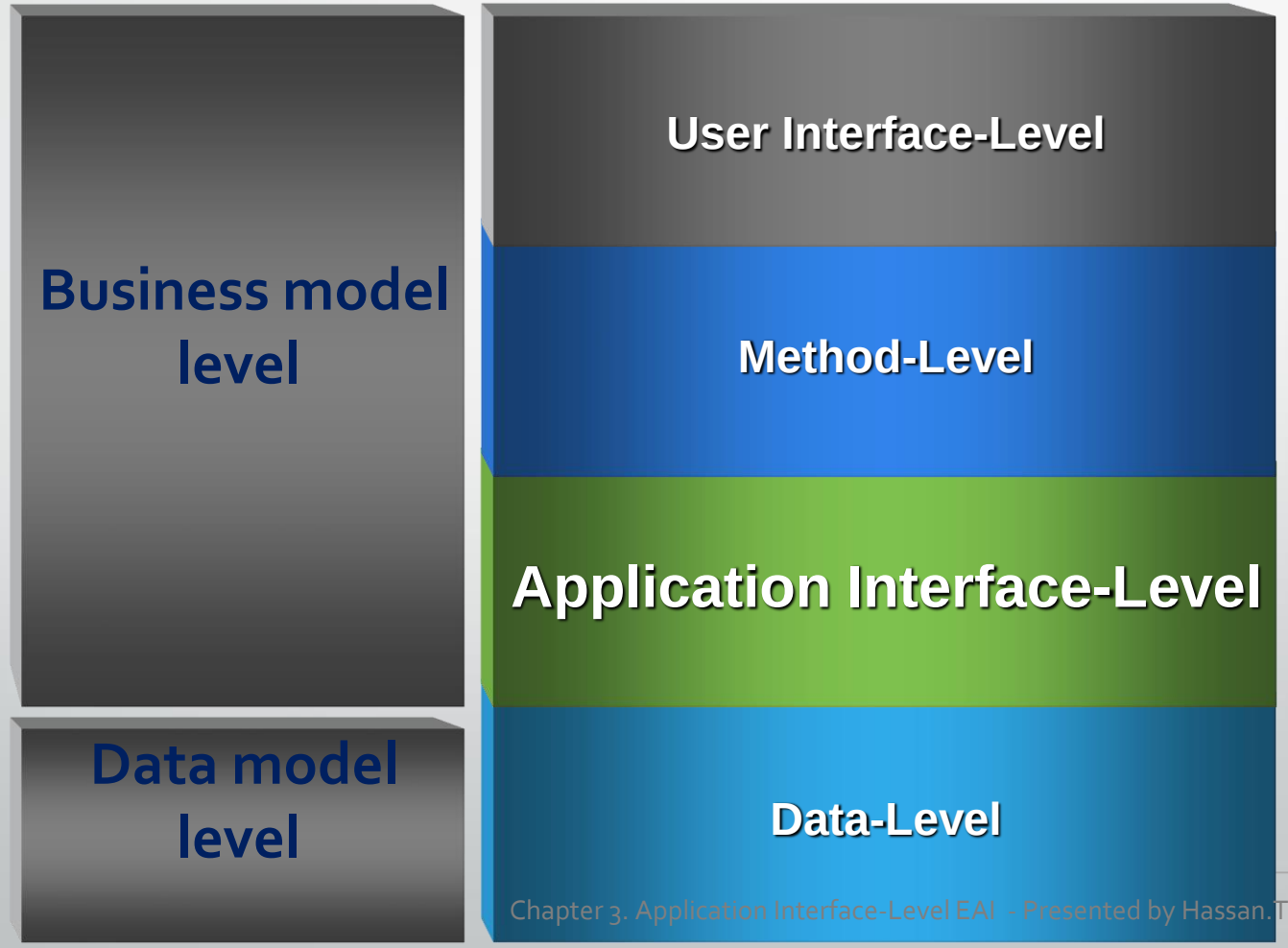
Enterprise Application Integration (EAI)

Chapter 3: Application Interface-Level EAI

Introduction



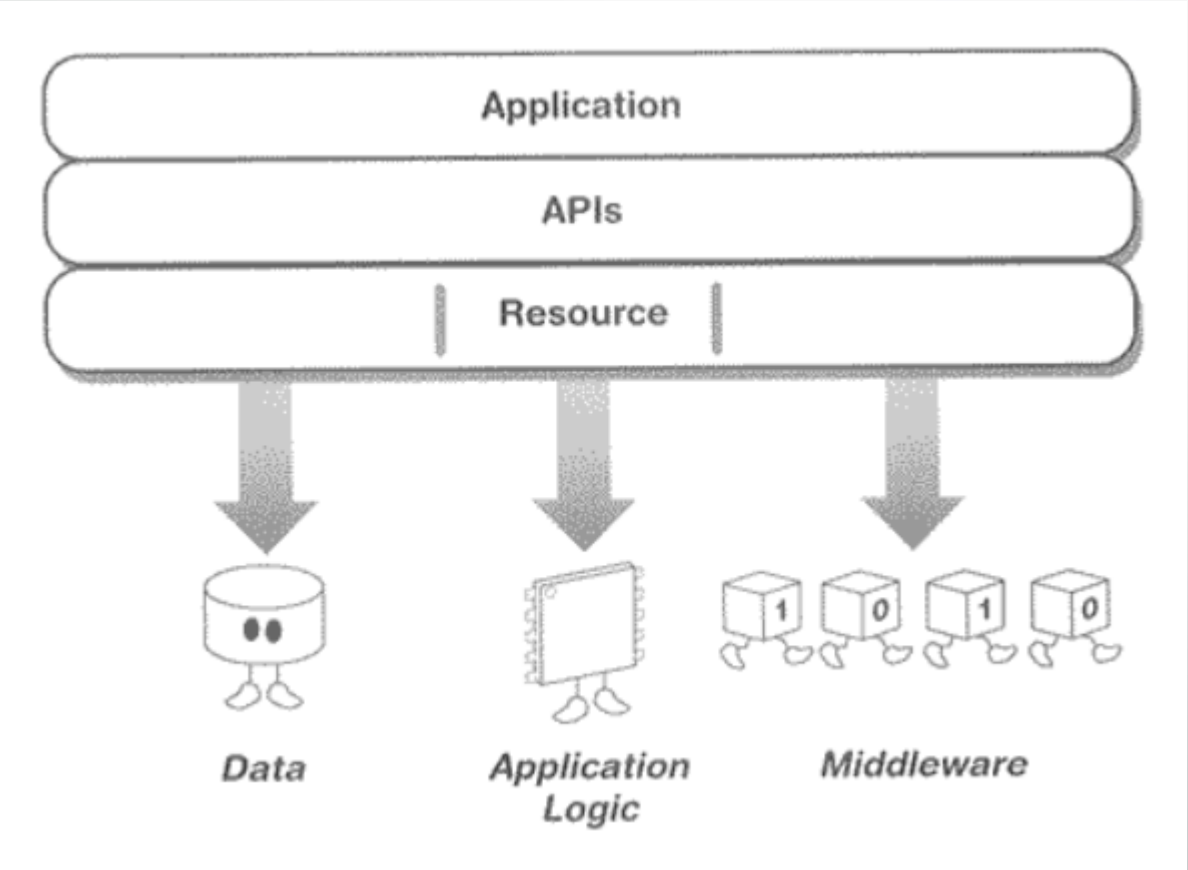
Definition

- **Application interface**s are interfaces that developers expose from a packaged or custom application to gain access to various levels or services of the application.
- Sometimes these interfaces allow access to business processes; sometimes they allow access directly to the data. Sometimes they allow access to both

What's an API?

- **Application Programming Interfaces (APIs)**
  - Is essential to understanding application interfaces
  - Are well-defined mechanisms that are built to connect to some sort of resource such as an application server, middleware layer, or database

APIs


Application
APIs
Resource
1
0
1
0
Data
Application Logic
Middleware

Interface by Example

- A company maintains two systems: an ERP system that was just configured and installed and a custom COBOL system that has been functioning for years. Each system exists on its own processor, connected by the corporate network.
- Assume that the data-level EAI solution won't work due to the complexity of the databases and the binding of logic to the data

Interface by Example (cont.)

- GetInvoiceInformation("12345");

```
<BOM>
John Smith
222 Main Street
Smalltown, VA 88888
Invoice Number: 12345
001      Red Bricks 1000      .50      500.00
<EOM>
```

The Interface Tradeoff

- If the world were a perfect place, all the features and functions provided by packaged applications would also be accessible through their "well-defined" application programming interfaces.
- Most disturbing is that many packaged applications offer no interface whatsoever.

user interface-level or data-level EAI

Packaged Applications

- Come in all shapes and sizes
- SAP, PeopleSoft, Oracle, and
- They offer certain advantages over their competitors
- **<u>These</u> represent special challenges for EAI**

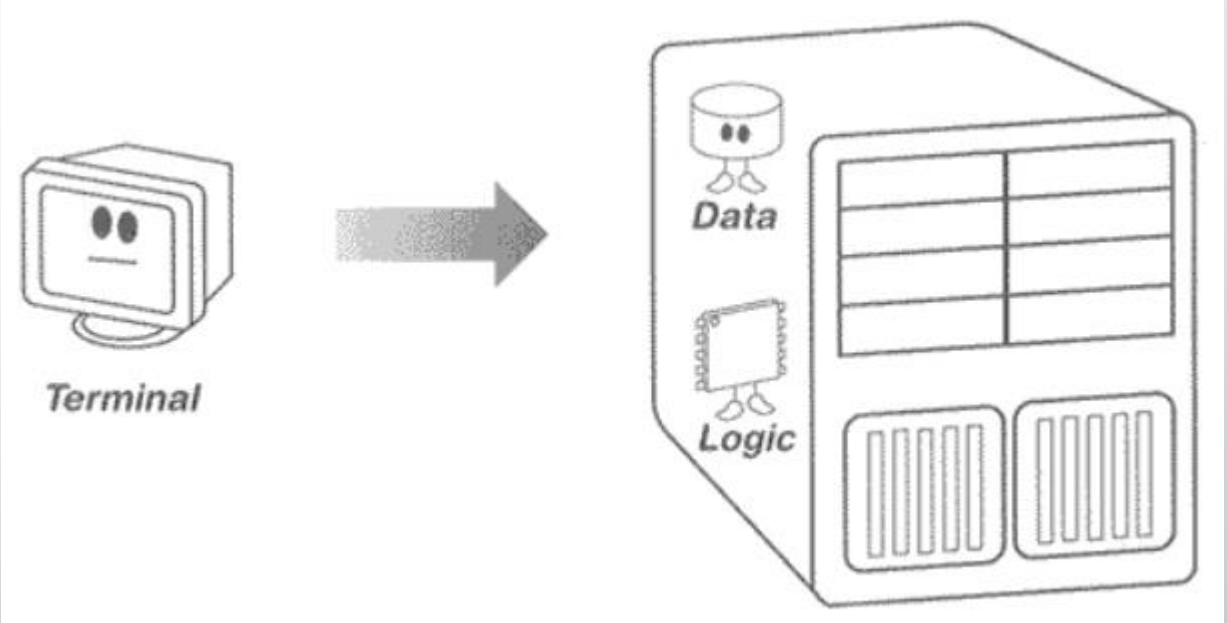
It is important to remember that, over the years, hundreds of packaged applications have entered the enterprise. Many of these no longer enjoy the support of their vendors, or, just as likely, the vendors are out of business.

Packaged Application Technology Architecture

- Centralized (most traditional)
- Two-tier
- Three-tier

Using the centralized architecture

- Places both data application logic and user interfaces within the same machine



Advantages to centralized architecture

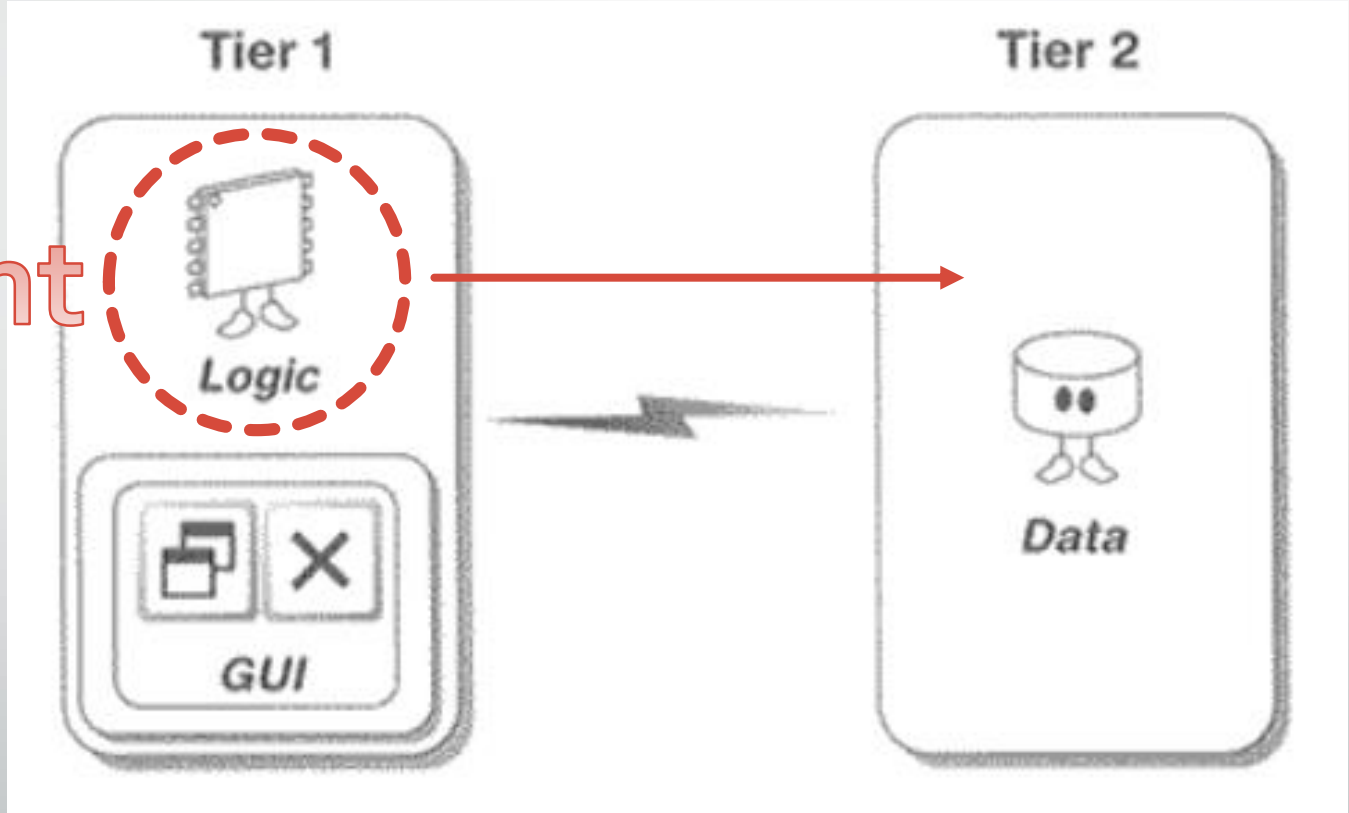
- Maintenance is much easier than in traditional distributed environments
- Integration is more easily accomplished within one machine than among several
- Encapsulated services and data that reside on a central computer can be accessed by a single gateway

Two-tier architecture

- drawn from the traditional, two-tier client/server model where the application is physically and logically separated onto just two layers
  - Client
    - Always contains the user interface, but it may or may not also contain the business logic
  - Server

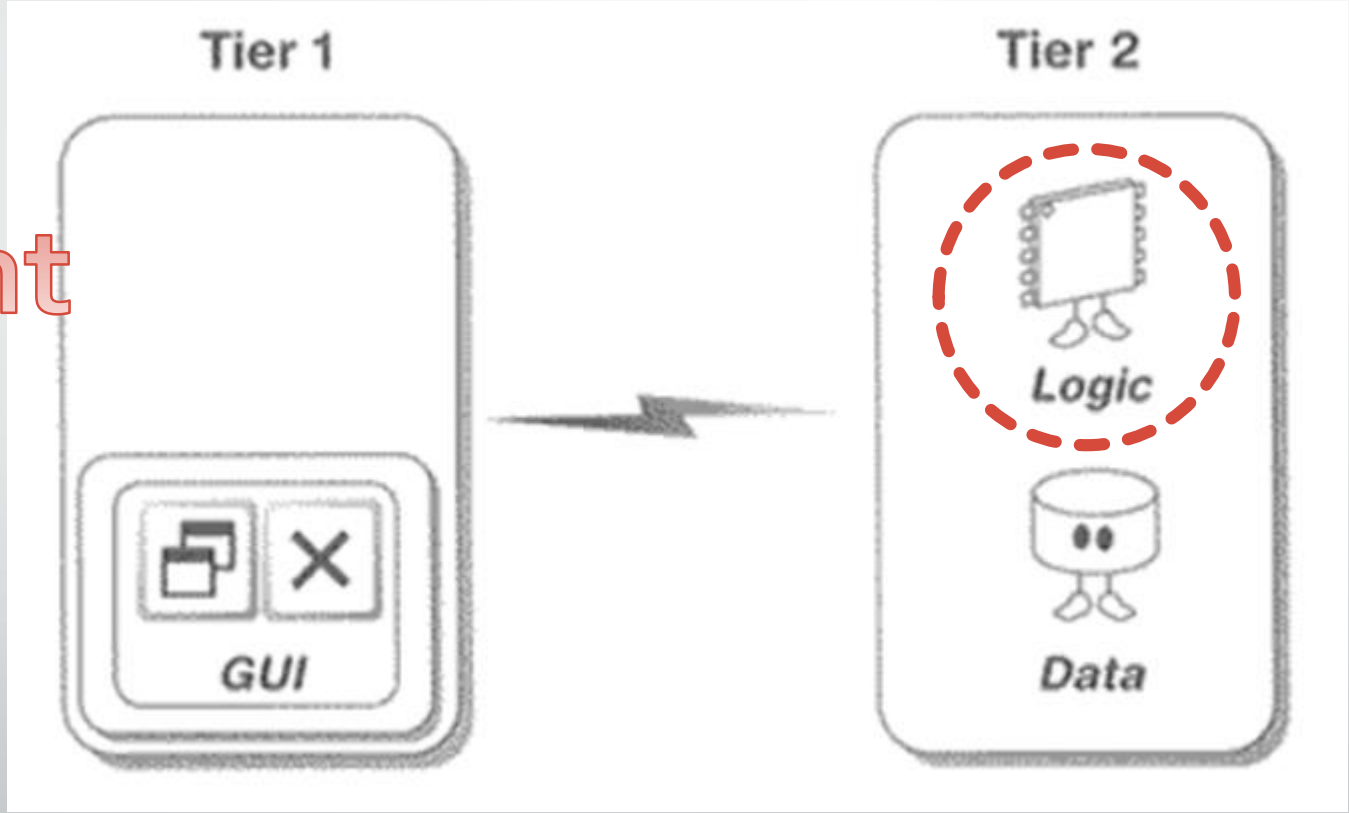
Two-tier architecture (cont.)

"fat" client

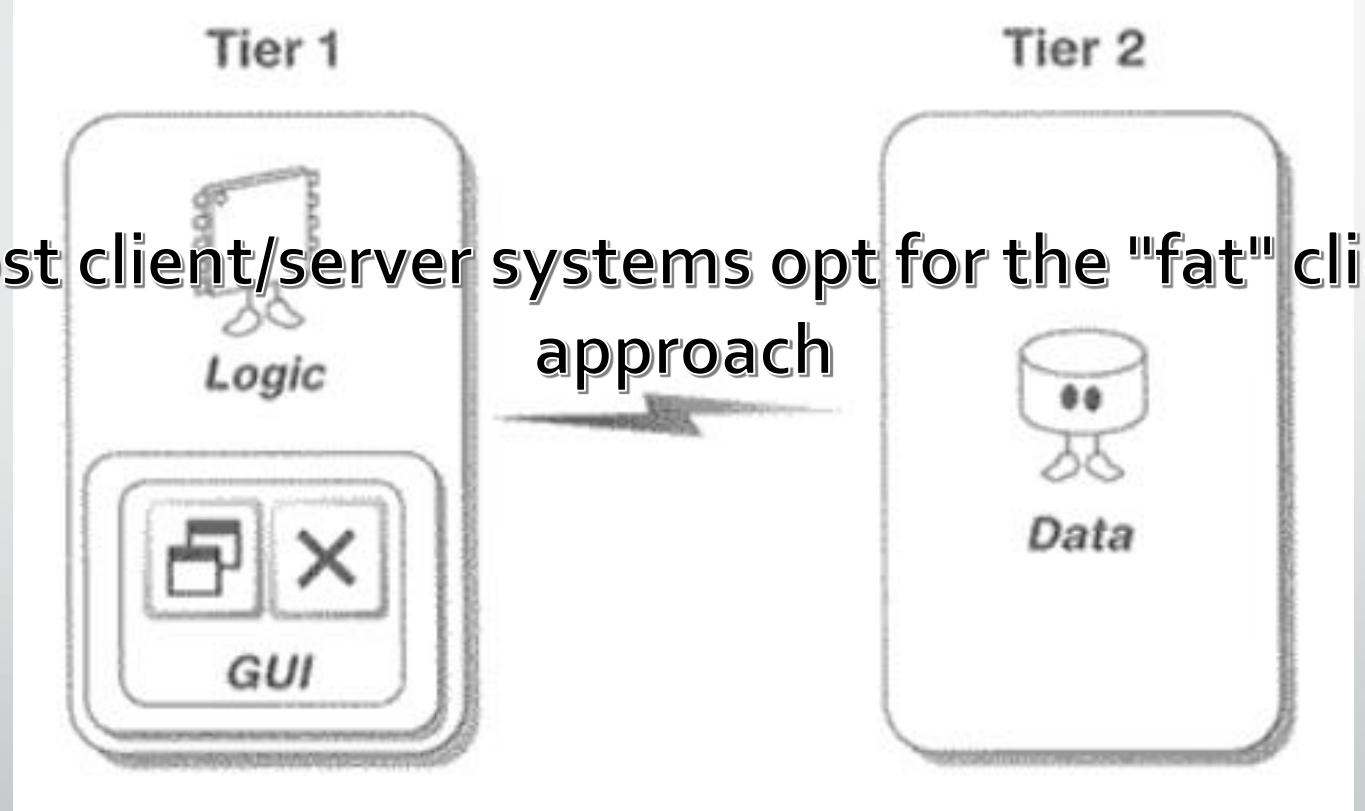


Two-tier architecture (cont.)

"thin" client



Two-tier architecture (cont.)



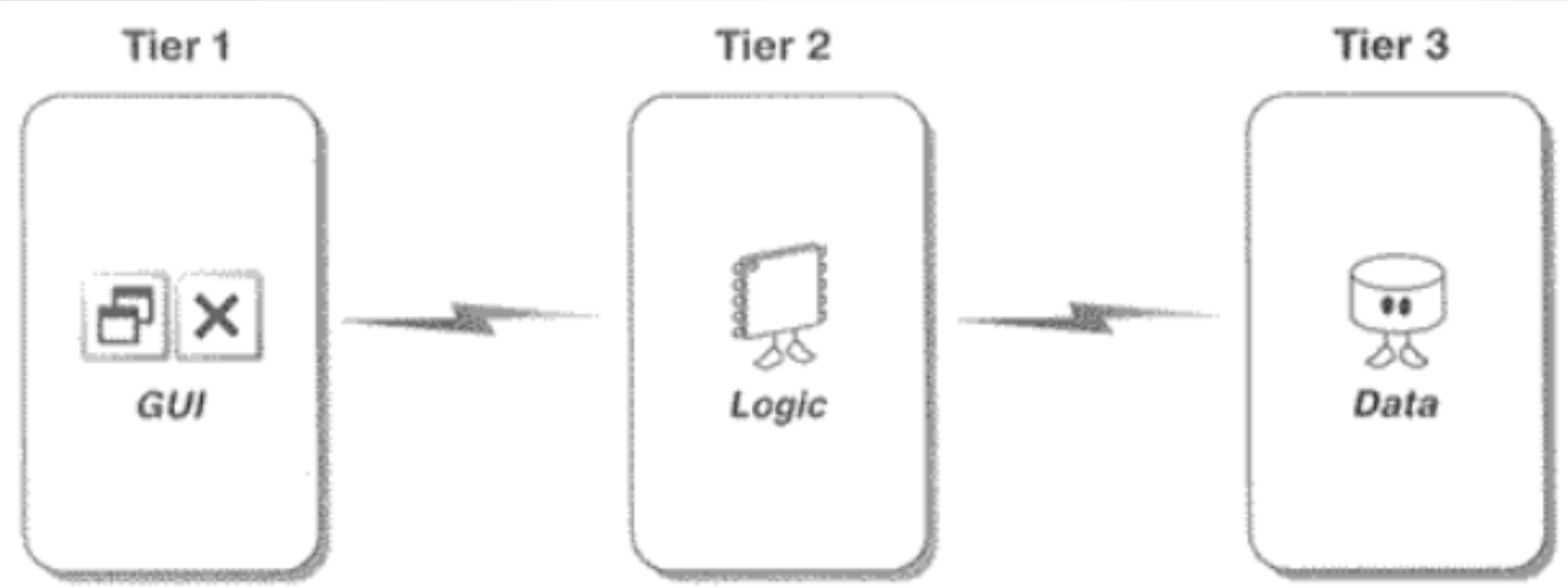
Most client/server systems opt for the "fat" client approach

Three-tier architecture

- drawn from the traditional, two-tier client/server model where the application is separated into three distinct layers:
  - User interface layer
  - Business logic layer
  - Data layer

The most popular architecture among packaged applications

Three-tier architecture (cont.)



Advantages to three-tier architecture

- Provides a clean separation between the user interface, the business logic, and the data
- Provides enhanced scalability
- Database funneling or connection pooling
- Rather than the
  - Direct proportionality of the two-tier architecture, 100 clients in three-tier architecture might require only ten connections to the back-end database.

Packaged Application APIs

- Packaged applications expose interfaces or APIs that allow other applications to access encapsulated services and data
- There are three types of services available to these interfaces:
  - Business service
  - Data service
  - Objects

Types of Services

- **Business services**, include interfaces to any piece of business logic that may exist within the packaged application
- While most application interfaces provide **data services**, the EAI architect has the option of going directly to the database by utilizing database-oriented middleware
- **Objects** are simply data and business services bound as objects

Types of Interfaces

- Full-Service
- Limited-service
- Controlled