Practical CM III: Best Configuration Management Practices

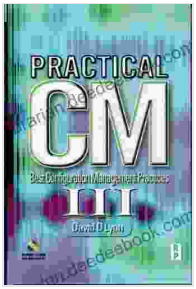
Configuration management (CM) is a process for managing the different versions of a software system. It helps to keep track of changes to the system, and to ensure that the system is always in a consistent state. CM is an essential part of software development, and can help to improve the quality, reliability, and maintainability of software systems.

There are many different CM practices, and the best practices for a particular system will depend on the size and complexity of the system, as well as the development process. However, some general best practices for CM include:

- **Version control**. Version control is a system for tracking changes to files. It allows developers to collaborate on changes to the system, and to roll back changes if necessary.
- **Source control**. Source control is a type of version control that is used to manage the source code for a software system. It allows developers to collaborate on changes to the code, and to track the history of the code.
- **Change management**. Change management is a process for managing changes to a software system. It helps to ensure that changes are made in a controlled and consistent manner.

Version control is an essential part of CM. It allows developers to collaborate on changes to the system, and to roll back changes if

necessary. There are many different version control systems available, including CVS, Subversion, and Git.



Practical CM III: Best Configuration Management Practices by Terrence McCloy

4.1 out of 5

Language	: English
File size	: 5571 KB
Text-to-Speech	: Enabled
Screen Reader	: Supported
Enhanced typesetting	: Enabled
Word Wise	: Enabled
Print length	: 305 pages
Lending	: Enabled

When choosing a version control system, it is important to consider the following factors:

- **The size and complexity of the system**. The size and complexity of the system will determine the features that you need in a version control system. For example, if you are developing a large and complex system, you may need a version control system that supports branching and merging.
- **The development process**. The development process will also determine the features that you need in a version control system. For example, if you are using a continuous integration process, you may need a version control system that supports automated builds.

Source control is a type of version control that is used to manage the source code for a software system. It allows developers to collaborate on changes to the code, and to track the history of the code. There are many different source control systems available, including CVS, Subversion, and Git.

When choosing a source control system, it is important to consider the following factors:

- **The size and complexity of the codebase**. The size and complexity of the codebase will determine the features that you need in a source control system. For example, if you are working on a large and complex codebase, you may need a source control system that supports branching and merging.
- **The development process**. The development process will also determine the features that you need in a source control system. For example, if you are using a continuous integration process, you may need a source control system that supports automated builds.

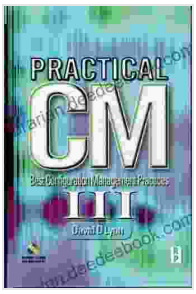
Change management is a process for managing changes to a software system. It helps to ensure that changes are made in a controlled and consistent manner. There are many different change management processes, but they all typically involve the following steps:

1. **Identify the change**. The first step in change management is to identify the change that needs to be made. This can be anything from a small bug fix to a major new feature.
2. **Plan the change**. Once the change has been identified, it is important to plan how the change will be made. This includes identifying the

resources that will be needed, and the timeline for the change.

3. **Implement the change**. Once the plan has been developed, the change can be implemented. This may involve making changes to the code, the documentation, or the testing process.
4. **Test the change**. Once the change has been implemented, it is important to test the change to ensure that it has been made correctly. This may involve running unit tests, integration tests, or system tests.
5. **Document the change**. Once the change has been tested and verified, it is important to document the change. This includes updating the documentation, and making a note of the change in the version control system.

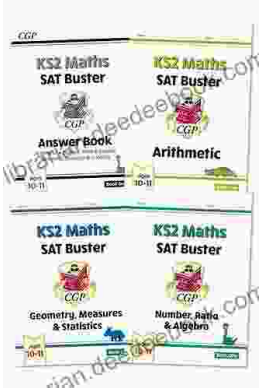
CM is an essential part of software development. It can help to improve the quality, reliability, and maintainability of software systems. By following the best practices for CM, developers can help to ensure that their software systems are always in a consistent state.



Practical CM III: Best Configuration Management Practices by Terrence McCloy

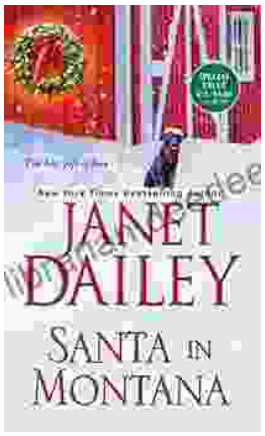
4.1 out of 5

Language	: English
File size	: 5571 KB
Text-to-Speech	: Enabled
Screen Reader	: Supported
Enhanced typesetting	: Enabled
Word Wise	: Enabled
Print length	: 305 pages
Lending	: Enabled



Supercharge Your Child's KS1 Maths Skills with the Ultimate SAT Buster (Comprehensive Guide for Parents)

As a parent, you want to provide your child with the best possible education. When it comes to mathematics, the Key Stage 1 (KS1) SATs (Standard Attainment Tests)...



Santa in Montana: Calder 11 - A Magical Destination for the Holidays

Nestled amidst the picturesque mountains of Montana, Calder 11 is a winter wonderland that transforms into a magical Christmas destination. As you...