Useful Patterns

Exploring More Patterns

Jeremy Clark
www.jeremybytes.com
jeremy@jeremybytes.com

pluralsight
hardcore developer training

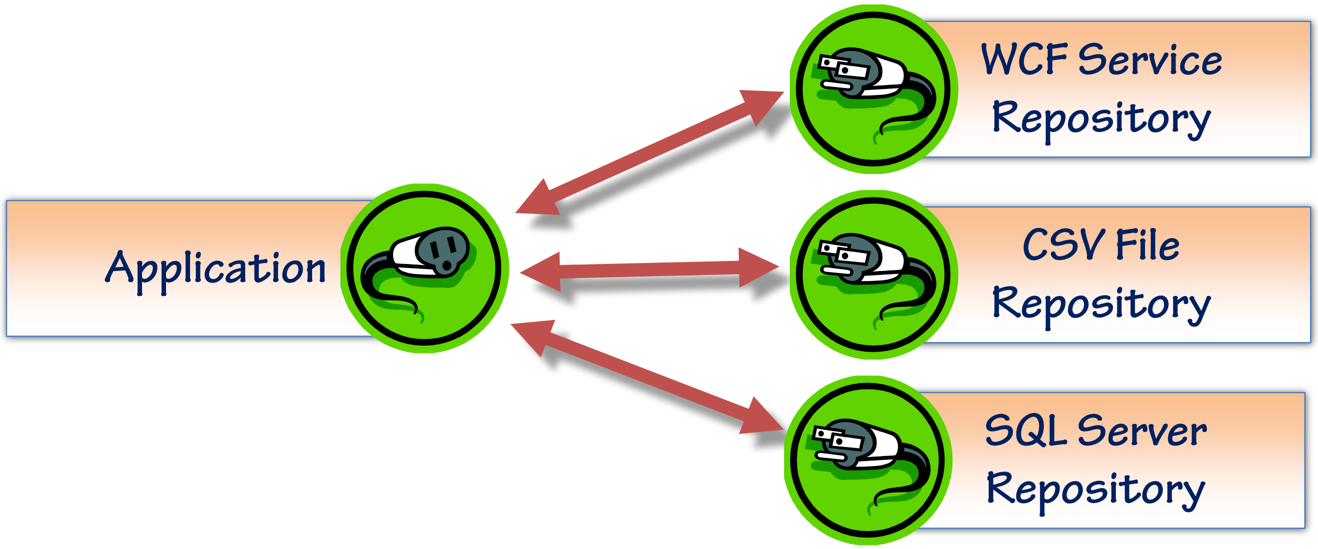
Factory Method Pattern

Define an interface for creating an object, but let subclasses decide which class to instantiate. Factory Method lets a class defer instantiation to subclasses.

Gamma, et al, *Design Patterns*. Addison-Wesley, 1994.

Pluggable Repositories

- Accessing Different Data Sources

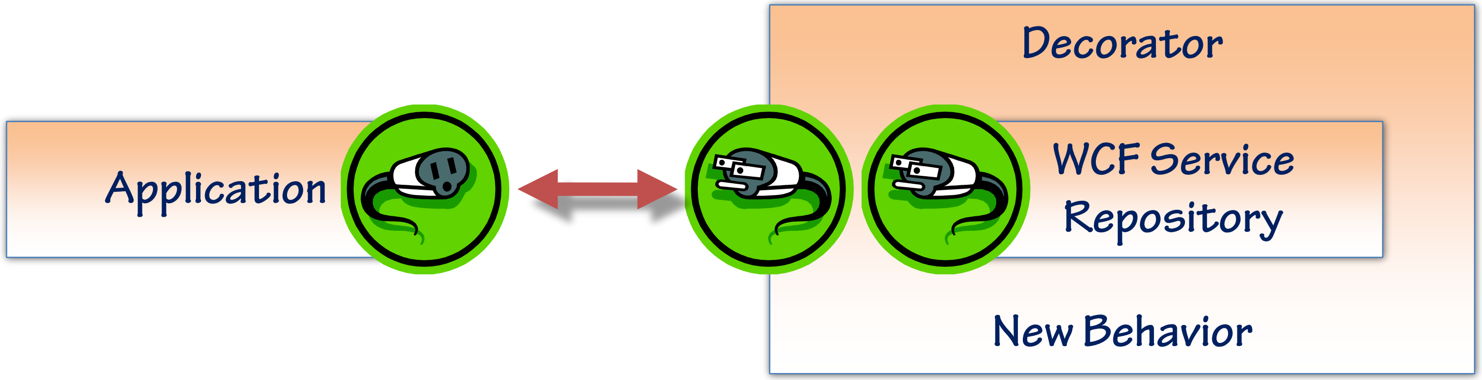


Decorator Pattern

Attach additional responsibilities to an object dynamically. Decorators provide a flexible alternative to subclassing for extending functionality.

Gamma, et al, *Design Patterns*. Addison-Wesley, 1994.

Decorating an Object

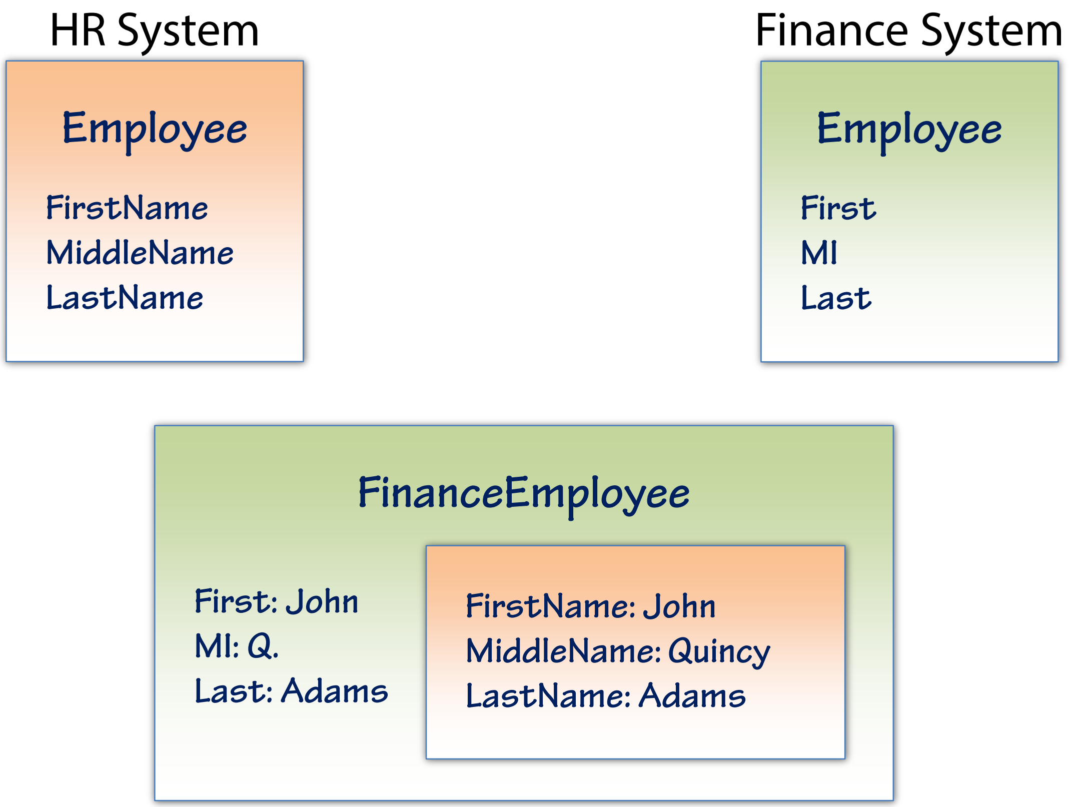


Adapter Pattern

Convert the interface of a class into another interface clients expect. Adapter lets classes work together that couldn't otherwise because of incompatible interfaces.

Gamma, et al, *Design Patterns*. Addison-Wesley, 1994.

Adapter Example



SQL Data Adapter

Application Class

```
public class Person
{
  string FirstName { get; set; }
  string LastName { get; set; }
  DateTime StartDate { get; set; }
  int Rating { get; set; }
}
```

SQL Data Entity

```
public class DataPerson
{
  string FirstName { get; set; }
  string LastName { get; set; }
  DateTime? StartDate { get; set; }
  int? Rating { get; set; }
}
```

ApplicationPerson

```
DateTime StartDate
{
  get { return _dataPerson.StartDate.Value; }
}
```

Summary

- **Factory Method**
  - Creating Objects
  - RepositoryFactory
- **Decorator**
  - Adding Functionality
  - CachingRepository
- **Adapter**
  - Resolving Incompatible Interfaces
  - ApplicationPerson
- **Next Up: Where To Go Next**
  What to do with our newfound knowledge