Software Engineering by Ian Sommerville

Chapter 7: Design and Implementation

SUMMARY

Software design and implementation is the part of software engineering where a working software system is created. For simple systems, this process includes everything related to software engineering. But for larger systems, design and implementation is just one part of a bigger process that also includes things like figuring out what the software needs to do and checking if it works correctly.

To develop a system design from concept to detailed you need to:

- Understand and define the context and external interactions
- Develop a system architecture
- Specify the principal objects of the system
- Design Models
- And specify the interfaces

System Context: Structural model that demonstrates the other systems in the environment of the system.

Interaction Model: Dynamic model that shows how the system interacts with its environment as it is used.

When designing models, the level of detail may depend on the design process, for example:

- If Agile is used, a simple drawing on a whiteboard may suffice. Developers and Designers work closely with meetings.
- Plan driven models may require a more detailed model. design and implementation may be developed in different places

Structural Models: Objects and class relationships that describe a static structure of a system

Dynamic Models: A changing (dynamic) structore of the system and expected runtime behavior between objects of the system.

There are three key UML models that are useful for detailing architecture models:

- Subsystem Models: Shows logical groupings of objects using a form of class diagrams with enclosed objects on a package..
- Sequence Models: Shows the order of interactions for an object.
- State Machines: Shows how objects change their state in response to events.

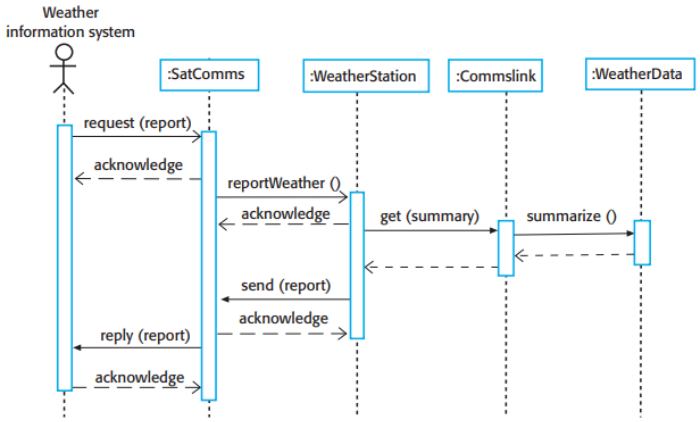

Figure 7.1 Sequence diagram showing the data collection

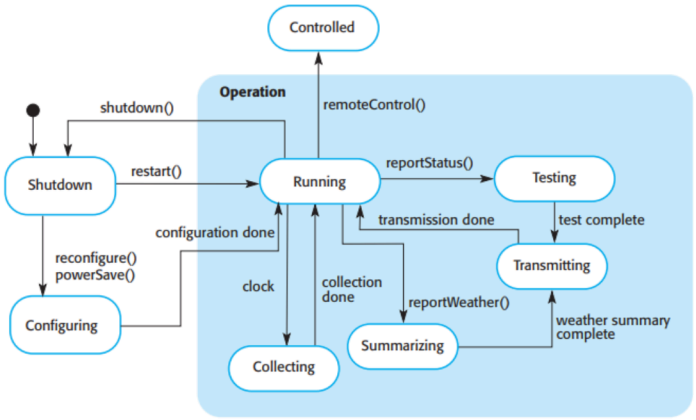

Figure 7.2 An example of a state diagram for a weather station

Design patterns

Description of a problem and the essence of its solution.

- The design patterns contain:
- A meaningful name
- A description of the problem on which the pattern will be applied
- A solution description with their relationships and responsibilities.
- A statement of the consequences, results and tradeoffs. This can help designers understand when to use the specified pattern and when not to.

Implementation Issues

Three important concepts of implementation on software engineering are:

- Reuse
- Configuration Management → Version Control (GIT)
- Host-Target Development → Production/Development Environments

Open Source

Main licenses: GPL, LGPL, BSD

GNU General Public License
Lesser General Public License
Berkeley Software Distribution