Applying Recursion and Fractals As Design Patterns In Complex Systems

Steve Mazza
Naval Postgraduate School

The term paper must be at least ten pages in length (single-spaced, normal font size, graphics are included in page count). There is no maximum length, although a concise, well-crafted analysis is clearly superior to a lengthy, rambling narrative. The subject of the term paper will be the student's synthesis of how some aspect of complexity, chaos, and other related topics relates to engineering of complex systems. That is, how does each student think at least one of these topics can be integrated into a general approach to systems engineering. The term paper will be due in class during Week 12.

Introduction

Present what I am going to say. (Mandelbrot, 1983, page 112)

I will show how applying recursive design patterns to complex systems reduces the complexity of their description. Furthermore, patterns for the recursive descriptions can be found in some classes of fractals. There exists an analog between certain fractals and the recursive descriptions for complex problems.

Background

Discuss descriptions of complex problems. Introduce methods of simplifying complexity. Introduce recursion as a solution to some classes of problems.

What Is Complexity?

What is complexity? What is the character that makes some systems complex and others not? Is there a hard line between complexity and non-complexity or do we just know it when we see it?[1]

At the root, problems are complex when they are difficult to understand or predict. Murray Gell-Mann proposed that systems should be considered complex when the difficulty in predicting their outcome is due, not to irregularities (or randomness), but to regularities in the system that are difficult to describe. This, he suggests, differentiates between purely random outcomes which are not necessarily complex and complex systems whose outcomes are difficult to predict. (Gell-Mann, 1995)

Miller counters that there should possibly not be any single unified definition of complexity, arguing that trying to unify them may be asking too much.

> Complexity can occur at many levels, including time, space, and interactions. Perhaps we are expecting too much if we want a single measure of complexity that captures all of our intuitions. (Miller, 2009, page 234)

There are several ways in which we can think about complexity, but we will focus on the definition proposed by

1. Introduce the term
2. Provide (and cite) a definition
3. Discuss interactions and how they lead to complexity
4. Provide an example showing the difference between complex and non-complex systems.

How to Describe Complex Systems

1. Discuss the importance of formal methods.
2. Talk about frameworks and discuss some standards.
3. Briefly talk about design patterns.
4. Talk about abstraction and simplification.

Methods Of Simplifying Complexity

1. Introduce abstraction and discuss (definition and examples).
2. Introduce encapsulation and discuss (definition and examples).
3. Discuss interface definition and contracts.
4. Discuss black box architecture.

Introducing Recursion

1. Provide (and cite) a definition.
2. Introduce the concept through example. Consider the description of some algorithm from computer science such as *factorial*.

[1] This is a clear reference to Justice Stewart's description of pornography. The point is, can complexity be definitely identified (as in the legal sense) or is its character more elusive?

3. Introduce the idea that recursion can be applied to the physical world and provide an example.

Fractals

Introduction

Introduce several classes of fractals (with figures).
1. Koch snowflake
2. Sieve

Consult Benoit Mandelbrot for examples and explanation.

Constructing Fractals

Describe and demonstrate their construction, emphasizing self-similarity and recursion.

Self-similarity and Fractals

Discuss self-similarity and its connection to fractals. Discuss how we can describe fractals at all levels of magnification by describing their construction at one given level. Show how simple rules can lead to complexity.

Design Patterns

Introduce the notion of using design patterns to describe solutions to problems. Point out that traditional patterns don't adequately encode recursion. Propose the use of some classes of fractals in describing recursive solutions to complex problems.

Use Of Design Patterns

Introduce the history and use of design patterns. (Gamma, Helm, Johnson, & Vlissides, 1994)

Shortcomings Of Traditional Design Patterns

Show the shortcomings of traditional design patterns in modeling complex systems.

Use of Fractals as Design Patterns

Propose the use of some classes of fractals as design patterns.

Example

Work through an example (with figures)

From Simplicity, Complexity

Discuss how simple rules can quickly lead to complexity.

Conclusion

Wrap up. Summarize what was covered. Restate the proposal to use certain classes of fractals to describe recursive solutions to complex problems.
1. Summarize and wrap up.
2. Restate the proposal

References

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design patterns: elements of reusable object-oriented software*. Pearson Education.

Gell-Mann, M. (1995). What is complexity? remarks on simplicity and complexity by the nobel prize-winning author of the quark and the jaguar. *Complexity*, *1*, 16-19.

Mandelbrot, B. B. (1983). *The fractal geometry of nature*. Macmillan.

Miller, . P. S. E., J. H. (2009). *Complex adaptive systems: An introduction to computational models of social life: An introduction to computational models of social life*. Princeton University Press.