Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

After our discussion of scenario-based，class-based model and function model in Chapters 9 and 10, it's reasonable to ask, "Aren't those requirement modeling representations enough?"

The only reasonable answer is, "That depends."

For some types of software, the use case may be the only requirements modeling representation that is required. For others, an object-oriented approach is chosen and class-based models may be developed. But in other situations,complex application requirements may demand an examination of how an application behaves as a consequence of external events

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

In this chapter, Each of these modeling representations supplements the scenario-based ,class-based model and function model discussed in Chapters 9 and 10.

Behavioral modeling depicts the states of the system and its classes and the impact of events on these states.

Pattern-based modeling makes use of existing domain knowledge to facilitate requirements analysis.

WebApp requirements models are especially adapted for the representation of content, interaction, function, and configuration-related requirements.

11.1 CREATING A BEHAVIORAL MODEL

The behavioral model indicates how software will respond to external events or stimuli.To create the model, you should perform the following steps: (1) evaluate all use cases to fully understand

the sequence of interaction within the system, (2) identify events that drive the interaction sequence and understand how these events relate to specific objects, (3) create a sequence for each use case, (4) build a state diagram for the system or classes, and (5) review the behavioral model to verify accuracy and consistency. Each of these steps is discussed in the sections that follow.

11.2 IDENTIFYING EVENTS WITH THE USE CASE

An event is not the information that has been exchanged, but rather the fact that information has been exchanged.

By the use case scenario to indicate events. An actor should be identified for each event; the information that is exchanged should be noted, and any conditions or constraints should be listed.

Once all events have been identified, they are allocated to the

objects involved. Objects can be responsible for generating events (e.g., Homeowner generates the password entered event) or recognizing events that have occurred elsewhere (e.g., ControlPanel recognizes the binary result of the password compared event).

11.3 STATE REPRESENTATIONS

In the context of behavioral modeling, two different characterizations of states must be considered: (1) the state of each class as the system performs its function and (2) the state of the system as observed from the outside as the system performs its function.

Two different behavioral representations are discussed. The first indicates how an individual class changes state based on external events and the second shows the behavior of the software as a function of time.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

State Diagrams for Analysis Classes:

One component of a behavioral model is a UML state diagram that represents active states for each class and the events (triggers) that cause changes between these active states. Figure 11.1 illustrates a state diagram for the ControlPanel object in the SafeHome security function.

Although the active state model provides useful insight into the "life history" of an object, it is possible to specify additional information to provide more depth in understanding the behavior of an object. In addition to specifying the event that causes the transition to occur, you can specify a guard and an action. A guard is a Boolean condition that must be satisfied in order for the transition to occur. For example, the guard for the transition from the "reading" state to the "comparing" state in Figure 11.1 can be determined by examining the use case:

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

if (password input = 4 digits) then *compare* to stored password

In general, the guard for a transition usually depends upon the value of one or more attributes of an object. In other words, the guard depends on the passive state(the current status of an object's attributes, value of attribute) of the object.

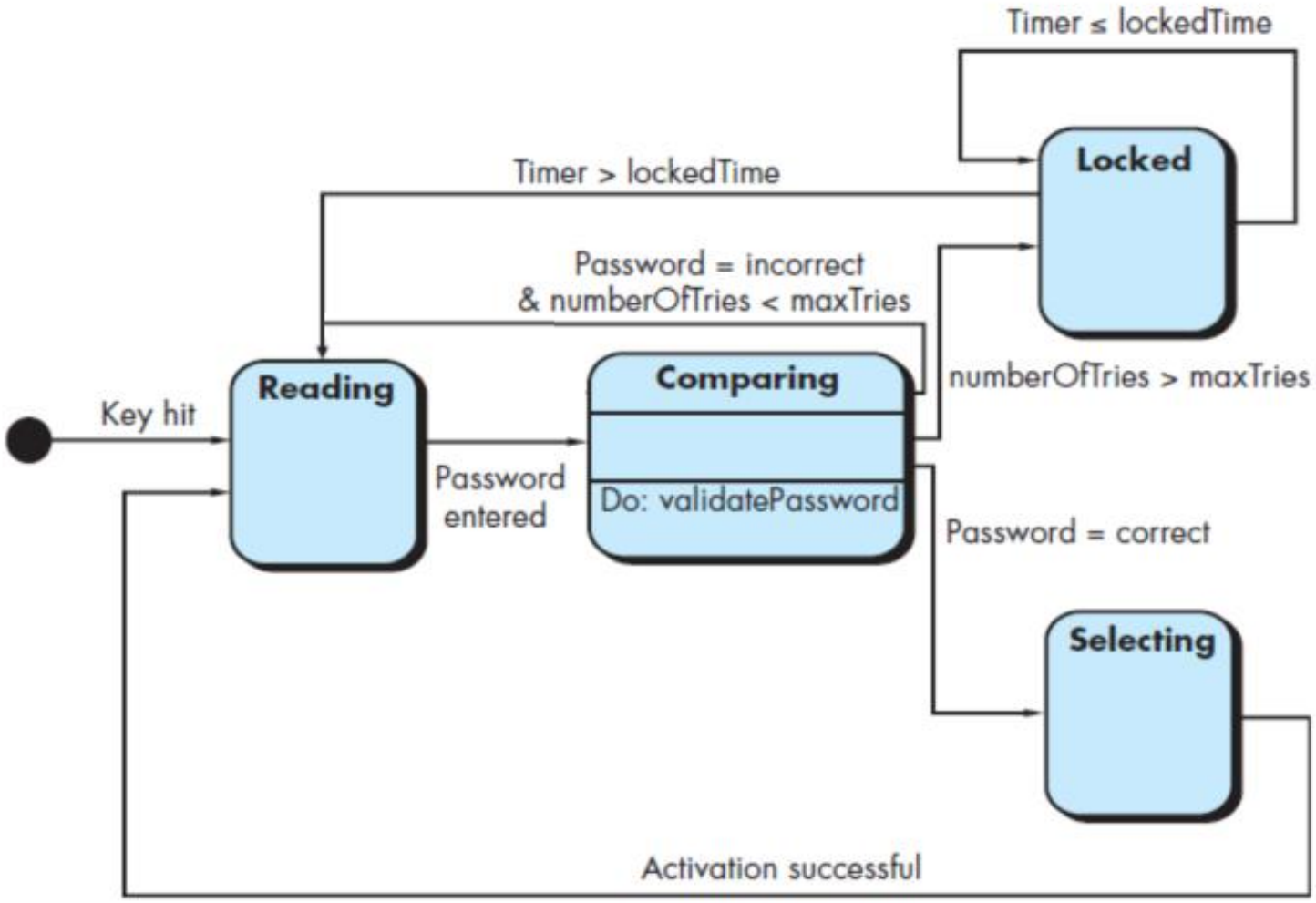
In the figure 11.1,

when the event "Password= incorrect &numberOftries < maxTries" is triggered, the state should be from Comparing to Reading. it is a wrong.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

FIGURE 11.1

State diagram for the ControlPanel class



Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

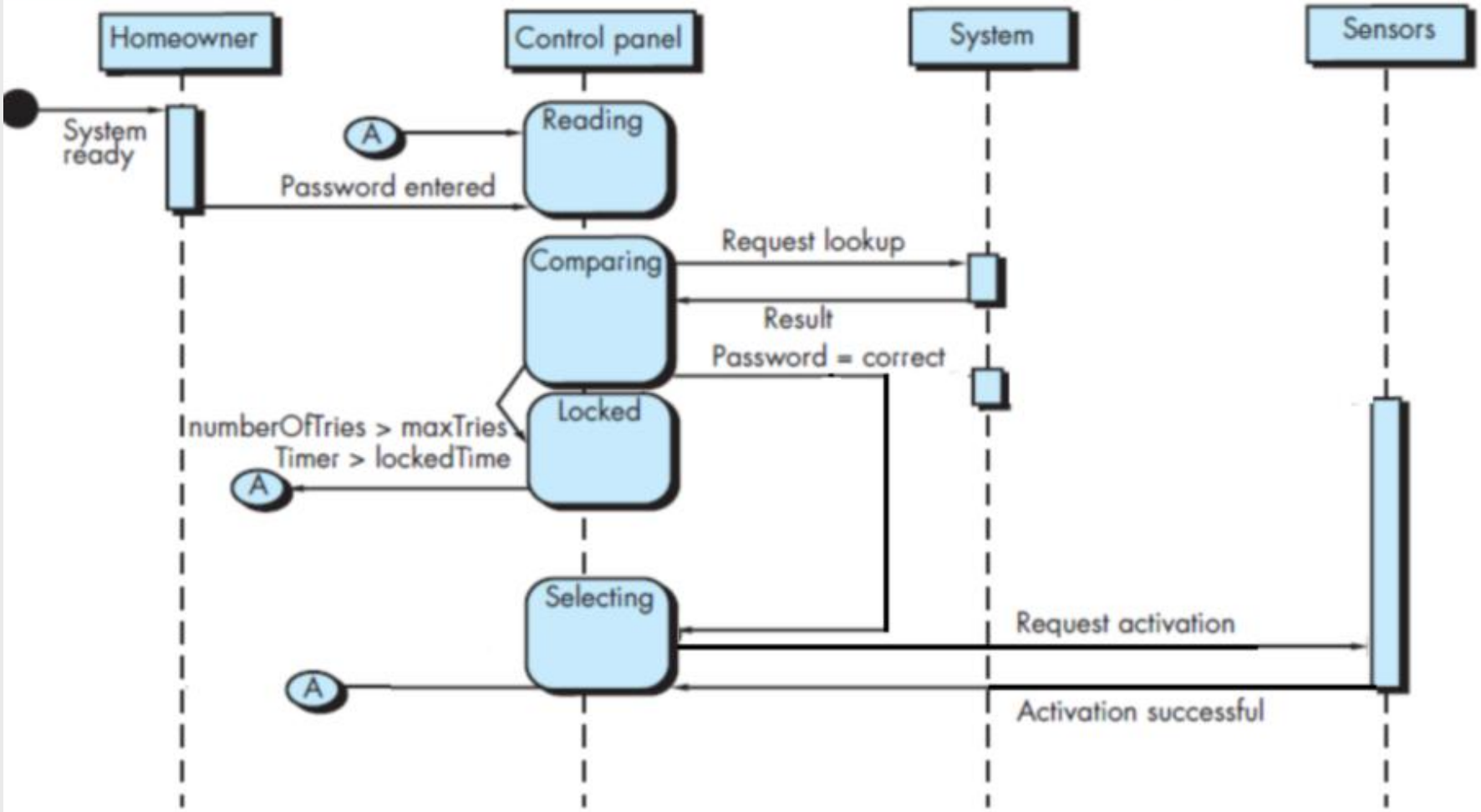
Sequence Diagrams:

The second type of behavioral representation, called a sequence diagram,indicates how events cause transitions from object to object. Once events have been identified by examining a use case, the modeler creates a sequence diagram—a representation of how events cause flow from one object to another as a function of time. In essence, the sequence diagram is a shorthand version of the use case. It represents key classes and the events that cause behavior to flow from class to class.

Once a complete sequence diagram has been developed, all of the events that cause transitions between system objects can be collated into a set of input events and output events (from an object). This information is useful in the creation of an effective design for the system to be built. Figure 11.2 illustrates a partial sequence diagram for the SafeHome security function(use case of activation system)

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

Request activation Sequence diagram (partial) for the *SafeHome* security function Figure 11.2



Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

11.4 PATTERNS FOR REQUIREMENTS MODELING

Software patterns are a mechanism for capturing domain knowledge in a way that allows it to be reapplied when a new problem is encountered. In some cases, the domain knowledge is applied to a new problem within the same application domain. In other cases, the domain knowledge captured by a pattern can be applied by analogy to a completely different application domain.

In Chapter 8, we introduced the concept of analysis patterns and indicated that these patterns represent something (e.g., a class, a function, a behavior) within the application domain that can be reused when performing requirements modeling for an application within a domain. Analysis patterns are stored in a repository so that members of the software team can use search facilities to find and reuse them. Once an appropriate pattern is selected, it is integrated into the requirements model by reference to the pattern name.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

SafeHome

Discovering an Analysis Pattern

The scene: A meeting room, during a team meeting.

The players: Jamie Lazar, software team member; Ed Robbins, software team member; Doug Miller, software engineering manager

The conversation:

Doug: How are things going with modeling the requirements for the sensor network for the *SafeHome* project?

Jamie: Sensor work is a little new to me, but I think I'm getting a handle on it.

Doug: Is there anything we can do to help you with that?

Jamie: It would be a lot easier if I'd built a system like this before.

Doug: True.

Ed: I was thinking this is a situation where we might be able to find an analysis pattern that would help us model these requirements.

Doug: If we can find the right pattern, we'd avoid reinventing the wheel.

Jamie: That sounds good to me. How do we start?

Ed: We have access to a repository that contains a large number of analysis and design patterns. We just need to search for patterns with intents that match our needs.

Doug: That seems like that might work. What do you think, Jamie?

Jamie: If Ed can help me get started, I'll tackle this today.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

11.5 REQUIREMENTS MODELING FOR WEB AND MOBILE APPS

Developers of Web and mobile applications are often skeptical when the idea of requirements analysis is suggested. "After all," they argue, "our development process must be agile, and analysis is time consuming. It'll slow us down just when we need to be designing and building the application."

Requirements analysis does take time, but solving the wrong problem takes even more time. The question for every WebApp and mobile developer is simple—are you sure you understand the requirements of the problem or product? If the answer is an unequivocal yes, then it may be possible to skip requirements modeling, but if the answer is no, then requirements modeling should be performed.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

11.5.1 How Much Analysis Is Enough?（read by yourselves）

The degree to which requirements modeling for Web and mobile apps is emphasized depends on the following size-related factors: (1) the size and complexity of the application increment, (2) the number of stakeholders (analysis can help to identify conflicting requirements coming from different sources), (3) the size of the app development team, (4) the degree to which members of the team have worked together before (analysis can help develop a common understanding of the project), and (5) the degree to which the organization's success is directly dependent on the success of the application.

The converse of the preceding points is that as the project becomes smaller, the number of stakeholders fewer, the development team more cohesive, and the application less critical, it is reasonable to apply a more lightweight analysis approach.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

Although it is a good idea to analyze the problem or product requirements before beginning design, it is not true that all analysis must precede all design. In fact, the design of a specific part of the application only demands an analysis of those requirements that affect only that part of the application. As an example from SafeHome , you could validly design the overall website aesthetics (layouts,color schemes, etc.) without having analyzed the functional requirements for e-commerce capabilities. You only need to analyze that part of the problem that is relevant to the design work for the increment to be delivered.

11.5.2 Requirements Modeling Input（read by yourselves）

An agile version of the generic software process discussed in Chapter 5 can be applied when Web or mobile apps are engineered.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

Information is the input for requirements modeling and Information is gathered by communication activity that identifies stakeholders and user categories, the business context, defined informational and applicative goals, general product requirements, and usage scenarios.

But for web/mobile apps, this function seemed relatively clear and was described in some detail as part of a use case (Section 9.2.1). However, a reexamination of the use case might uncover information that is missing, ambiguous, or unclear.

Some aspects of this missing information would naturally emerge during the design. Examples might include the specific layout of the function buttons, their aesthetic look and feel, the size of snapshot views, the placement of camera views and the house floor plan, or even minutiae such as the maximum and minimum length of

passwords. Some of these aspects are design decisions (such as the layout of the buttons) and others are requirements (such as the length of the passwords) that don't fundamentally influence early design work.

But some missing information might actually influence the overall design itself and relate more to an actual understanding of the requirements. For example:

Q1: What output video resolution is provided by SafeHome cameras?

Q2: What occurs if an alarm condition is encountered while the camera is being monitored?

Q3: How does the system handle cameras that can be panned and zoomed?

Q4: What information should be provided along with the camera view? (For example, location? time/date? last previous access?)

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

11.5.3 Requirements Modeling Output

Requirements analysis provides a disciplined mechanism for representing and evaluating application content and function, the modes of interaction that users will encounter, and the environment and infrastructure in which the WebApp or mobile app resides.

Each of these characteristics can be represented as a set of models that allow application requirements to be analyzed in a structured manner. While the specific models depend largely upon the nature of the application, there are five main classes of models:

- ***Content model*** **—identifies the full spectrum of content to be provided by the application. Content includes text, graphics and images, video, and audio data.**
- ***Interaction model*** **—describes the manner in which users interact with the app.**

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

- ***Functional model*** **—defines the operations that will be applied to manipulate content and describes other processing functions that are independent of content but necessary to the end user.**
- ***Navigation model*** **—defines the overall navigation strategy for the app.**
- ***Configuration model*** **—describes the environment and infrastructure in which the app resides.**

You can develop each of these models using a representation scheme (often called a "language") that allows its intent and structure to be communicated and evaluated easily among members of the engineering team and other stakeholders.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

11.5.4 Content Model

The content model contains structural elements that provide an important view of content requirements for an application.These structural elements encompass *content objects* —user-visible entities that are created or manipulated as a user interacts with the app through a browser or a mobile device.

A *content object* might be a textual description of a product, an article describing a news event, a graphical representation of retrieved data (e.g., stock price as a function of time), an action photograph taken at a sporting event, a user's response on a discussion forum, an animated representation of a corporate logo, a short video of a speech, or an audio overlay for a collection of presentation slides.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

The content objects might be stored as separate files or obtained dynamically from a database. They might be embedded directly into Web pages, displayed on the screen of a mobile device. In other words, a content object is any item of cohesive information that is to be presented to an end user.

Content objects can be determined directly from use cases by examining the scenario description for direct and indirect references to content.

For example, a WebApp that supports SafeHome is established at *www.safehomeassured.com*. A use case, *Purchasing Select SafeHome Components*, describes the scenario required to purchase a SafeHome component and contains the sentence:

I will be able to get descriptive and pricing information for each product component.

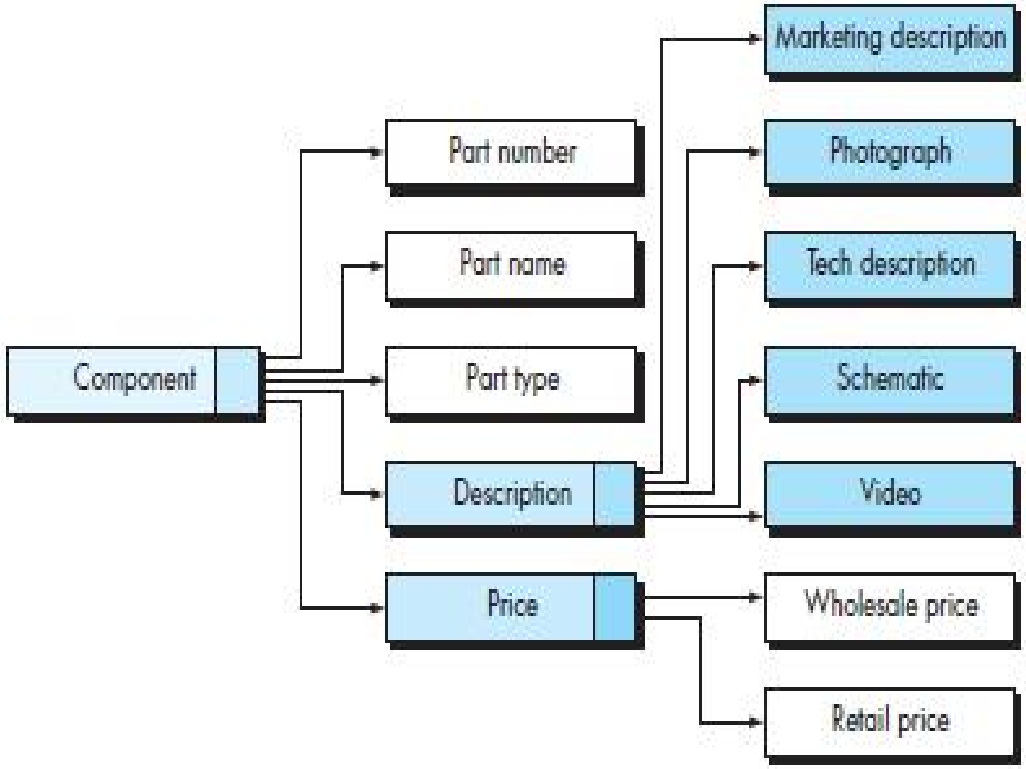
Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

The content model must be capable of describing the content object *Component*. In many instances, a simple list of content objects, coupled with a brief description of each object, is sufficient to define the requirements for content that must be designed and implemented. However, in some cases, the content model may benefit from a richer analysis that graphically illustrates the relationships among content objects and/or the hierarchy of content maintained by a WebApp.

For example, consider the data tree created for a www.safehomeassured .com *component* shown in Figure 11.5 .The tree represents a hierarchy of information that is used to describe a component. Simple or composite data items (one or more data values) are represented as unshaded rectangles. Content objects are represented as shaded rectangles. In the figure, *description* is

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

FIGURE 11.5 Data tree for a **www.safehomeassured.com** component



defined by five content objects (the shaded rectangles). In some cases, one or more of these objects would be further refined as the data tree expands.

A data tree can be created for *any content that is composed of multiple content objects and data items*. The data tree is developed in an effort to define hierarchical relationships among content objects and to provide a means for reviewing content so that omissions and inconsistencies are uncovered before design commences. In addition, the data tree serves as the basis for content design.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

11.5.5 Interaction Model for Web and Mobile Apps

The vast majority of Web and mobile apps enable a "conversation" between an end user and application functionality, content, and behavior.

This conversation can be described using an interaction model that can be composed of one or more of the following elements: (1) use cases, (2) sequence diagrams, (3) state diagrams, and/or (4) user interface prototypes.

In many instances, a set of use cases is sufficient to describe the interaction at an analysis level (further refinement and detail is introduced during design).However, when the sequence of interaction is complex and involves multiple analysis classes or many tasks, it is sometimes worthwhile to depict it using a more rigorous diagrammatic form.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

Although it can be argued that the creation of a user interface prototype is a design activity, it is a good idea to perform it during the creation of the analysis model. The sooner that a physical representation of a user interface can be reviewed, the higher the likelihood that end users will get what they want.

it is best to create the interface prototype using such tools(exercise: find a tool that creates the interface prototype).The prototype should implement the major navigational links and represent the overall screen layout in much the same way that it will be constructed.

11.5.6 Functional Model

Many WebApps deliver a broad array of computational and

manipulative functions that can be associated directly with content (either using it or producing it) and that are often a major goal of user-WebApp interaction.

Mobile apps tend to be more focused and provide a more limited set of computational and manipulative functions. Regardless of the breadth of functionality, functional requirements should be analyzed, and when necessary, modeled.

The *functional model* addresses two app processing elements, each representing a different level of procedural abstraction: (1) user observable functionality that is delivered by the app to end users, and (2) the operations contained within analysis classes that implement behaviors associated with the class.

User-observable functionality encompasses any processing

functions that are initiated directly by the user. For example, a financial mobile app might implement a variety of financial functions (e.g., computation of mortgage payment).These functions may actually be implemented using operations within analysis classes, but from the point of view of the end user, the function (more correctly, the data provided by the function) is the visible outcome.

At a lower level of procedural abstraction, the requirements model describes the processing to be performed by analysis class operations. These operations manipulate class attributes and are involved as classes collaborate with one another to accomplish some required behavior.

Regardless of the level of procedural abstraction, the UML activity(Swimlane) diagram can be used to represent processing details.

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

11.5.7 Configuration Models for WebApps

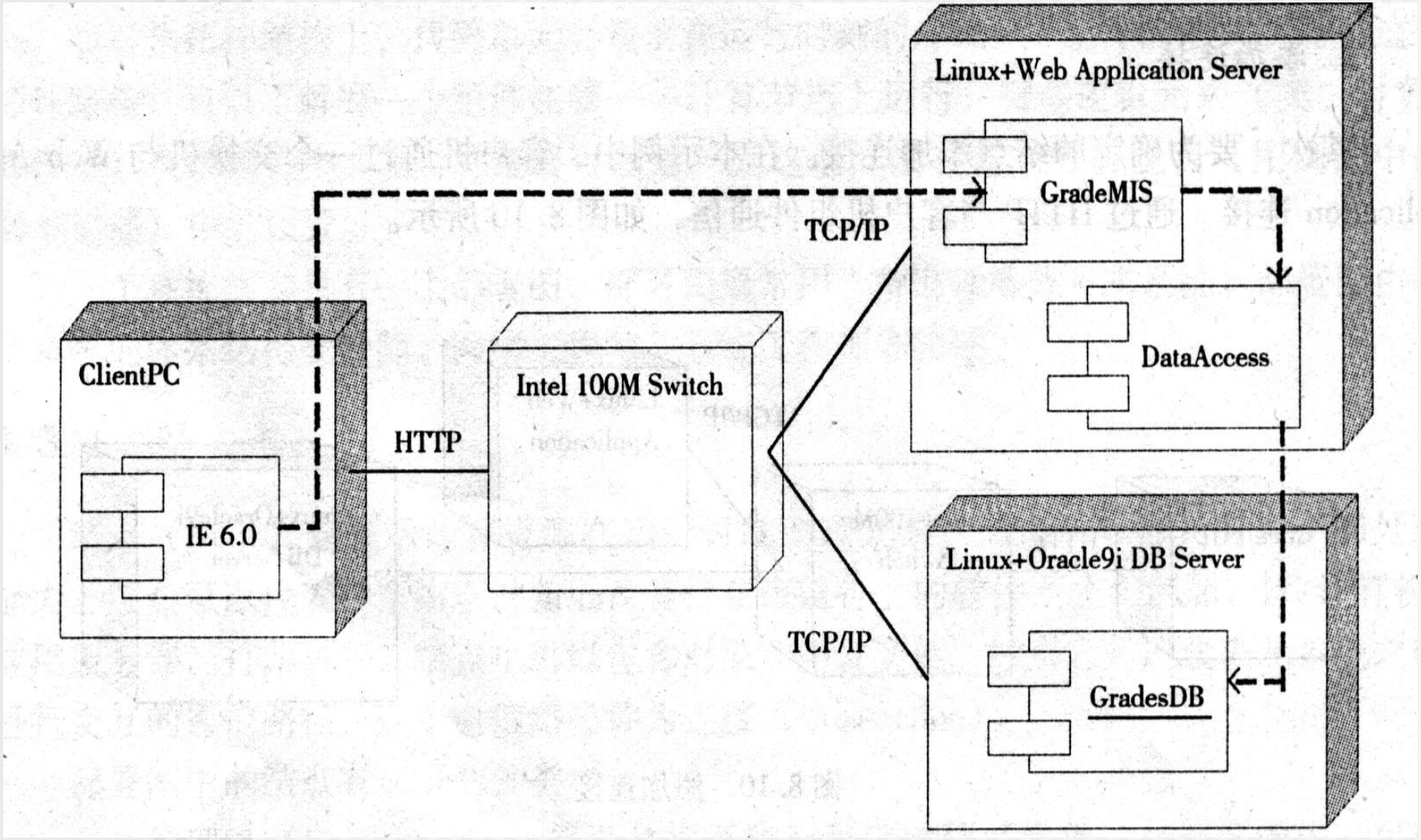
In some cases, the configuration model is nothing more than a list of server-side and client-side attributes. However, for more complex apps, a variety of configuration complexities (e.g., distributing load among multiple servers, caching architectures,remote databases, multiple servers serving various objects) may have an impact on analysis and design.

The UML *deployment diagram* can be used in situations in which complex configuration architectures must be considered.

11.5.8 Navigation Modeling

In most mobile applications that reside on smartphone platforms, navigation is generally constrained to relatively simple button lists and icon-based menus. In addition, the depth of navigation (i.e., the

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS



number of levels into the hypermedia hierarchy) is relatively shallow. For these reasons, navigation modeling is relatively simple.

For WebApps and an increasing number of tablet-based mobile applications, navigation modeling is more complex and often considers how each user category will navigate from one WebApp element (e.g., content object) to another. The mechanics of navigation are defined as part of design. At this stage, you should focus on overall navigation requirements. The following questions should be considered:

- Should certain elements be easier to reach (require fewer navigation steps) than others? What is the priority for presentation?
- Should certain elements be emphasized to force users to navigate in their direction?

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

- How should navigation errors be handled?
- Should navigation to related groups of elements be given priority over navigation to a specific element?
- Should navigation be accomplished via links, via search-based access, or by some other means?
- Should certain elements be presented to users based on the context of previous navigation actions?
- Should a navigation log be maintained for users?
- Should a full navigation map or menu (as opposed to a single "back" link or directed pointer) be available at every point in a user's interaction?
- Should navigation design be driven by the most commonly expected user behaviors or by the perceived importance of the defined WebApp elements?

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

- Can a user "store" his previous navigation through the WebApp to expedite future usage?
- For which user category should optimal navigation be designed?
- How should links external to the WebApp be handled? Overlaying the existing browser window? As a new browser window? As a separate frame?

These and many other questions should be asked and answered as part of navigation analysis. You and other stakeholders must also determine overall requirements for navigation. For example, will a "site map" be provided to give users an overview of the entire WebApp structure? Can a user take a "guided tour" that will highlight the most important elements (content objects and functions) that are available?

Chapter 11 REQUIREMENTS MODELING: BEHAVIOR, PATTERNS, AND WEB/MOBILE APPS

11.6 SUMMARY

Behavioral modeling during requirements analysis depicts dynamic behavior of the software. The behavioral model uses input from scenario-based or classbased elements to represent the states of analysis classes and the system as a whole. To accomplish this, states are identified, the events that cause a class (or the system) to make a transition from one state to another are defined, and the actions that occur as transition is accomplished are also identified. State diagrams and sequence diagrams are the notation used for behavioral modeling.

Requirements modeling for mobile applications and WebApps can use most, if not all, of the modeling elements discussed. However, these elements are applied within a set of specialized models that address content, interaction,function, navigation, and the configuration in which the mobile app or WebApp resides.