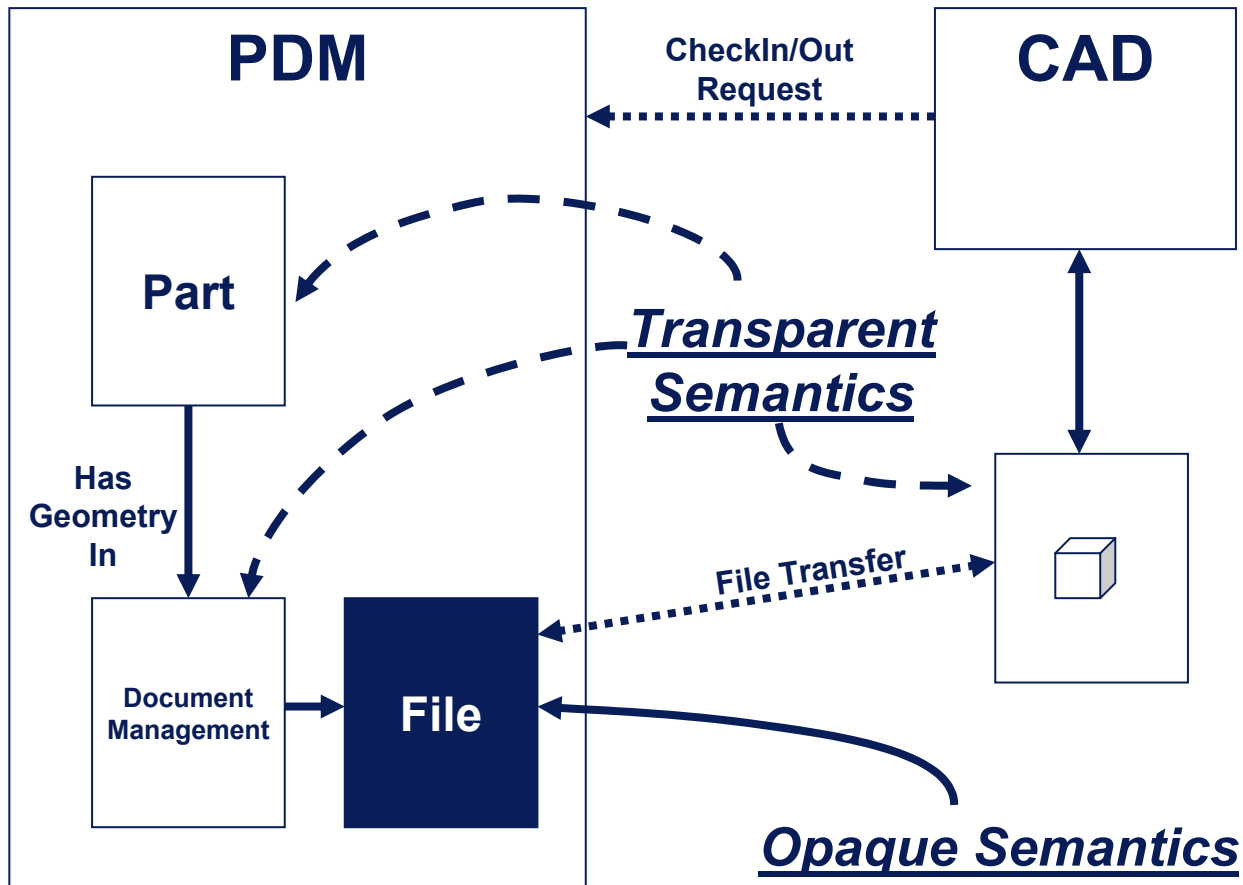
# Incremental Infrastructure - 2



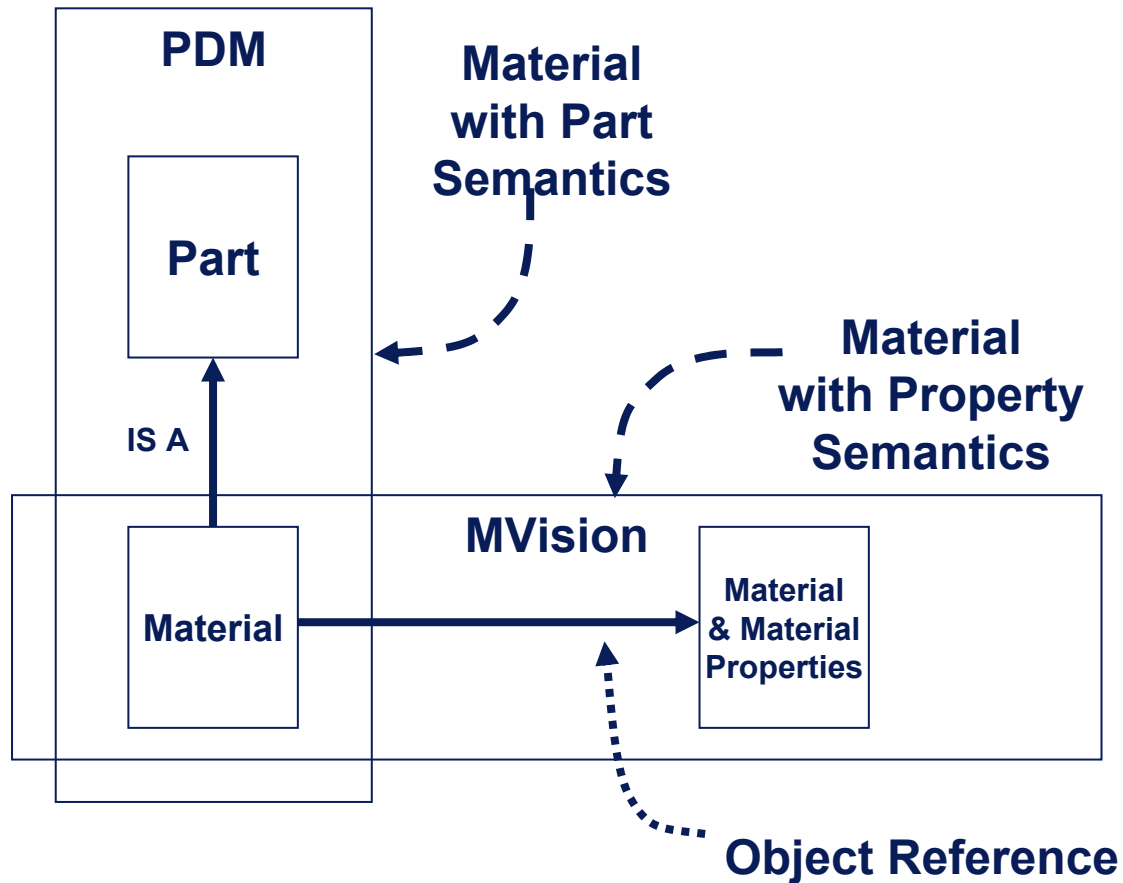
# Blend & Build



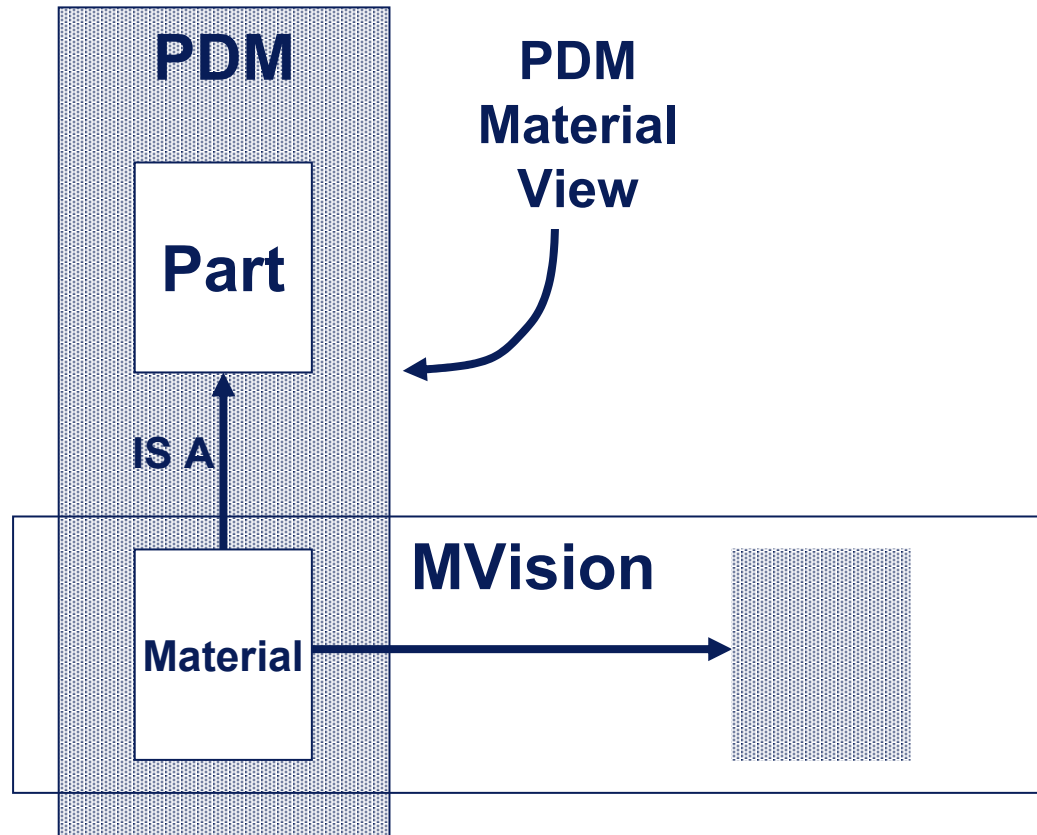
# Traditional PDM “Integration”



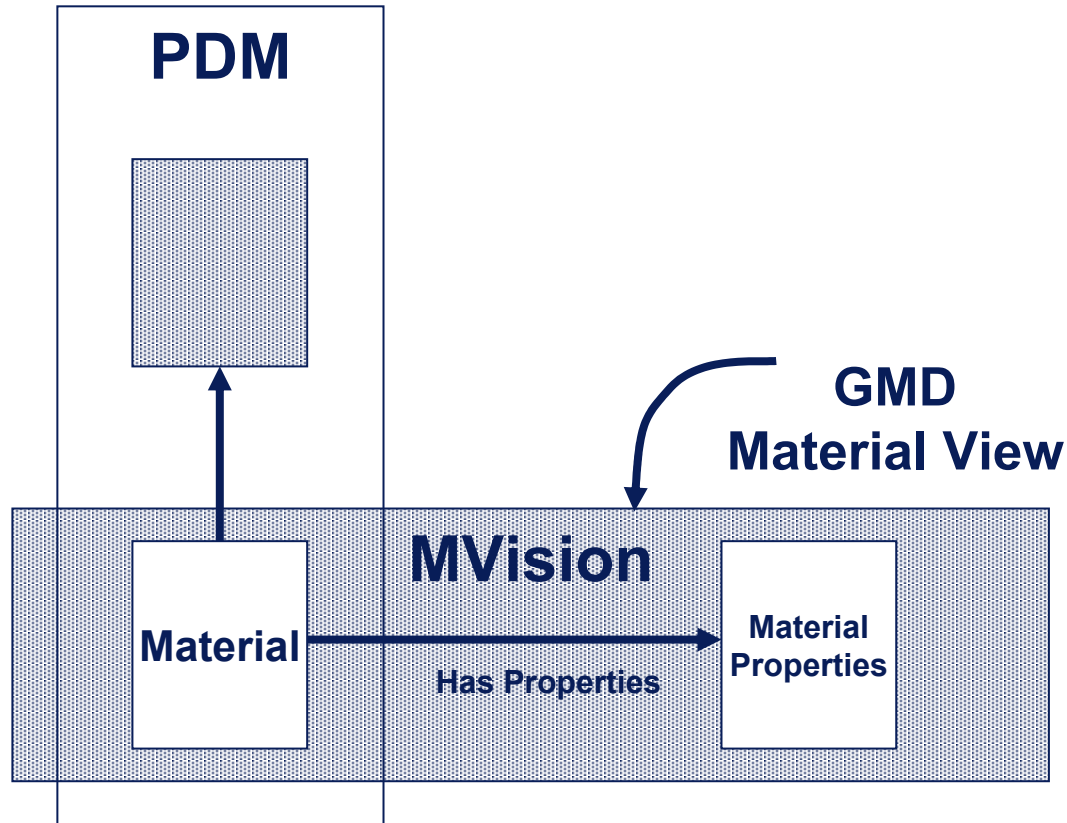
# Semantic PDM Integration



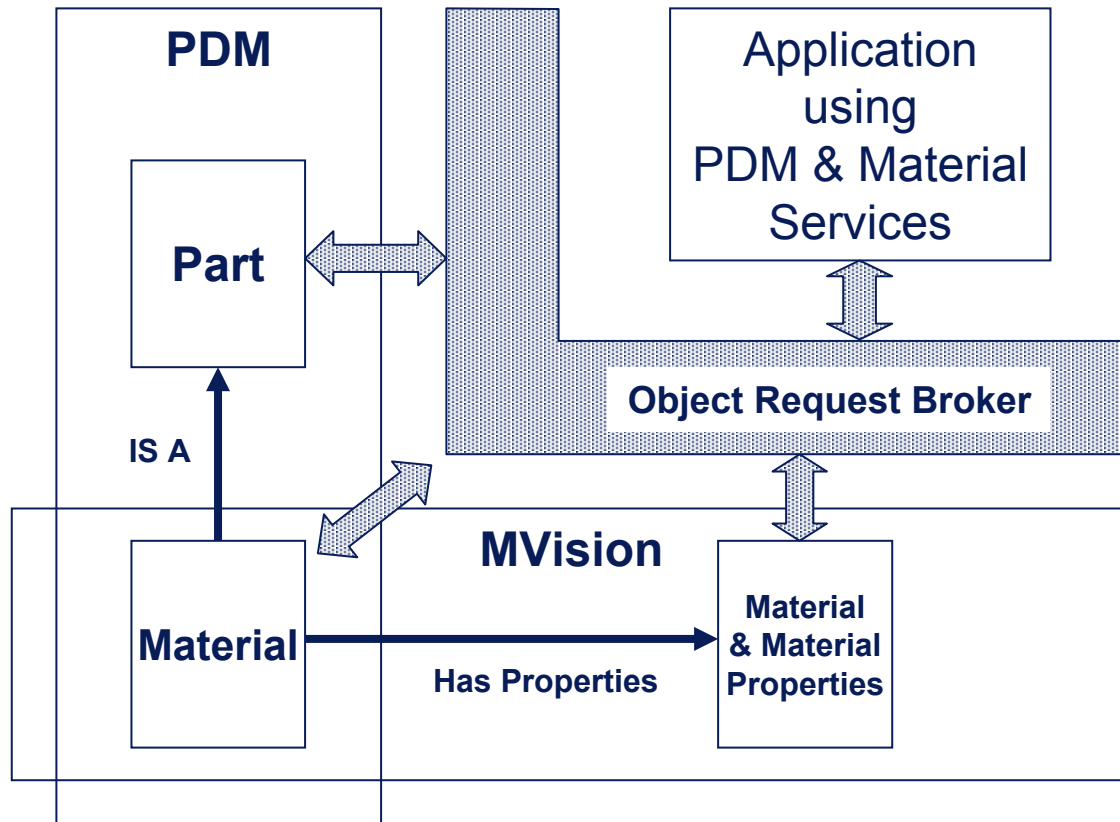
# Legacy PDM View



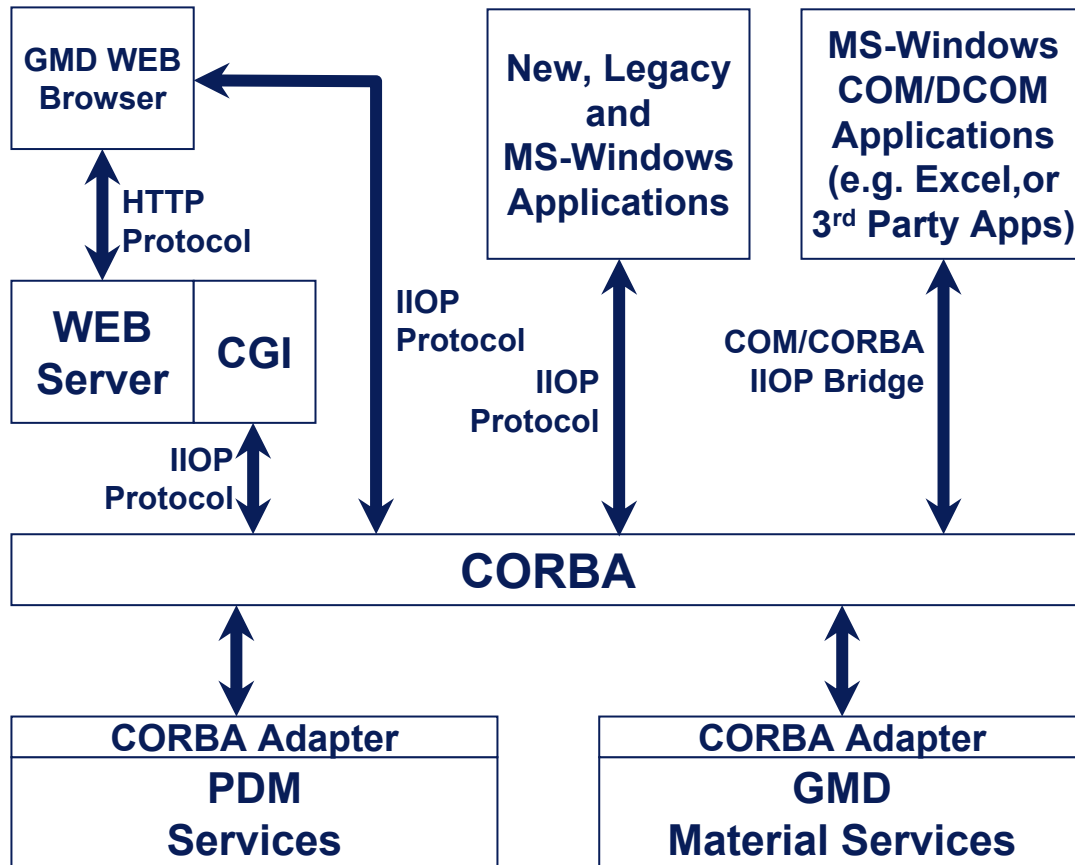
# Legacy MVision View



# Integrated GMD View



# Application Architecture



# Legacy Integration Architecture

