D'accord, voyons un exemple d'un diagramme UML pour un scénario utilisant le design pattern Proxy et comment vous pourriez l'implémenter en C#.

Considérez un cas d'utilisation où un client veut télécharger un fichier volumineux à partir d'un serveur. Nous utiliserons un Proxy pour gérer le téléchargement du fichier de manière plus efficace en le cachant localement après le premier téléchargement.

Voici comment pourrait être le diagramme UML :

lua

Client

^

|

| +------------------+

+--| <<Interface>> |

| IFileDownloader |

+------------------+

| +DownloadFile() |

+------------------+

^ ^

| |

+------------------+ +----------------------+

| RealFileDownloader| | CachedFileDownloader |

+------------------+ +----------------------+

| +DownloadFile() | | +DownloadFile() |

| | | +realDownloader |

+------------------+ +----------------------+

* Le Client utilise l'interface IFileDownloader pour télécharger des fichiers.
* RealFileDownloader est la classe qui implémente vraiment le téléchargement de fichiers.
* CachedFileDownloader est notre Proxy qui gère le cache des fichiers téléchargés pour une utilisation ultérieure.

Voici l'implémentation C# pour ce scénario :

csharp

public interface IFileDownloader

{

void DownloadFile(string url);

}

public class RealFileDownloader : IFileDownloader

{

public void DownloadFile(string url)

{

// Logic to download file from the URL

}

}

public class CachedFileDownloader : IFileDownloader

{

private RealFileDownloader realDownloader;

private Dictionary<string, string> cachedFiles;

public CachedFileDownloader()

{

realDownloader = new RealFileDownloader();

cachedFiles = new Dictionary<string, string>();

}

public void DownloadFile(string url)

{

if (!cachedFiles.ContainsKey(url))

{

realDownloader.DownloadFile(url);

cachedFiles.Add(url, "File Data");

}

else

{

Console.WriteLine("File retrieved from cache");

}

}

}

Dans cet exemple, le RealFileDownloader effectue le véritable téléchargement du fichier, tandis que le CachedFileDownloader sert de proxy pour gérer le cache des fichiers téléchargés.

C'est un exemple simple pour illustrer le pattern Proxy, en réalité vous auriez probablement des méthodes pour écrire/charger le fichier depuis un emplacement local, gérer les erreurs de téléchargement et autres.