Flyweight and Proxy

3-9-2004

Opening Discussion

- Const objects/references requirer const methods.
- Interfaces should be minimal yet complete.
- Unions in C were a feeble (and unsafe) way of providing polymorphism.
- Do you have any questions about the reading for today?

Flyweight

- The objective of this pattern is to allow the sharing of objects to efficiently allow systems that would otherwise require too many objects.
- In order for this to work well, the shared objects need to be identical. You can sometimes share only some of the data between objects and pass in differences.
- You should note that immutable objects can make this safer and a bit more flexible in some instances.

Example

- GoF uses characters in a text document as an example. Without flyweight these can't be objects. With it you have a table/pool of the characters and then reference them.
- You could have a different table for different fonts though personally I would consider that as something that could be passed in. Otherwise changing the font on a paragraph is a somewhat time consuming process of redirecting many pointers.

Benefits and Drawbacks

- Using the flyweight can have overhead costs to speed because you have to look things up. These are offset by memory savings. If creating objects is costly the speed might not really be hurt either.
- How much you save depends on how many objects you need and how much sharing happens. Big systems can gain more.
- This is often combined with Composite to create a tree like structure.

Proxy

- With this pattern you provide a placeholder or surrogate for a given object that gives you remote/indirect access to it.
- The primary use of this is when the creation of certain objects is expensive, we don't want to create them unless they are needed and only when they are needed. Instead we create a proxy object that can answer simple questions about the full object and only instantiates the full object when required.

Example

- GoF uses the example of an image in a document. These can be expensive to load and store so the proxy only gets enough information for the size, not the full raster. The full raster is created and remembered only when it is viewed.
- This pattern is very similar in many ways to how dynamic loading can happen in the project. We keep a proxy object for things that aren't currently viewed and those objects load when needed.
- RMI also works a a Proxy.
- In C++ the auto_ptr class is a proxy for pointers. Similar things can be done that allow reference counting.

Benefits and Drawbacks

- The extra level of indirection for a Proxy enables numerous types of enhancements depending on the type of proxy object.
  - Remote proxy
  - Virtual proxy
  - Protection proxy
  - Smart reference
- A copy-on-write proxy allows sharing of large objects as long as they aren't changed. It includes reference counting. The large object is only copied when modified.

Progress Presentation

- Is there anyone who wants to report on their progress in the project?
- After Spring Break this will have to go into full swing as there are quite a few of you who haven't presented yet.
- You should all have things to hand me today.