Functional Decomposition Encyclopedia Article

Functional Decomposition

The following sections of this BookRags Literature Study Guide is offprint from Gale's For Students Series: Presenting Analysis, Context, and Criticism on Commonly Studied Works: Introduction, Author Biography, Plot Summary, Characters, Themes, Style, Historical Context, Critical Overview, Criticism and Critical Essays, Media Adaptations, Topics for Further Study, Compare & Contrast, What Do I Read Next?, For Further Study, and Sources.

(c)1998-2002; (c)2002 by Gale. Gale is an imprint of The Gale Group, Inc., a division of Thomson Learning, Inc. Gale and Design and Thomson Learning are trademarks used herein under license.

The following sections, if they exist, are offprint from Beacham's Encyclopedia of Popular Fiction: "Social Concerns", "Thematic Overview", "Techniques", "Literary Precedents", "Key Questions", "Related Titles", "Adaptations", "Related Web Sites". (c)1994-2005, by Walton Beacham.

The following sections, if they exist, are offprint from Beacham's Guide to Literature for Young Adults: "About the Author", "Overview", "Setting", "Literary Qualities", "Social Sensitivity", "Topics for Discussion", "Ideas for Reports and Papers". (c)1994-2005, by Walton Beacham.

All other sections in this Literature Study Guide are owned and copyrighted by BookRags, Inc.

Contents

Functional Decomposition Encyclopedia Article.....1

Contents.....2

Functional Decomposition.....3

Functional Decomposition

The term functional decomposition is used to describe the technique for breaking down a task into a number of smaller and simpler sub-tasks. It is a technique that is widely used in computer **programming** and it confers many advantages on the **design** and implementation process.

The goal of functional decomposition is to reduce a system step-by-step to a collection of irreducible fundamental functions, starting at the topmost level of functionality. For example, if we have a program that simulates a lawnmower, the main functional description might be "cut the grass." At each level of reduction, abstractions are projected from the level immediately below. The collection of sub-functions that comprises the decomposed main **function** are called the "functional hierarchy." Using the lawnmower example, the second level of functionality might include the sub-function "propel lawnmower," which itself could include the sub-functions "start engine" and "rotate wheels."

Sometimes a functionally decomposed design is called "black box design" because each function has an interface, in the form of function parameters, that represents its inputs; a return **value** that represents its outputs; and a behavior that is fully specified for all combinations of inputs. In other words, the function is opaque like a black box: it does not matter *how* the function does something; *what* it does is what matters. In this way, functional decomposition makes testing easier because each **functional unit** or black box can be designed, coded, and tested independently before being combined with its peers into the next level of functionality in the functional hierarchy.

This reduced testing is extremely important in complex systems. As an example, imagine a simple system that takes four inputs, each one of which can be either true or false. Simple math shows that the number of possible input patterns is 2^4 or 16. But if this function is broken down into two sub-functions each taking two parameters, then each one has a maximum of 2^2 or 4 patterns of input. Because there are two functions, the total number of test input patterns is 8, half of the original. And as systems get even more complex, the savings can be immense.

Functional decomposition also encourages code-reuse, as small and useful sub-functions tend to get written into function libraries where they can be used again and again by many different programs and programmers.