7 Design Considerations

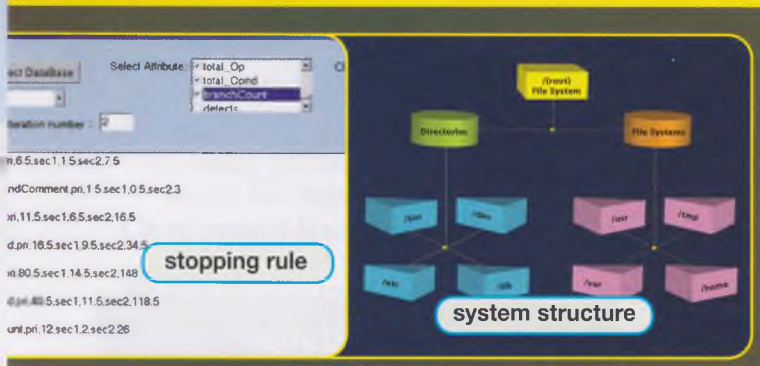
Date: 14 June

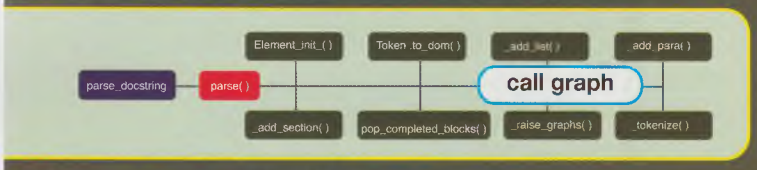
Subject: A Little Planning Leads to a Big Payoff in Software Design

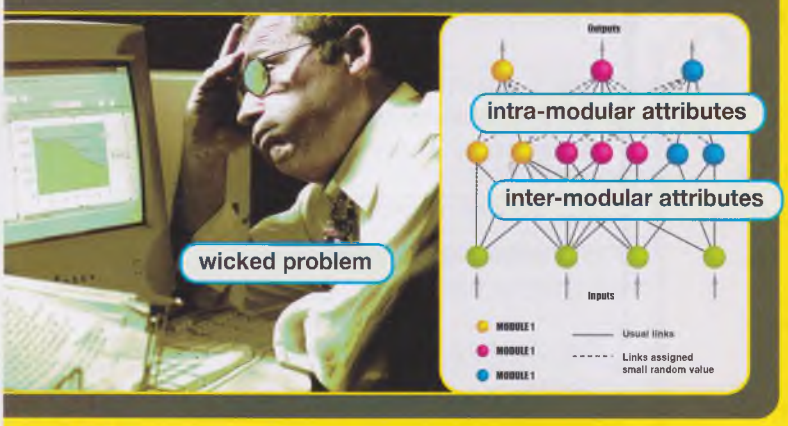
Every engineer worries about **wicked problems.** Without **stopping rules**, engineers often don't know whether problems are fixed or not. But I'd like to point out that these problems can be easily avoided. Programmers just need to use **call graphs** to depict **system structures**. These let the programmer eliminate problems before they even begin.

Some programmers are geniuses with code, but they're unfamiliar with call graphs. This is unfortunate, because these systems greatly **simplify** the design process. Software planning provides programmers with an **abstraction** of the final product. Abstract systems are described in terms of their **modularity**, **cohesion**, and **coupling**. Programmers can even make allowances for **information hiding**.

When the plan is finished, the programmer can examine the **inter-modular attributes** and the **intra-modular attributes.** Errors can be eliminated while the software's **complexity** is low. Then, engineers are less likely to encounter complex problems later on.







Get ready!

1 Before you read the passage, talk about these questions.

1 What does a call graph show?

2 How do call graphs help programmers avoid problems?

Reading

2 Read the blog. Then, choose the correct answers.

1 What is the purpose of the blog?

- **A** to compare different software planning methods
- **B** to explain the value of call graphs
- **C** to give solutions for common wicked problems
- **D** to describe the inter-modular attributes of a system

2 Which of the following is NOT a reason to use call graphs?

- **A** to avoid wicked problems
- **B** to examine intra-modular attributes
- **C** to eliminate problems at an early stage
- **D** to create stopping rules

3 Which is a part of abstraction?

- **A** modularity
- **B** minute details
- **C** increased complexity
- **D** stopping points

Vocabulary

3 Fill in the blanks with the correct words and phrases from the word bank.

WORD BANK

call graph information hiding simplify
stopping rule wicked problem

1 A(n) _____ can have multiple causes and may be difficult to solve.

2 A(n) _____ shows the basic structure of how a system will work.

3 Modules conceal information from each other in a process called _____.

4 A problem without a(n) _____ may be difficult or impossible to solve.

5 Use of systems and procedures can _____ complicated processes.

4 **Read the sentence pairs. Choose which word or phrase best fits each blank.**

1 **cohesion / coupling**

A _____ describes the strength of connections between modules.

B _____ is the connection between modules in a system.

2 **inter-modular attributes / intra-modular attributes**

A Characteristics of individual modules are _____.

B _____ describe the characteristics of an entire system.

3 **system structure / abstraction**

A A(n) _____ is the network of connections between modules.

B A(n) _____ ignores details.

4 **complexity / modularity**

A _____ is judged by the amount of time it would take to change something in a system.

B _____ indicates that a system is made up of smaller interconnected systems.

5 🎧 **Listen and read the blog again. How can programmers avoid wicked problems in software designs?**

Listening

5 🎧 **Listen to a conversation between an engineer and an intern. Mark the following statements as true (T) or false (F).**

1 __ The woman is having difficulty reading a call graph.

2 __ The woman suggests removing information from a design.

3 __ The man explains the importance of excluding details from a design.

7 🎧 **Listen again and complete the conversation.**

Intern: Actually there is. I'm **1** _____ _____ about a few things.

Engineer: Let's see if we can get that **2** _____ _____. What is confusing you?

Intern: Why do we need to **3** _____ - _____ _____ before adding the details?

Engineer: What do you mean?

Intern: Wouldn't it save time to **4** _____ _____ _____ as we create the design?

Engineer: No, you need to consider the purpose of an **5** _____.

Intern: An abstraction?

Engineer: Yes. Creating a design without details lets us find problems early. That way we can fix them **6** _____ _____ _____ is too complex. Does that make sense?

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

Is there anything ...

I don't see why ...

What you're not considering is ...

Student A: You are an engineer. Talk to Student B about:

- the software design process
- reasons for a particular process
- the value of particular design tools or elements

Student B: You are an intern. Talk to Student A about reasons for using a particular software design process.

Writing

9 **Use the conversation from Task 8 to complete the intern's notes.**

Design	Benefits
_____	provides a plan of what a final product will look like
abstraction	_____ _____
_____	_____ _____