Running Head: DESIGN PATTERNS

Week 3 Discussion: Design Patterns

Jered McClure

Walden University

Week 3 Discussion: Design Patterns

Decorator Pattern and Proxy Pattern are both offshoots of the Adapter Pattern. Both patterns are similar in that they wrap an object thereby acting as a layer of indirection. However, each performs a slightly different function to that of the Adapter pattern, and to each other.

Where an adapter pattern changes the interfaces of an object, the Proxy Pattern changes the behavior of the object (Net Objectives, 2009). Essentially, “proxies control access to their real subject” (Geary, 2002). This means that the proxy acts as a level of indirection. Thus, allowing through only those threads which have permission and or reason to access the object (Source Making, NA)

Unlike an Adapter Pattern, the Decorator Pattern adds to the object’s functions without affecting the object’s interface (Portland Pattern Repository, 2011). In essence, the Decorator Pattern allows the programmer to extend the overall functions of an object without changing the code within the object itself. However, this does not mean that all objects should be decorated, or that they be open to extension. This is because adding Decorators, or Proxies, to classes adds complexity to the overall system (Freeman, Freeman, Sierra, & Bates, NA).

Reference

Freeman, E., Freeman, E., Sierra, K., & Bates, B. (NA). *Headf First Design Patterns.* Retrieved July 31, 2012, from O'Reilly: http://oreilly.com/catalog/hfdesignpat/chapter/ch03.pdf

Geary, D. (2002, February 2). *Take Control with the Proxy design Pattern.* Retrieved July 31, 2012, from Java World: http://www.javaworld.com/javaworld/jw-02-2002/jw-0222-designpatterns.html?page=2

Net Objectives. (2009, June 10). *AdapterVersusProxyVersusFacadePatternComparison.* Retrieved July 31, 2012, from Net Objectives: http://www.netobjectivestest.com/PatternRepository/index.php?title=AdapterVersusProxyVersusFacadePatternComparison

Portland Pattern Repository. (2011, March 21). *Decorator Pattern*. Retrieved July 31, 2012, from Portland Pattern Repository: http://c2.com/cgi/wiki?DecoratorPattern

Source Making. (NA). *Proxy Design Pattern.* Retrieved July 31, 2012, from Source Making - Teaching IT Professionals: http://sourcemaking.com/design\_patterns/proxy