

## Re: design pattern

**Source:** <http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework/2005-02/1512.html>

---

**From:** Nick Malik [Microsoft] ([nickmalik\\_at\\_hotmail.nospam.com](mailto:nickmalik_at_hotmail.nospam.com))

**Date:** 02/25/05

Date: Fri, 25 Feb 2005 08:30:26 -0800

The pattern is called Inversion of Control. It is heavily described in various literature and forms the basis of a branch of computing called Aspect Oriented Programming.

There is a port of the AOP framework "Spring" in the .Net spaces available here:

<http://www.springframework.net/>

This is an exciting area, and I encourage you to dig in.

--

--- Nick Malik [Microsoft]  
MCSD, CFPS, Certified Scrummaster  
<http://blogs.msdn.com/nickmalik>

Disclaimer: Opinions expressed in this forum are my own, and not representative of my employer.

I do not answer questions on behalf of my employer. I'm just a programmer helping programmers.

--

"Hugo Batista" <[hlatibataATgmail@nospam.com](mailto:hlatibataATgmail@nospam.com)> wrote in message  
news:eEYm0IzGFHA.2136@TK2MSFTNGP14.phx.gbl...

> Hi everybody,

>

> I would like to have your opinion...

>

>

> Imagine i have a pattern that defines:

>

> - abstract factory : my consumer always handle with a interface and not  
> with direct implementation, and calls a factory to get an object instance;

> - the kind of object returned in that instance varies according to some  
> settings and, in runtime, the factory reads settings and return the  
> configured object type that implements interface;

> -Singleton/singlecall: the returned object can be singleton or singlecall  
> and that can be changed in settings at any time. The consumer should not  
> care about this and the factory handles this to know if he should maintain  
> a singleton instance or not;

> - inproc/outproc : the object can be a remote one or a local one. the  
> consumer does not know this and the factory handles it all. that is also  
> defined in settings;

>

>

> a settings example:

>

> <PatternProviderSettings xmlns:xsd="<http://www.w3.org/2001/XMLSchema>"

## microsoft.public.dotnet.framework: Re: design pattern

```
> xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
>   <ObjectMode>SingleCall</ObjectMode>
>   <AllowWeakReferenceOnSingleton>>false</AllowWeakReferenceOnSingleton>
>   <ProcessModel>InProc</ProcessModel>
>   <RemotingUrl>tcp://localhost:9999/LocalPatternsProvider</RemotingUrl>
>   <Provider>DotNetX.CodeGeneration.Patterns.LocalPatternsProvider,
> DotNetX.CodeGeneration</Provider>
> </PatternProviderSettings>
>
>
>
> - Objectmode allows singleton or singlecall
> - AllowWeakReferenceOnSingleton indicates to the factory if he should keep
> a weakreference or a reference, so the GC can handle it if needed
> - ProcessModel indicates inproc or outproc
> - RemotingUrl indicates the object remote url if outproc
> - Provider indicates the final implementation (the type of object to be
> created)
>
>
> my question is:
> between existing enterprise patterns, which do you think that apply to
> this ? i already know that abstract factory applys, but i wonder if other
> also apply ... what about strategy design pattern ? do you think it
> applies ?
>
>
> regards!
> and thanks for your contribution..
> HB
>
> www.dotnetx.org
>
>
>
```