Enterprise Application Integration (EAI)

Chapter 4. Method-Level EAI

Definition

- Method-level EAI allows the enterprise to be integrated through the sharing of common business logic, or methods.
- "**Reuse**" is an Important concept in this context. By defining a common set of methods, we're able to reuse those methods among enterprise applications.

As a result, the need for redundant methods, and/or applications, is significantly reduced.

What's a Process?

- The goal of method-level EAI is to share processes
- For the purposes of EAI:
  - A **business process** is any rule or piece of logic that exists within the enterprise that has an effect on how information is processed

  for example, logic tied to a taxation system within the organization or a method within a C++ application that updates customer information

Scenarios

- How best to proceed?
- The first step should be to break the processes down into their scenarios, or types.
- For the purposes of EAI, these types are:
  - **Rules:** set of conditions that have been agreed upon
  - **Logic:** differs from rules in that logic is simply a sequence of instructions in a program
  - **Data:** is nothing more than sharing information between enterprise applications and computers or humans
  - **Objects:** objects that do something when an interface is invoked

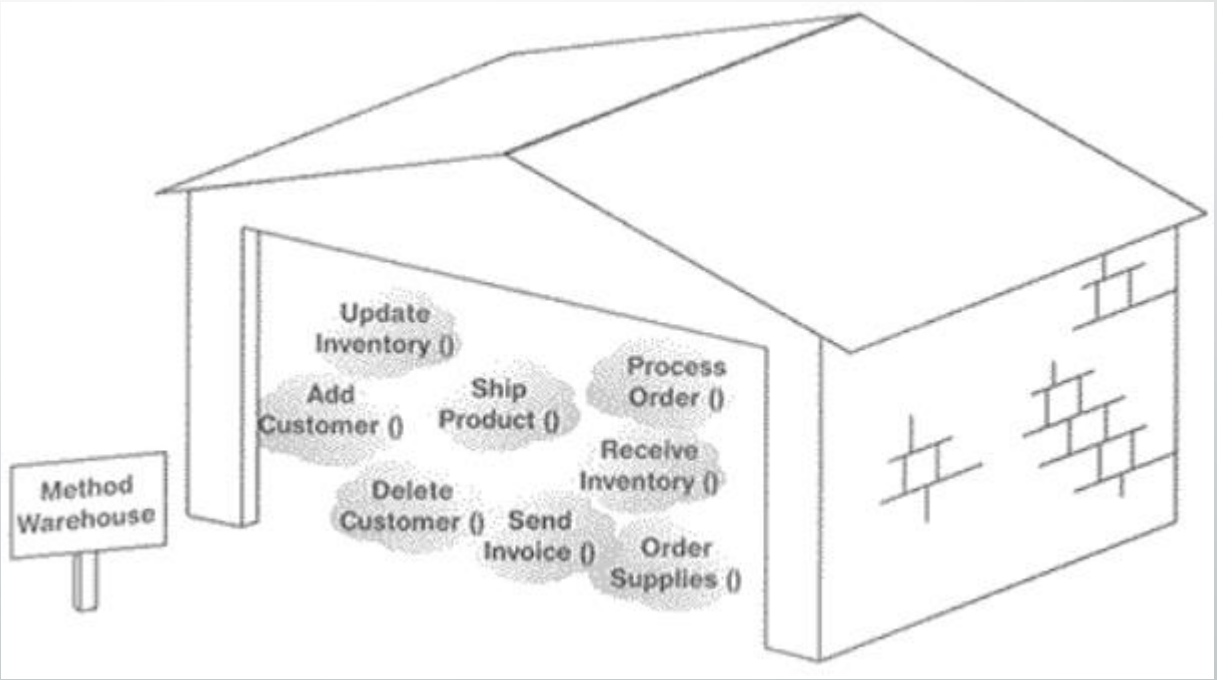
Rule Servers

- Sequential processing
- Selection
- Iteration

Method Warehousing

- The process of aggregating all business processes within an organization and then hosting them on a centralized server

If the tax law changes, it would be unnecessary to locate all applications within the enterprise that use tax law in order to incorporate the changes. The changes would automatically propagate to all link applications simply by a change in the logic, or rules, within the method warehouse.

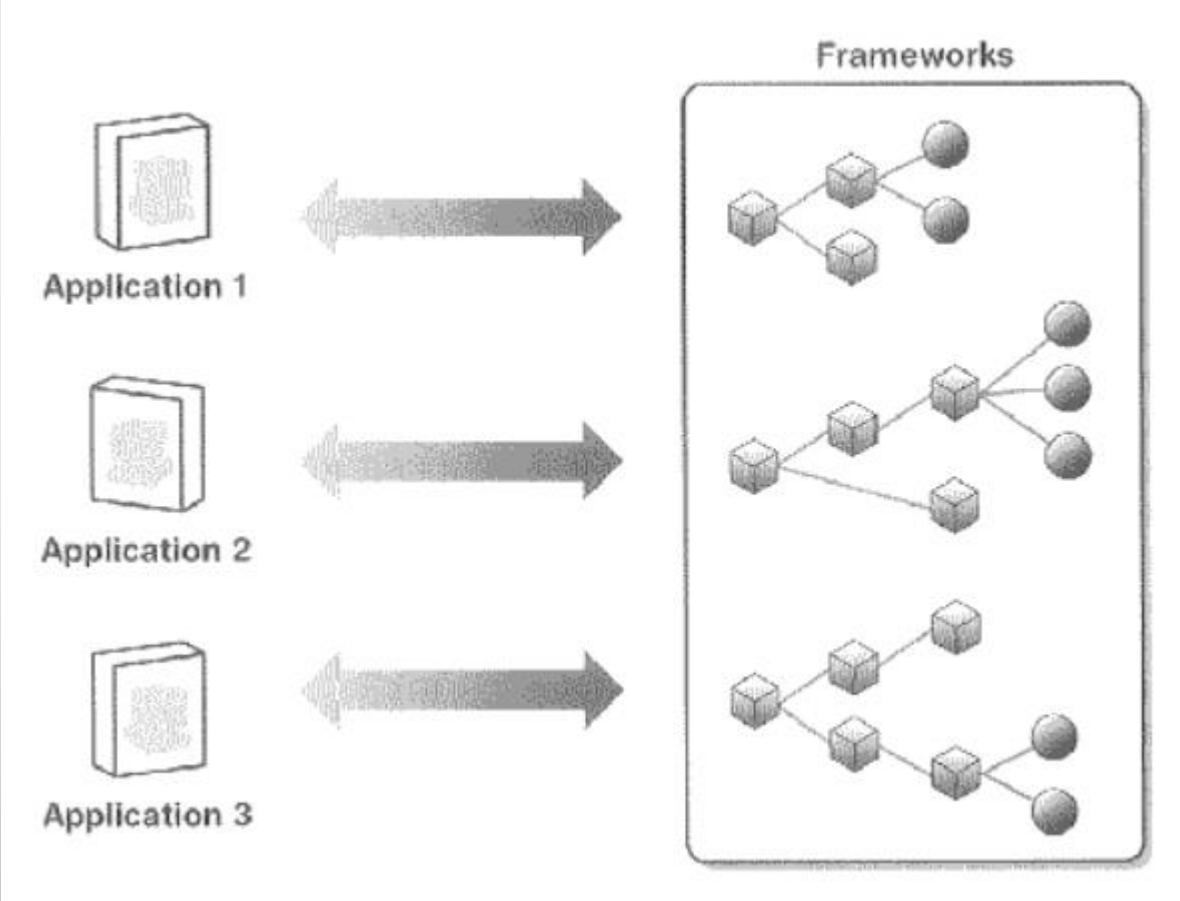


Leveraging Frameworks for EAI

- Consist of abstract classes and their object collaboration as well as concrete classes.
  - While Object-oriented programming supports software reuse, frameworks support design reuse

(Freedman, 1993)

Leveraging Frameworks for EAI (cont.)



The Value of Frameworks

- The most obvious benefit of frameworks is the ability to reuse existing code.
- Frameworks provide more than reuse within an organization; they provide reuse throughout an industry—such as shipping or logistics business—where many of the processes are the same and reusable frameworks make sense.
- Frameworks also provide a standard set of predesigned and pretested application architectures.
- Frameworks hide all the "dirty little details" from the developer

Framework Types

- **Object frameworks** are made up of both abstract and concrete classes
  - Developers get a copy of the framework source code and have an opportunity to customize the framework
- **Service frameworks**
  - They provide services and, as such, are the best fit from the perspective of most EAI problem domains
- **Procedural Frameworks**
  - If a functionality required for a particular application in a procedural language is missing…tough luck. The procedural framework itself will have to be modified to add the functionality, or the function will have to be coded directly into the application without the use of a framework.

Framework Types (cont.)

- **Component frameworks**
  - Components deal with the interface on the client, where application services and procedural frameworks provide basic application services on the back-end. In addition, component frameworks don't provide the reuse granularity of true object-framework development. Instead, they provide reusable features that eliminate the need-to-know component implementation details.

Framework Categories

- Three categories of frameworks based on the types of features they offer:
  - **Application service**
    - Most frameworks today are examples of application service frameworks. such as Microsoft's Foundation Classes (MFC)
  - **Domain**
    - The idea here is to build common application architectures into a common framework shared across applications (e.g., accounting, marketing, or logistics)
  - **Support**
    - Offer native system-level services such as network support, device access, or file access

➢ Any of these frameworks could be object, service, procedural, or component enabled.