APPENDIX

Patterson Superstore

CHAPTER 1: PATTERSON SUPERSTORE

This course will introduce many new concepts regarding object-oriented analysis and design. To make these concepts more relevant and understandable, we will apply the concepts introduced in each chapter to a fictitious company called Patterson Superstore.

Patterson is a retail chain established in Pittsburgh, PA, in 1985. The chain has expanded from four stores in the Pittsburgh area to a well-known national presence.

Initially Patterson sold diversified merchandise, including a variety of clothing, toys, housewares, sporting goods, and electronics. However, during the 2000s, it expanded its offerings into groceries and pharmacies and began branding itself as a superstore.

In 2008, Patterson's extended its pharmacy services by offering free blood pressure and cholesterol screening and affordable flu shots. From the immediate success of these services, the VP of the pharmacy division, Max Ross, recognized a growth opportunity and expanded the pharmacy offerings to include in-store health clinics. Services offered include diagnosis and treatment of minor illnesses (colds, strep, flu), skin conditions (impetigo, chicken-pox, shingles), injuries (burns, cuts), and vaccinations (tetanus, HPV). Additionally, wellness services such as school and sport physicals are available. The in-store medical clinics are staffed by nurses and physician assistants or nurse practitioners and operate on an appointment or walk-in basis.

Superstores, such as Patterson, enjoy several advantages over medical centers in offering these services.

1. Since superstores have multiple types of income streams, delays in Medicaid and other types of insurance reimbursement are significantly less problematic than for medical centers that lack several revenue streams.
2. Superstores also enjoy reduced overhead cost while still generating the same copay revenue collected by medical centers. A copay is still a copay.
3. Patients like the convenience of one-stop shop with a seamless care, diagnosis, prescription fulfillment process.
4. Personnel costs tend to be lower than medical centers because the clinics are overseen by a nurse practitioner or physician assistant with nurses providing much of the care.

Max Ross has identified an additional opportunity related to the health clinic segment. Currently, Patterson uses a mobile application to facilitate prescription order and refill, notification, and auto-refill services. This service is widely used by Patterson's client base, and Patterson has leveraged this mobile app to gain an advantage over less technically advanced competitors.

Clients now want to use this technology to access health clinic services. Max Ross wants to use this opportunity to position Patterson as a leader in the use of technology use for clinic access. The system that he envisions would enable real-time communication with medical

personnel (audio, video, and text), mobile appointment scheduling, tele-health assessment, and diagnosis of minor problems through video house calls. In addition, Patterson desires data analytic and tracking capabilities.

This project would build on existing expertise within the IT department. The IT department staff designed, developed, and maintains the sophisticated prescription fulfillment system already in place at Patterson and can leverage that experience in creating the proposed system.

The IT department has enthusiastically moved toward RAD and Object-Oriented Methodologies and views familiarity with these methodologies as a strategic advantage. This project would lend itself to such development and thus increase expertise in this area.

Based on the reading above and the criteria for selecting a methodology that you learned in Chapter 1, what methodology would you recommend?

Information Systems projects are approved at Patterson by a steering committee that consists of high-level division representatives (such as Max) and IT division leaders. There are always many projects to consider and to prioritize. Max Ross plans to present a systems request that outlines his idea more fully at the next steering committee meeting. In this document, he will explain the business need, opportunity, and business value of the proposed system.

One problem that Max envisions is pushback from other segments of the organization who feel that the medical clinic concept is not part of the Patterson mission. However, the pharmacy and health clinic area has been the most profitable division for the last two years, and Max plans to explain how this project would further increase Patterson's profitability by outlining the expected financial benefits of the new systems.

How might you address the pushback?

In preparation for this meeting, Max is working with his team to develop high-level requirements for the proposed system and is also identifying issues and constraints related to his proposed system.

Requirements:

- Defined level of service offerings
- Data analytics and tracking
- Viewable wait time in real time
- Walk-in clinic and automated response system for scheduling appointments
- Referral information for conditions beyond the scope of the clinics service
- Intuitive Auto response with periodic human monitoring to avoid unhandled clients
- Video conferencing capability
- Limited diagnostic capabilities for call-ins

Add to the list of requirements based on what you have read and your experience with medical care.

Problems currently experienced within the clinic that the system should address include:

- Patients want to be able to schedule treatment but are often required to be evaluated prior to a treatment appointment being scheduled

- How to deal with referring items cannot be treated at the clinic
- Staff cutbacks have caused delays in responsiveness

Constraints:

- Must be HIPPA compliant
- Security is vital
- Staffing regulations
- Highly regulated field

Are there other constraints that Max and his group have not identified? Do you have any concerns about this project?

In Chapter 2, we will look more closely at the completed system request that Max Ross and his team developed. We will also look at the feasibility analysis that accompanies the request and see how the project is staffed and managed.

As we progress through the text, learning how Patterson navigates through the systems analysis and design process will help us to understand real-world implementation of the concepts presented.

CHAPTER 2: PATTERSON SUPERSTORE

In this segment of the Patterson Superstore case, we look more closely at the integrated health clinic delivery system that Max Ross envisions, which will enable real-time communication and scheduling for Patterson's health clinics. In addition, we will examine the completed system request that Max Ross and his team developed. Finally, we will review the feasibility analysis that accompanies the request and see how the project is staffed and managed.

Project Identification and Systems Request

At Patterson, potential projects are reviewed during quarterly steering committee meetings where participants from IT and the major business departments decide which projects to approve. Approval is based on business need and on how well the project advances the strategic objectives of the organization. Using the systems request template (Figure 2-1), Max Ross prepared a system request for the Integrated Health Clinic Delivery System (Figure 2-A).

The business need is to accommodate clients' desire to electronically access health clinic services. Doing so will heighten Patterson's competitive advantage, improve customer service, and increase the effectiveness of clinic offerings. Business need does not focus on the technology itself but instead on business elements, such as customer service, competitiveness, and efficacy. At this juncture, business requirements are described at a high level of detail. Max's vision for the requirements includes:

- Mobile appointment scheduling
- Real-time communication with medical personnel (audio, video, and text)
- Tele-health assessment and diagnosis of minor problems through video house calls
- Data analytic and tracking capabilities

Business value describes how the requirements will affect the business. Intangible business value will come from the increased satisfaction of current clinic customers and the enhanced recognition of value-added aspects of Patterson's clinical services. The proliferation of mobile applications and the growing interest of consumers in having larger and more convenient roles in their own healthcare further enhance the business value of this project. Max expects that the system will increase the number of clinic clients by offering convenient scheduling and service. This increase is projected to subsequently raise prescription and non-prescription sales due to the upsurge of foot traffic in the clinics and stores. Market research indicates that customers are seeking convenience in scheduling health appointments and that there is growing frustration with the requirement of face-to-face visits for routine diagnosis. Based on current clinic usage and the type of services currently requested, many customers do not utilize available clinic services due to wait times and scheduling conflicts. Max estimates that approximately 5 percent of potential service income is currently lost. A more convenient system could increase existing customer base service income as well as generating new clinic customers.

Feasibility Analysis

After reviewing the submitted systems request, the steering committee ranked the project as a high priority. Kelly Herman, a senior systems analyst, was assigned to work with Max to study the feasibility of the Integrated Health Clinic Delivery System. Kelly had been the team lead for the prescription order notification and auto refill mobile app project and was eager to develop further mobile services. Kelly and Max worked closely to develop the feasibility analysis below based on the technical, economic, and organization perspectives of the project.

System Request—Integrated Health Clinic Delivery System

Project sponsor: Max Ross, Vice President of Pharmacy Services

Business Need: This project has been initiated to integrate health clinic services by providing real-time electronic communication and scheduling for Patterson Superstore health clinics.

Business Requirements:

- Mobile appointment scheduling
- Real-time communication with medical personnel (audio, video, and text)
- Tele-health assessment and diagnosis of minor problems through video house calls
- Data analytic and tracking capabilities

Business Value:

We expect this integrated health clinic delivery system to lead to improved customer satisfaction and increased brand recognition due to its first mover advantage and increased convenience for clinic clients. Implementation of this system is also expected to boost in-pharmacy sales due to increased foot traffic in stores.

Conservative estimates of tangible value to the company per clinic include:

- $375,000 (75 percent of $500,000) in clinic services from new customers
- $750,000 (75 percent of $1,000,000) in clinic services from existing customers
- $50,000 in pharmacy sales from increase foot traffic in stores

Special Issues or Constraints:

- The Pharmacy Department views this as a strategic system that will add value to the current business model and will also provide customers with increased convenience and satisfaction.
- In order to gain first mover advantage, the system should be implemented in phases with the appointment scheduling piece in place within six months from the approval date.
- Increased staffing will be needed to operate the new system from both the technical and business operations aspect.

FIGURE 2-A Systems Request

Technical Feasibility

Technically, this project carries a low level of risk due to the expertise developed in the previous mobile application project. The IT department staff designed, developed, and maintains the sophisticated prescription fulfillment system already in place at Patterson and can leverage that experience in creating the proposed system. The IT department has enthusiastically moved toward Rapid Application Development and considers familiarity with these methodologies as a strategic advantage. This project would lend itself to RAD development and thus is expected to further increase proficiency in this area. The project size is considered medium risk because the project team will include fewer than ten people. User involvement will be required for proof of concept, testing, and requirements determination.

Economic Feasibility

Economic feasibility, based on the cost benefit analysis income shown in Figure 2-B, shows that this project would significantly add to Patterson's bottom line. While the development costs would be a one-time expenditure (with subsequent maintenance), the operating costs would be incurred at each clinic. However, as Figure 2-B indicates, even allocating total costs including development to an individual clinic, the clinic would return a profit in the first year (using a conservative estimate of income in the first year). Estimating a modest increase of 5 percent per year yields substantial increases in each following year. Intangible costs and benefits include increased satisfaction of current clinic customers and enhanced recognition of the ease of using Patterson's clinical services.

Income per clinic	2015	2016	2017
Clinic services from new customers	$375,000	$393,750.00	$413,437.50
Clinic services from existing customers	$750,000	$787,500.00	$826,875.00
Increased pharmacy sales	$50,000	$52,500.00	$55,125.00
TOTAL BENEFITS:	**$1,125.000**	**$1,181,250**	**$1,240,313**
Cost			
Labor: Analysis and design	$60,000	**0**	**0**
Labor: Implementation	$120,000	**0**	**0**
Staff training	$7,000	**0**	**0**
Office space and equipment	$2,000	**0**	**0**
Software	$10,000	**0**	**0**
Hardware	$35,000	**0**	**0**
TOTAL DEVELOPMENT COSTS:	**$234,000**	**0**	**0**
Labor: Computer operations	$50,000	$52,000	$54,000
Labor: Customer support	$45,000	$47,000	$49,000
Labor: Management oversight	$65,000	$67,000	$69,000
Labor: 3 staff	$90,000	$96,000	$102,000
Software upgrade/licensing	0	$4,000	$4,000
Hardware upgrades	0	$3,000	$3,000
User training	$2,000	$1,000	$1,000
Connectivity/Communication charges	$30,000	$30,000	$30,000
Promotional expenses	$50,000	$30,000	$30,000
TOTAL OPERATIONAL COSTS	**$332,000**	**$330,000**	**$342,000**
TOTAL COSTS	**$566,000**	**$330,000**	**$342,000**
TOTAL PROJECT BENEFITS/COST	**$559,000**	**$851,250**	**$898,313**

FIGURE 2-B Cost Benefit Analysis

Organizational Feasibility

From an organizational perspective, this project has low risk. The goals of the system to enhance competitive advantage, improve customer service, and increase the effectiveness of clinic offerings are aligned with the senior management's goal of increasing sales for Patterson Superstore. The project has a project champion, Max Ross, VP of Pharmacy Services, who is well-positioned to sponsor this project and to educate the rest of the senior management team to the benefits of the project. To date, much of senior management is aware of and support the initiative. Since health clinic clients have led to this proposal through requests for a more integrated and convenient health clinic system, user acceptance is expected to be high. Given the increased sales potential, store managers should be willing to accept the system.

Project Selection

Based on the strong profit potential of this project, the steering committee selected the Integrated Health Clinic Delivery System for funding and development. Because the first phase of the project was scheduled for implementation six months after approval, Max and Kelly were apprehensive about the short time frame. They quickly set to work finding a project manager to form a team and develop a project schedule. Ruby Neiley was chosen to manage the project due to her management of the prescription fulfillment project that was completed on time and within budget. In addition, Ruby has experience in leading phased development projects.

The system was approved based on implementation in phases. The phased development-based methodology apportions an overall system into a series of versions that are developed

sequentially. Phased development-based methodologies quickly put a useful system into the hands of the users. Because users begin to work with the system sooner, they are more likely to identify important additional requirements sooner than with structured design. The time boxing technique was chosen in conjunction with phased development to control scope and scheduling.

Time boxing steps include:

1. Set the date for system delivery.
2. Prioritize the functionality that needs to be included in the system.
3. Build the core of the system (the functionality ranked as most important).
4. Postpone functionality that cannot be provided within the time frame.
5. Deliver the system with core functionality.
6. Repeat steps 3 through 5 to add refinements and enhancements.

Since the appointment scheduling portion of the system needs to be in place within six months from the approval date, the version 1 system delivery time is set. While the appointment scheduling portion of the project is only one of the requirements, it is the most requested from customers. Delivery of this phase to the clinic clients should increase satisfaction and convenience for customers and prepare them for subsequent versions of the envisioned system.

In the upcoming analysis phase, the overall system concept will be further defined and the team will categorize the requirements into a series of versions.

Project Effort Estimation

One of Ruby's project management duties was to estimate the project's effort and schedule. Using the Use Case Point Worksheet (see Figure 2-15), Ruby estimated the effort to create the new system using the following steps.

1. Ruby and Max identified the business processes that the system would support and the users who would interact with the system. Then they sorted different user types into actors and arranged the business processes into use cases. The next step was to classify each actor and use case as being simple, average, or complex. In the case of the actors, the existing Pharmacy System had a well-defined API. As such it was classified as a simple actor. Three average actors include interaction with the web, mobile, and patient database. The Customer, Medical Staff, and Clinic Staff actors were classified as being complex. This gave an Unadjusted Actor Weight Total Value of 14.
2. Alec and Margaret classified each use case based on the number of transactions the use case had to handle. For the Mobile Appointment Scheduling (Version 1), there was one simple use case (Confirm Appointment), one average use case (Determine Suitability), and one complex use case (Make Appointment). Based on these, a value of 30 to the Unadjusted Use Case Weight Total was computed.
3. Ruby computed a value of 44 for the Unadjusted Use Case Points.
4. She rated each of the technical complexity factors, rated each of the environmental factors, and computed the values for TCF and EF.
5. Using the Unadjusted Use Case Points and the TCF and EF values, Ruby calculated a value of 52.73 for Adjusted Use Case Points.
6. Based on the decision rule for determining whether to use 20 or 28 as the value of the person hours multiplier, Ruby used 20. Using these figures, Ruby estimated the effort for the project to be 1,054.6 person hours. This equates to about 6.59 person months (1,318.2/160). In other words, it would take a single person working full time about 6½ months to complete the project.

Unadjusted Actor Weighting Table:

Actor Type	Description	Weighting Factor	Number	Result
Simple	External System with well-defined API	1	1	1
Average	External System using a protocol-based interface, (e.g., HTTP, TCT/IP, or a database)	2	2	4
Complex	Human	3	3	9
			Unadjusted Actor Weight Total (UAW)	**14**

Unadjusted Use Case Weighting Table:

Use Case Type	Description	Weighting Factor	Number	Result
Simple	1–3 transactions	5	1	5
Average	4–7 transactions	10	1	10
Complex	>7 transactions	15	1	15
			Unadjusted Use Case Weight Total (UUCW)	**30**

Unadjusted use case points (UUCP) = UAW + UUCW 44 = 14 + 30

Technical Complexity Factors:

Factor Number	Description	Weight	Assigned Value (0–5)	Weighted Value	Notes
T1	Distributed system	1.0	5	5.0	
T2	Response time or throughput performance objectives	1.0	5	5.0	
T3	End-user online efficiency	1.0	5	5.0	
T4	Complex internal processing	1.0	3	3.0	
T5	Reusability of code	1.0	3	3.0	
T6	Easy to install	0.5	3	1.5	
T7	Ease of use	0.5	5	2.5	
T8	Portability	2.0	4	8.0	
T9	Ease of change	1.0	3	3.0	
T10	Concurrency	1.0	3	3.0	
T11	Special security objectives included	1.0	5	5.0	
T12	Direct access for third parties	1.0	5	5.0	
T13	Special User training required	1.0	3	3.0	
			Technical Factor Value (TFactor)	**52.0**	

Technical complexity factor (TCF) = 0.6 + (0.01 * TFactor) 1.12 = 0.6 + (0.01 * 52)

Environmental Factors:

Factor Number	Description	Weight	Assigned Value (0–5)	Weighted Value	Notes
E1	Familiarity with system development process being used	1.5	2	3.0	
E2	Application experience	0.5	2	1.0	
E3	Object-oriented experience	1.0	2	2.0	
E4	Lead analyst capability	0.5	2	1.5	
E5	Motivation	1.0	3	3.0	
E6	Requirements stability	2.0	2	4.0	
E7	Part time staff	–1.0	0	0.0	
E8	Difficulty of programming language	–1.0	2	–3.0	
			Environmental Factor Value (EFactor)	**11.0**	

Environmental factor (EF) = 1.4 + (–0.03 * EFactor) 1.07 = 1.4 + (–.03 * 11)
Adjusted use case points (UCP) = UUCP *TCF *ECF 52.73 = 44 * 1.12 * 1.07
Person hours multiplier (PHM) PHM = 20
Person hours = UPC * PHM 1,054.6 = 52.73 * 20

FIGURE 2-C Project Effort Estimation Version 1 of the Integrated Health Clinic Delivery System

Staffing the Project

Ruby created a list of roles that needed to be filled. These included an infrastructure analyst to ensure both that the new system adheres to Patterson's infrastructure standards and that this infrastructure can support the new system. Integration with existing systems will be an important part of this project. Ruby also wanted both a systems analyst and a business analyst on the team to advocate for the technical and business perspectives of the analysis and design of the project. Because data tracking and analysis is a central requirement of the system, Ruby decided that a data analytics specialist was a necessary member of the team. Lastly, she needed a programmer with expertise in mobile application development and experience with video capture. Ruby chose the members of the team from the previous prescription fulfillment project team due to the expertise that they had developed. Most importantly, the group chosen formed an already jelled team with a high level of trust, project ownership, and synergy among members. The team roles and individuals assigned are listed staffing plan shown in Figure 2-D:

Role	Description	Assigned To
Project Manager	Oversees the project to ensure that it meets its objectives in time and within budget.	Ruby Neiley
Infrastructure Analyst	Ensures the system conforms to infrastructure standards at Patterson and that the Patterson infrastructure can support the new system	Sam Wilson
Systems Analyst	Designs the information system using a technology focus	Kelly Herman
Business Analyst	Designs the information system using a business focus	Sarah Kirschner
Data Analytics Specialist	Develops plan and structure for data tracking and analytics	Ben Joseph
Programmer	Codes system	Alice Smith
Reporting Structure: All project team members will report to Ruby		

FIGURE 2-D Staffing Plan

Creating and Managing the Workplan for Version 1 of the Integrated Health Clinic Delivery System

After completing staffing plan and project effort estimation, Ruby created an evolutionary work breakdown structure for Version 1. She started by reviewing the Enhanced Unified Process phases and workflows (Figure 1-16) and the evolutionary work breakdown structure template (Figure 2-17). At this juncture, Ruby does not have enough information to create a complete workplan and so has included as much detail as she knows to be correct (Figure 2-E). For example, Ruby is confident about the estimation of time to create the requirements definition and to elicit the requirements. However, she will not know whether how long it will take to develop the functional, structural, or behavioral analysis models until the actual requirements are defined. Until this determination can be made, any estimation as to the time required would be simply a guess. As time passes, Ruby expects to know much more about the development process and will add much more detail to the workplan. (Remember that the development process and the project management processes are iterative and incremental in nature).

	Duration	Dependency
I. Business Modeling		
a. Inception		
1. Understand current business situation		
2. Uncover business process problems		
3. Identify potential projects		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
II. Requirements		
a. Inception		
1. Identify appropriate requirements analysis technique		
2. Identify appropriate requirements gathering techniques		
3. Identify functional and nonfunctional requirements		II.a.1, II.a.2
4. Analyze current systems		II.a.1, II.a.2
5. Create requirements definition		II.a.3, II.a.4
A. Determine requirements to track		
B. Compile requirements as they are elicited		II.a.5.A
C. Review requirements with sponsor		II.a.5.B
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
III. Analysis		
a. Inception		
1. Identify business processes		
2. Identify use cases		III.a.I
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
IV. Design		
a. Inception		
1. Identify potential classes		III.a
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
V. Implementation		
a. Inception		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
VI. Test		
a. Inception		
b. Elaboration		

(Continued)

FIGURE 2-E Workplan for Version 1 of the Integrated Health Clinic Delivery System

	Duration	Dependency
c. Construction		
d. Transition		
e. Production		
VII. Deployment		
a. Inception		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
VIII. Configuration and change management		
a. Inception		
1. Identify necessary access controls for developed artifacts		
2. Identify version control mechanisms for developed artifacts		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
IX. Project management		
a. Inception		
1. Create workplan for the inception phase		
2. Create system request		
3. Perform feasibility analysis		IX.a.2
A. Perform technical feasibility analysis **B.** Perform economic feasibility analysis **C.** Perform organizational feasibility analysis		
4. Identify project size		IX.a.3
5. Identify staffing requirements		IX.a.4
6. Compute cost estimate		IX.a.5
7. Create workplan for first iteration of the elaboration phase		IX.a.1
8. Assess inception phase		I.a, II.a, III.a IV.a, V.a, VI.a VII.a, VIII.a, IX.a, X.a, XI.a XII.a
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
X. Environment		
a. Inception		
1. Acquire and install CASE tool		
2. Acquire and install programming environment		
3. Acquire and install configuration and change management tools		
4. Acquire and install project management tools		
b. Elaboration		

(Continued)

FIGURE 2
(*Contined*)

	Duration	Dependency
c. Construction		
d. Transition		
e. Production		
XI. Operations and Support		
a. Inception		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
XII. Infrastructure Management		
a. Inception		
1. Identify appropriate standards and enterprise models		
2. Identify reuse opportunities, such as patterns, frameworks, and libraries		
3. Identify similar past projects		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		

CHAPTER 3: PATTERSON SUPERSTORE

The Integrated Health Clinic Delivery System will enable mobile appointment scheduling, real-time communication with medical personnel (audio, video, and text), and facilitate clients' desire to electronically access health clinic services. The system will be developed using the phase development methodology and will begin with the mobile appointment portion of the project.

Determining the system's requirements is the most important activity in the systems development process. A requirement is WHAT the system must do or WHAT characteristics it must have. If the requirements are not fully or correctly defined, the system developed is unlikely to meet the needs of the user. In other words, if the requirements are wrong, the system will be wrong. Max defined the requirements in the systems request, at a very high level of detail:

- Mobile appointment scheduling
- Real-time communication with medical personnel (audio, video, and text)
- Tele-health assessment and diagnosis of minor problems through video house calls
- Data analytic and tracking capabilities

As the team moves into requirements determination, the high-level requirements will be expanded and refined. Requirements are either functional (WHAT the system must do) or nonfunctional (HOW the system will behave). Functional requirements answer the question of WHAT processing the system must perform or WHAT information the system must contain. Nonfunctional requirements refer to the behavioral properties of a system and will be explored in depth during the design phase (when the focus is on HOW the system will operate) but must be considered from a high-level viewpoint during analysis. Creating a requirements definition is an ongoing process of collecting information from users, analyzing the information collected, identifying the appropriate business requirements, and adding them to the requirements definition report. While requirements definition is an iterative process, it must be carefully managed to ensure that the evolving requirements fit the defined scope of the project. Scope creep has caused many projects to fail because the requirements grow to the point that the project is never finished. Ruby and Max are well aware that the scope of this project must be controlled. Their plan is to retain requirements beyond the scope of the project in a requirements list that can be addressed in the future versions.

Requirements Analysis Techniques

The envisioned system will improve the existing health clinic model by utilizing technology to improve the efficiency and effectiveness of clinic operations. Moderate change will be made to the way the clinic operates but the processes in place in the physical setting will see little disruption. For this reason, the team needs to understand the current system but will mainly focus on how to improve business processes. Some techniques that the team chose to use are technology analysis, informal benchmarking, and duration analysis. Max suggested that they plan joint application development (JAD) sessions with clinic managers, front-line employees of the clinics (who take calls, receive complaints, and handle delays), and with IT members who were involved with the prescription fulfillment rollout. Together this group could explore current processes and problems, and brainstorm technical solutions. To encourage all participants to freely share ideas, Ruby decided to run the session as an e-JAD session utilizing the existing laptops and installed software in the Training/JAD room. Sarah suggested that the team also schedule JAD sessions with technically savvy clients currently using the clinic to gain an understanding of the userexperience and how it might be improved.

Ruby and Max conducted the internal e-JAD sessions over a three-day period. Ruby used technology analysis to uncover available mobile and video technologies for the group to consider. The first day's session yielded a brainstormed list of how the health clinics might use these technologies. Based on the potential for adding business and fit with the objectives of the proposed system, Ruby categorized the ideas into three groupings: definite, possible, and unlikely. On the second day, Max projected websites and promotional materials that tele-healthcare providers and competing businesses currently use. While the sites were not very specific with what they showed, the JAD participants were able to use the information to begin a list of suggested business requirements for the project team. The third day's session did not go as well as the other two sessions. Ruby used duration analysis and attempted to introduce the activity elimination technique. Because employees became defensive and territorial regarding the speed and importance of their work, she quickly took a different tack and instead used the remaining time to continue brainstorming on the use of technology and to further develop business and high-level technical requirements.

Ruby and Sarah, the business analyst, conducted a one-day JAD session with existing clients from the busiest clinic. To encourage participation, they provided breakfast, lunch, and a superstore gift card to participants. Ruby started the session by stating that Patterson Superstore had listened to customers and was developing an Integrated Health Clinic Delivery System. When Ruby outlined the proposed features, the group became every excited. She explained that systems development was a lengthy process and that this project would be completed in phases with the mobile appointment scheduling coming first. Sarah then explained the importance of the user in developing the requirements for the system. Brainstorming occupied most of the morning session with ideas again sorted into definite, possible, and unlikely. One problem voiced about the current clinic that the system would address is that patients want to be able to schedule treatment but are often required to be evaluated prior to a treatment appointment being scheduled. During lunch, Ruby noticed that the participants had begun complaining about wait times and other problems with the current operation of the clinic. As soon as lunch was over, Ruby introduced the concepts of duration analysis and fairly successfully turned the complaints into an analysis of how long current processes took (from the customer's perspective). She decided not to use activity elimination because the clients lacked knowledge of the clinics' internal processes. Instead she used the results of the duration analysis to solicit suggestions for reducing activities that the clients experienced themselves (again being cognizant of internal processes). This would be useful information to share with development team as well as with the clinic managers.

Requirements Gathering Techniques

In addition to providing information and ideas, the JAD sessions established trust and rapport with the stakeholders. Realizing that they needed a deeper understanding of the existing processes, the team used document analysis, interview, and observation techniques to gather further information. First, Kelly, the systems analyst, collected existing reports (e.g., appointment schedules, input forms, diagnosis reference materials) and system documentation (functional, structural, and behavioral models) that shed light on the as-is system. In this way, Kelly was able to better understand the clinic processes and systems. When questions arose, Kelly conducted short interviews with the individual who provided clarification. Next, Kelly interviewed the senior analysts for the prescription fulfillment system to better understand the lessons learned from that project. Kelly asked if there were integration issues that she would need to address and also asked for input for the new system. Ruby interviewed the vendor of the Cloud platform that Patterson was using and spoke at length with the Patterson IT individual currently supporting the fulfillment system. Both provided information about the existing

communications infrastructure and its capabilities. Finally, Sarah spent a half-day visiting two of the health clinics and observing exactly how scheduling, clinic visits, documentation, and follow-up processes worked in the facilities.

Requirements Definition

Through the information collected, Kelly and Sarah tried to identify the business requirements for the system. As the project progressed, requirements were added to the requirements definition and grouped by requirement type. When questions arose, they worked with Max and Ruby to confirm that requirements were in scope. Requirements that fell outside of the scope of the current system were captured in a separate document to be saved for future use. After gathering and documenting the requirements, a draft requirements definition was distributed to Max and several health clinic managers. This group, along with the project team then met for a two-day JAD session to clarify, finalize, and prioritize business requirements. The project team created functional, structural, and behavioral models (Chapters 4, 5, and 6) that depicted the objects in the future system. Members of the IT department and pharmacy division reviewed the documents during interviews with the project team. Figures 3-A and 3-B show the functional and functional requirements.

System Proposal

Ruby reviewed the requirements definition and the other deliverables that the project team created. Given the six-month time frame for delivery of the first phase of the project, Ruby decided to time box the project into three versions. The first version would implement the mobile scheduling portion for delivery in six months. The second version, planned for late spring or early summer, would implement the real-time communication with medical personnel (audio, video, and text). The third version would initiate tele-health assessment and diagnosis of minor problems through video house calls. Data analytic and tracking capabilities would be built into all three versions. Figure 3-C shows a portion of the Systems Proposal.

Nonfunctional Requirements

1. Operational requirements
   - 1.1 The system will operate on any web browser including mobile
   - 1.2 The system will integrate with the current clinic systems
   - 1.3 The system will automatically back up each day at midnight
2. Performance requirements
   - 2.1 The system must be available 24 hours daily (365 days per year)
   - 2.2 Response time for interactions between the system and the user will be less than three seconds
   - 2.3 The system will store and retrieve appointment and other transactional information every two seconds
3. Security requirements
   - 3.1 Access to patient medical information is limited to medical staff only
   - 3.2 Scheduling and administrative personnel can access patient contact and billing information but not medical information
4. Cultural and political requirements
   - 4.1 The system will comply with all regulatory requirements. The health clinics operate in a highly regulated field. Compliance with all regulation is imperative
   - 4.2 Strict compliance with all aspects of HIPAA will be maintained at all times

FIGURE 3-A Nonfunctional Requirements

Functional Requirements

1. Schedule appointment
   - 1.1 Client requests to be seen by the clinic
   - 1.2 The system displays the defined service offerings list
   - 1.3 Client either chooses a defined service offering from the list or requests that a service need survey be completed so that the system can determine whether the service needed falls within the scope of the clinic's capabilities
   - 1.4 Referral information will be listed for conditions beyond the scope of the clinic's service
     - 1.4.1 Compare and evaluate referral need against referral list
     - 1.4.2 Display appropriate referrals
   - 1.5 Appointment information will be listed for conditions that fall within the scope of the clinic's services
     - 1.5.1 Current real-time availability will be displayed with wait time listed
     - 1.5.2 Clients can choose appointment time for the current day or make an appointment in advance
     - 1.5.3 The calendar will be updated to reflect scheduled appointment
     - 1.5.4 Confirmation will be sent to client
2. Communicate real time
   - 2.1 Client can request real-time meeting with caregiver
   - 2.2 Client indicates available time and technology preference
   - 2.3 Caregiver responds with duration and availability
   - 2.4 Session scheduled with clients, caregivers
3. Assess via tele-health
   - 3.1 Client answers question matrix to determine suitability for tele-health assessment
   - 3.2 Limited diagnosis developed based on matrix answers from client
   - 3.3 Diagnosis info reviewed by caregiver
   - 3.4 Diagnosis info given to client if the problem is minor and diagnosis info is conclusive
   - 3.5 Or Video-conference scheduled
   - 3.6 Video conference held with diagnosis, follow-up, or referral

FIGURE 3-B Functional Requirements

Outline of the Systems Proposal for the Integrated Health Clinic Delivery System

1. Table of Contents
2. Executive Summary (To be completed once everything else is done)
3. System Request (Figure 2-A)
4. Economic Feasibility (Figure 2-B)
5. Evolutionary Work Breakdown Structure (Figure 2-E)
6. Requirements Definition (Figures 3-A and 3-B)
7. Functional Model: To be completed in the future (see Chapter 4).
8. Structural Models: To be completed in the future (see Chapter 5).
9. Behavioral Model: To be completed in the future (see Chapter 6).
10. Appendices
    - A. Staffing Plan (2-D)

FIGURE 3-C Systems Proposal Outline

CHAPTER 4: PATTERSON SUPERSTORE CASE

While the Integrated Health Clinic Delivery System will be analyzed, designed, developed, and implemented in phases, Ruby wanted to also maintain a big picture perspective of the project. Therefore, as a first step toward developing a model of the functional requirements for the system, Ruby directed Sarah, the business analyst, to capture thehigh-level business processes in a use case diagram. Doing so provided a straightforward way to view the main functions of the entire system and also depict the interactions between the business processes and the systems environment. Figure 4-A shows a use case diagram of the three high-level use cases (that correspond to the three versions to be developed) for the envisioned Integrated Health Clinic Delivery System.

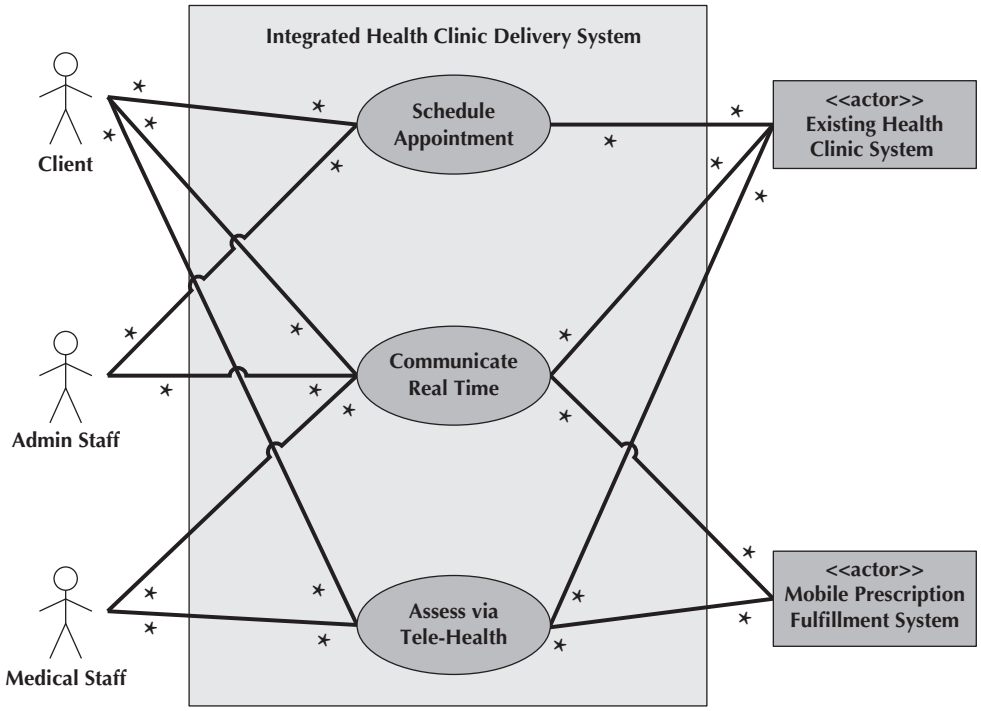


FIGURE 4-A Use Case Diagram for the Integrated Health Clinic Delivery System

To further develop a high-level perspective of the Integrated Health Clinic Delivery System, three overview use cases were developed. These overview use cases will help the analysts and user agree on the requirements from a high-level perspective. For this reason, the three overview essential use cases shown below document the information known from the use case diagram. As more information is learned about the use cases, the use case descriptions will be converted from overview to detailed use case descriptions.

The use case diagram (Figure 4-A) showed the business processes and system environment; the overview essential use cases (Figures 4-B, 4-C, 4-D) modeled the high-level processes for the overall system. Given the six-month time frame for delivery of the first phase,

Use Case Name: Schedule Appointment		**ID:** 1	**Importance Level:** High
Primary Actor: Health Clinic Client	**Use Case Type:** Overview, Essential		
Stakeholders and Interests: Client wants to electronically schedule an appointment at the health clinic Existing Health Clinic System provides information about clinic services and appointment availability Administrative Staff confirms appointments			
Brief Description: This use case describes how an appointment is scheduled electronically			
Trigger: Client requests to be seen **Type:** External			
Relationships: **Association:** Client, Admin Staff, Existing Health Clinic System, Mobile Prescription Fulfillment System **Include:** **Extend:** **Generalization:**			
Normal Flow of Events:			
SubFlows:			
Alternate/Exceptional Flow:			

FIGURE 4-B Overview Use Case Description for the Schedule Appointment Use Case

Use Case Name: Communicate Real Time		**ID:** 2	**Importance Level:** High
Primary Actor: Health Clinic Client	**Use Case Type:** Overview, Essential		
Stakeholders and Interests: Client wants to electronically communicate with health clinic provider in real time Existing Health Clinic System Service displays availability information Medical Staff communicates with client based on schedule availability Administrative Staff confirms real-time session time			
Brief Description: This use case describes how clients and medical staff communicate in real time			
Trigger: Health Clinic Client requests real-time communication **Type:** External			
Relationships: **Association:** Client, Admin Staff, Medical Staff, Existing Health Clinic System, Mobile Prescription Fulfillment System **Include:** **Extend:** **Generalization:**			
Normal Flow of Events:			
SubFlows:			
Alternate/Exceptional Flow:			

FIGURE 4-C Overview Use Case Description for the Communicate Real-Time Use Case

Use Case Name: Assess via Tele-Health	**ID:** 3	**Importance Level:** High
Primary Actor: Health Clinic Client	**Use Case Type:** Overview, Essential	
Stakeholders and Interests: Client requests tele-health assessment Existing Health Clinic System Service provides information about tele-health clinic services and appointment availability Medical Staff provides tele-health assessment and diagnosis based on schedule availability Administrative Staff confirms appointments		
Brief Description: This use case describes how an appointment is scheduled electronically		
Trigger: Health Clinic Client requests tele-health assessment **Type:** External		
Relationships: **Association:** Client, Medical Staff, Existing Health Clinic System, Mobile Prescription Fulfillment System **Include:** **Extend:** **Generalization:**		
Normal Flow of Events:		
SubFlows:		
Alternate/Exceptional Flow:		

FIGURE 4-D Overview Use Case Description for the Assess via Tele-Health Use Case

Ruby directed the team to get started by closely studying the business processes for Schedule Appointment and to again review the functional and nonfunctional requirements (Figures 3-A and 3-B). Once the team was comfortable with their understanding of the requirements for Schedule Appointment, they began the modeling process for Version 1 by drawing an activity diagram for the use case.

Business Process Modeling with Activity Diagrams

While developing the activity diagram for the Schedule Appointment process, the team identified two additional activities needed in the process: comparing the referral need with the list of available referrals and subsequently displaying the appropriate referral information. Figure 4-E shows the updated functional requirements and the activity diagram for the Schedule Appointment use case.

1. Schedule appointment
   - 1.1 Client requests to be seen by the clinic
   - 1.2 The system displays the defined service offerings list
   - 1.3 Client either chooses a defined service offering from the list or requests that a triage questionnaire be completed so that the systems can determine whether the service needed falls within the scope of the clinic's capabilities
   - 1.4 Referral information will be listed for conditions beyond the scope of the clinic's service
     - 1.4.1 Compare and evaluate referral need against referral list
     - 1.4.2 Display appropriate referrals

(Continued)

1.5 Appointment information will be listed for conditions that fall within the scope of the clinic's services

1.5.1 Current real-time availability will be displayed with wait time listed

1.5.2 Clients can choose appointment time for the current day or make an appointment in advance

1.5.3 The calendar will be updated to reflect scheduled appointment

1.5.4 Confirmation will be sent to client

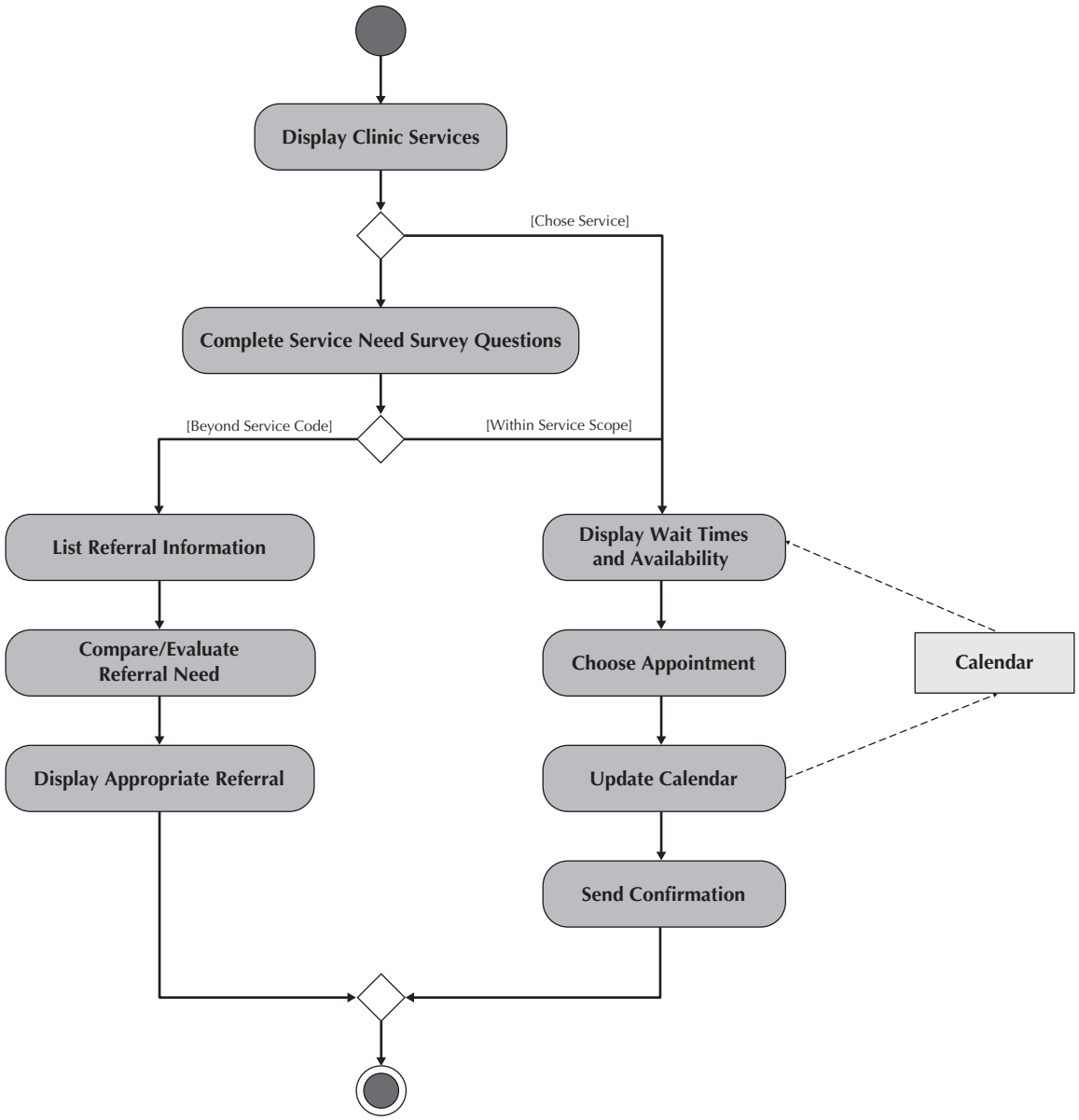


FIGURE 4-E Functional Requirements and Activity Diagram for the Schedule Appointment Use Case

Business Process Documentation with Use Cases and Use Case Descriptions

The activity diagram (Figure 4-E) described the underlying activities that support the Schedule Appointment process. When Kelly and Sarah conducted a walkthrough of the activity diagram for the Schedule Appointment Use Case, the complexity of the activities led them to conclude that they needed to decompose the functions of the schedule appointment use case into a set of simpler use cases.

By reviewing the shape of the activity diagram, the team realized that the Display Clinic Services, Complete Service Need Survey Questions, and the determination of whether the service need was beyond the scope of the services provided by the clinic or not should be associated with the original Schedule Appointment use case. The team also saw that the two separate branches in the original activity diagram naturally could be split into two separate use cases—one for making a referral and one for making an appointment.

Lastly, given that these two new use cases were mutually exclusive from one another, Kelly and Sarah realized that both use cases should be modeled with an extends relationship with the Schedule Appointment use case. These changes dictated a revised use case diagram (Figure 4-F) and new activity diagrams and use case descriptions for each of the use cases (Figures 4-G, 4-H, 4-I, 4-J, 4-K, and 4-L).

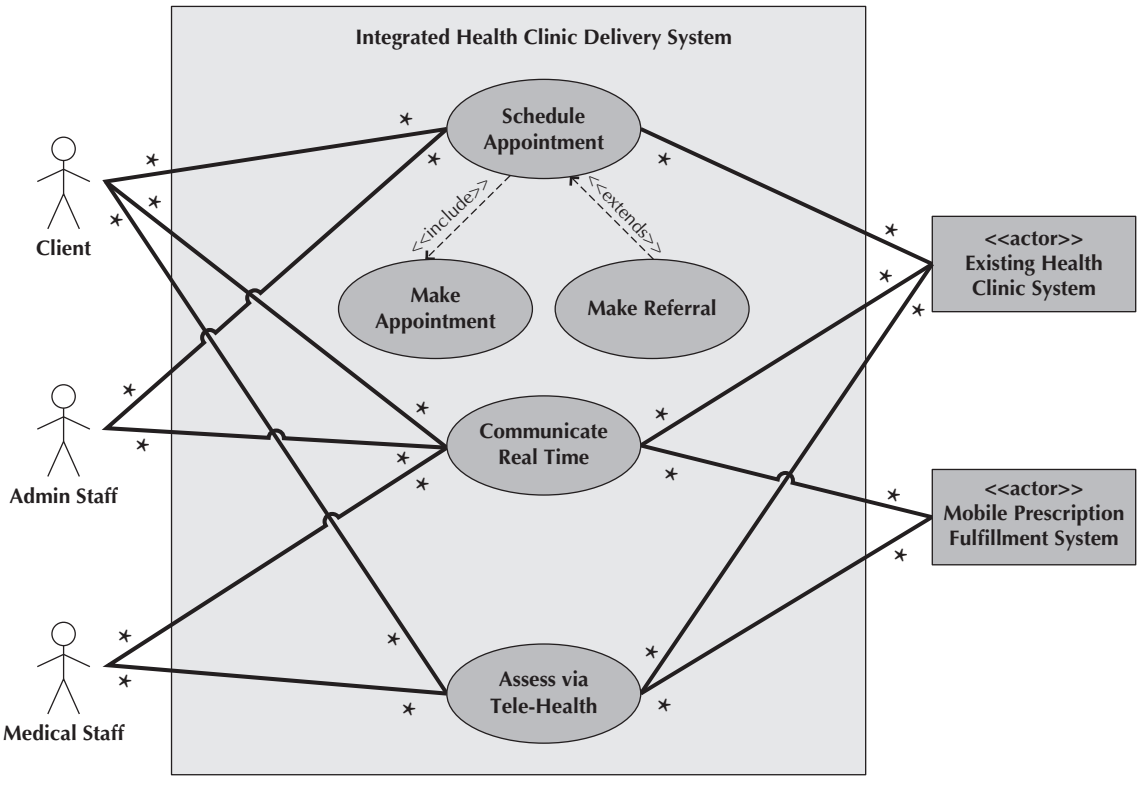


FIGURE 4-F Revised Use Case Diagram for the Integrated Health Clinic Delivery System

Use Case Name: Schedule Appointment	**ID:** 1	**Importance Level:** High
Primary Actor: Client	**Use Case Type:** Detail, Essential	
Stakeholders and Interests: Client wants to schedule an appointment Existing Health Clinic System provides information about clinic services		
Brief Description: This use case describes how an appointment is scheduled electronically		
Trigger: Client requests to be seen **Type:** External		
Relationships: **Association:** Client, Existing Health Clinic System **Include:** **Extend:** Make Appointment, Make Referral **Generalization:**		
Normal Flow of Events: 1. Client requests to be seen by the clinic 2. The system displays the defined service offerings list 3. Client Chooses an existing service and Executes the Make Appointment use case		
SubFlows: S-1: Determine Suitability 1. Complete service need survey questions 2. Determine whether service need is within scope of clinic's services 3. Execute Make Appointment use case		
Alternate/Exceptional Flow: 3a. Execute S-1: Determine Suitability S-1, 3a. Execute Make Referral use case		

FIGURE 4-G Detail Use Case Description for the Schedule Appointment Use Case

Object-oriented systemsare developed in an incremental and iterative manner. This is especially true when the phased approach is used. The three versions of the Integrated Health Clinic Delivery System will each go through individual process and functional modeling as well as structural and behavior modeling with iteration across all of these tasks. Fully capturing and representing the requirements for this business information system will require iteration between the three architectural models for each version.

Validation and Verification

After creating detail use case descriptions and activity diagrams for the use cases, Sarah and Kelly checked them for consistency against the requirement determination documents. When they did this, they uncovered a series of mistakes. First, they realized that the Make Appointment use case was called from the Schedule Appointment use case only if the services needed by the client were available at the clinic. This was modeled correctly in both the use case description (Figure 4G) and the activity diagram (Figure 4-H). However, the use case diagram (Figure 4-F) modeled it as an include relationship. Given that the relationship should be optional, this should have been modeled as an extends relationship instead of an include relationship. Second, as they compared the use case diagram (Figure 4-F) with the use case

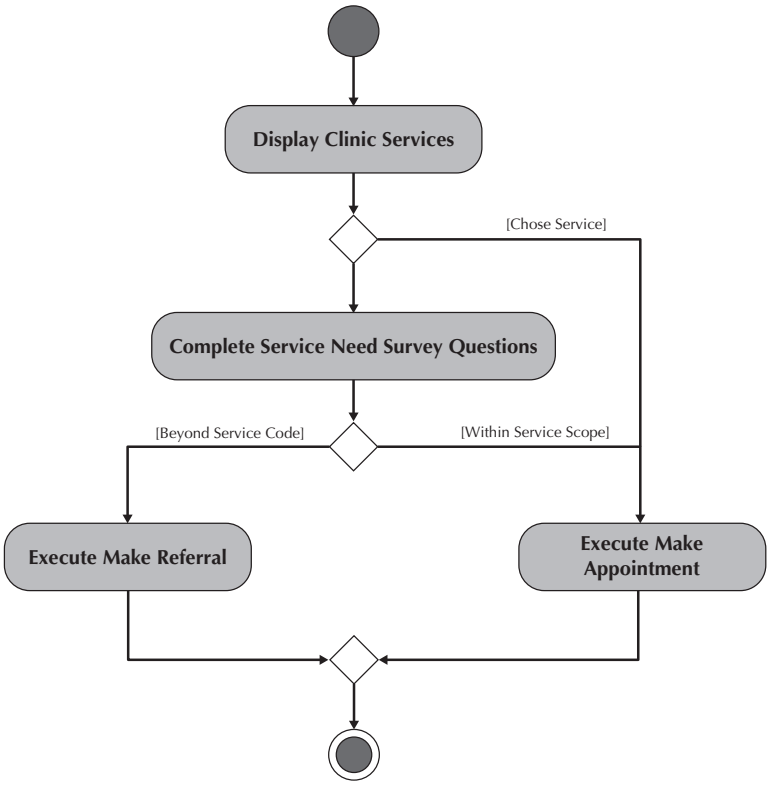


FIGURE 4-H Activity Diagram for the Schedule Appointment Use Case

Use Case Name: Make Appointment	**ID:** 1-1	**Importance Level:** High
Primary Actor: Client	**Use Case Type:** Detail, Essential	
Stakeholders and Interests: Client wants to schedule an appointment Administrative Staff provides wait time information, updates calendar, and sends confirmation Existing Health Clinic System Service supplies information about appointment availability		
Brief Description: This use case describes how the client chooses an appointment		
Trigger: Client wishing to schedule a clinic appointment has service needs that match clinic capability **Type:** External		
Relationships: **Association:** Client, Admin Staff, Existing Health Clinic System **Include:** **Extend:** **Generalization:**		
Normal Flow of Events: 1. Display current health clinic traffic/wait time appointment availability to client 2. Client enters appointment preference date/time 3. Client's appointment preference is checked against appointment availability		

(*Continued*)

4. Matching appointment availability displayed
5. Client chooses available and desired appointment
6. Calendar is updated
7. Appointment confirmation is sent to client

SubFlows:

Alternate/Exceptional Flow:

5a. If no match occurs, client iterates steps 2 through 5 until satisfactory time is found

FIGURE 4-I Detail Use Case Description for the Make Appointment Use Case

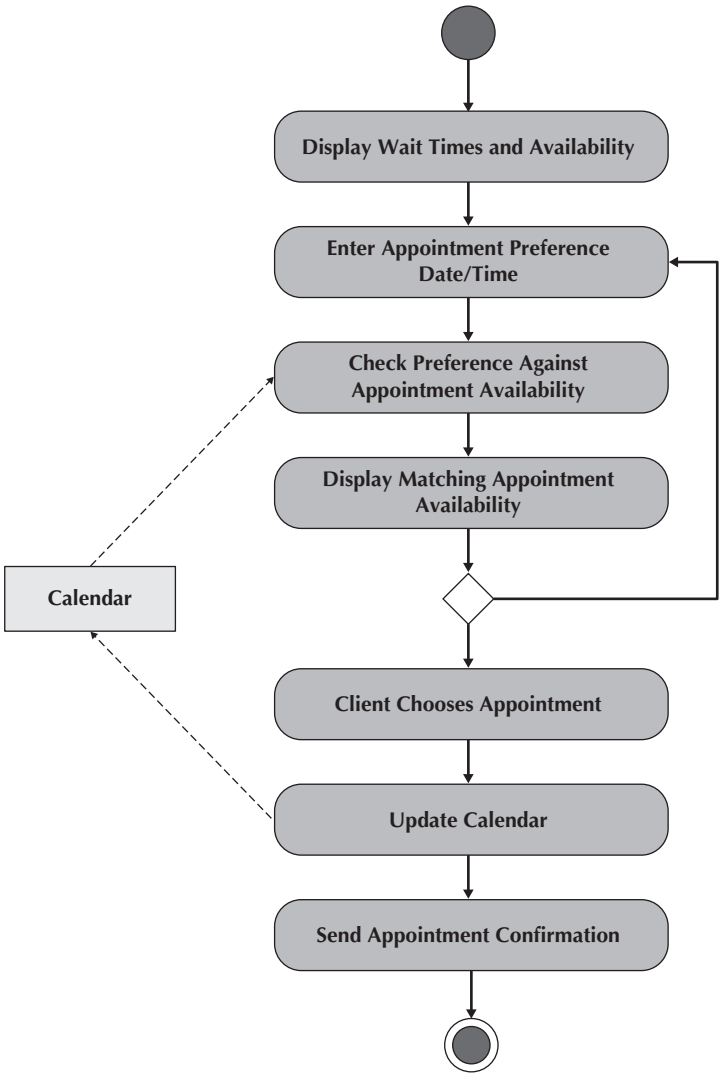


FIGURE 4-J Activity Diagram for the Make Appointment Use Case

Use Case Name: Make Referral	**ID:** 1-2	**Importance Level:** High
Primary Actor: Client	**Use Case Type:** Detail, Essential	
Stakeholders and Interests: Client needs a referral Existing Health Clinic System Service provides information about referrals		
Brief Description: This use case describes how referrals how handled		
Trigger: Client need cannot be met by clinic **Type:** External		
Relationships: **Association:** Client, Existing Health Clinic System **Include:** **Extend:** **Generalization:**		
Normal Flow of Events: 1. List Referral Information 2. Compare and evaluate referral need against referral list 3. Display appropriate referrals		
SubFlows:		
Alternate/Exceptional Flow:		

FIGURE 4-K Detail Use Case Description for the Make Referral Use Case

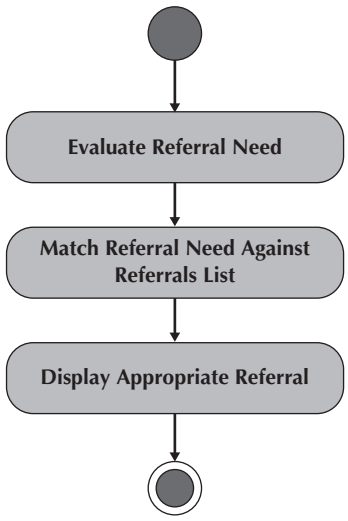


FIGURE 4-L Activity Diagram for the Make Referral Use Case

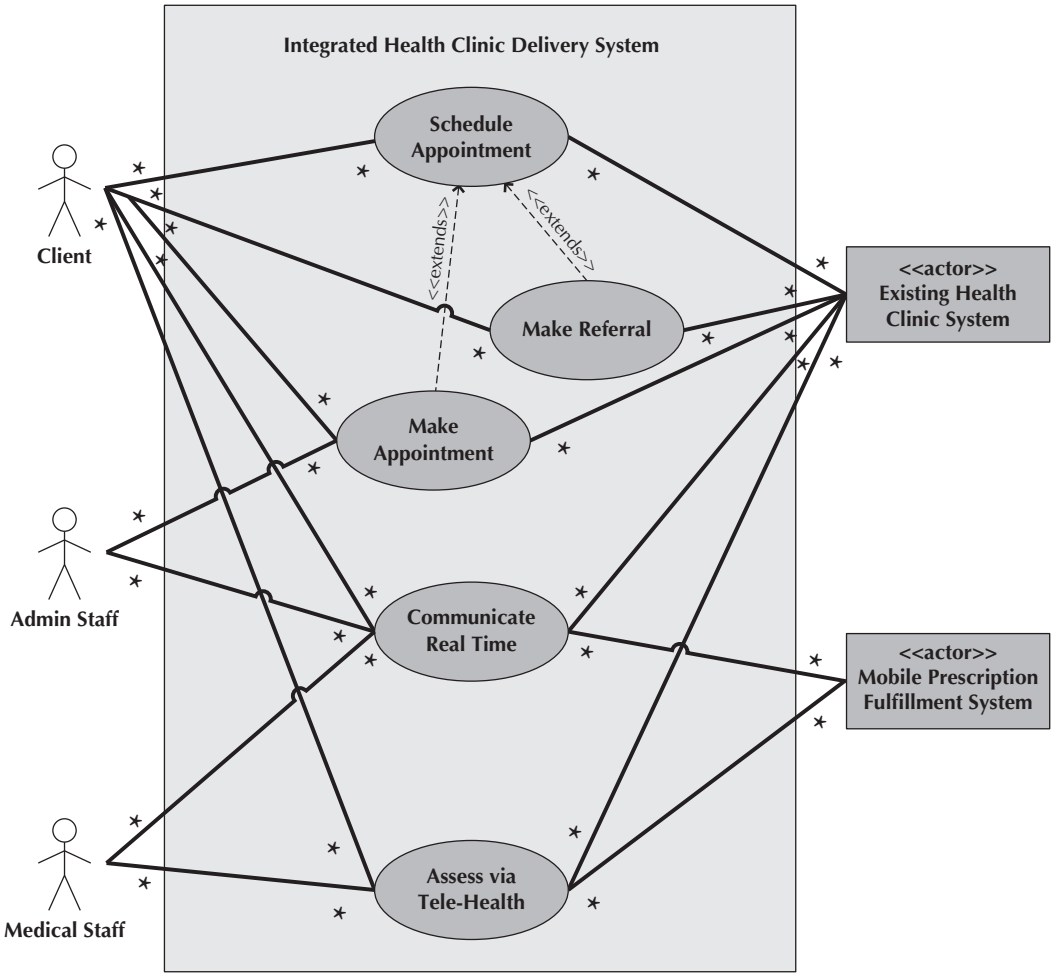


FIGURE 4-M Revised Use Case Diagram for the Integrated Health Clinic Delivery System

descriptions (Figures 4-G, 4-I, and 4-K), they noticed that the Stakeholders and Interests section of the use case descriptions no longer matched the associations in the use case diagram. Upon careful review, they decided that again, the use case diagram was in error and corrected it (Figure 4-M). Third, they noticed that there was no guard conditions associated with the decision in the activity diagram of the Make Appointment use case (Figure 4-J). Therefore, they looked at the use case description (Figure 4-I) to identify an appropriate guard condition. The corrected activity diagram is shown in Figure 4-N.

Sarah and Kelly felt foolish that they had made these errors but they were experienced enough to realize how easy it is to overlook the obvious; they were also thankful that the errors had been uncovered now rather than later! As the team models the static structure of the system, more information will be uncovered that may require further iteration of the functional model.

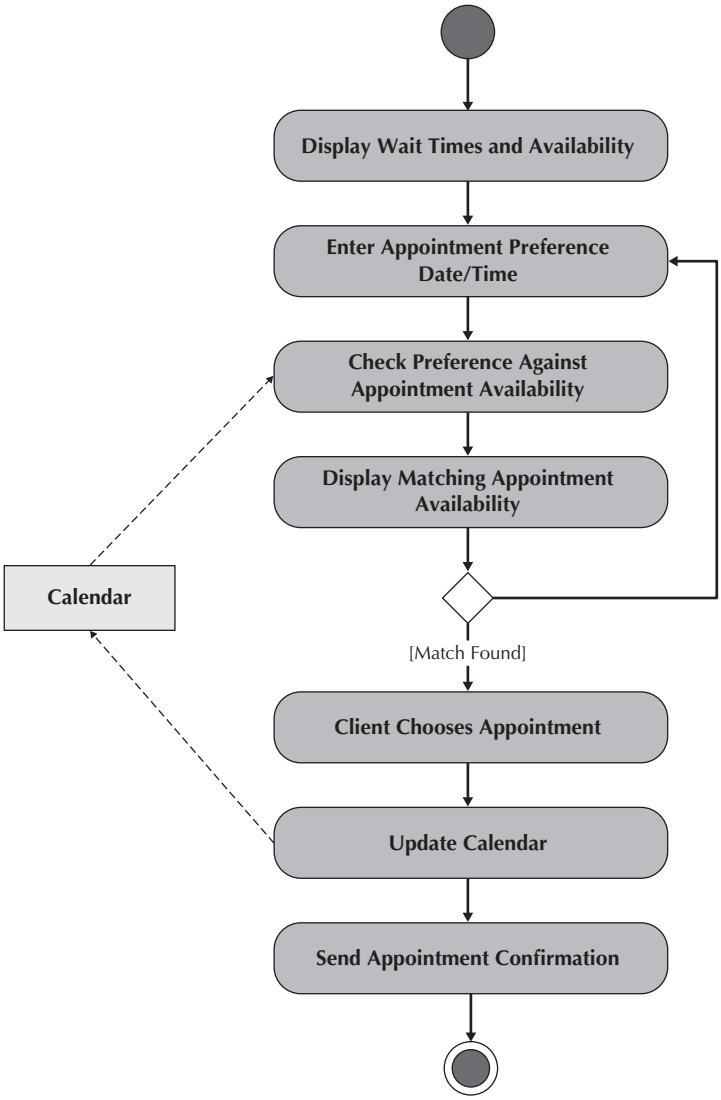


FIGURE 4-N Revised Activity Diagram for the Make Appointment Use Case

CHAPTER 5: PATTERSON SUPERSTORE CASE

Structural Modeling using Class Responsibility Collaboration (CRC) Cards

After creating the functional model for the Mobile Scheduling phase (Version 1) of the Integrated Health Clinic Delivery System, the team had a good understanding of the business processes. Now it was time to identify the key data and to develop the structural model of the objects that support those business processes.

Ruby and the team conducted a textual analysis (Figure 5-1) of the use case descriptions (Figures 4-G, 4-I, and 4-K) created during the functional modeling activities to create CRC cards for possible candidate classes. By examining the Stakeholders and Interests, Brief Description, Trigger, Normal Flow of Events, and the Alternative/Exceptional Flow section of each use case description, they identified a set of candidate classes: client, existing health clinic system, appointment, referral, clinic services, service need, survey, administrative staff, wait time, time/data, appointment preference, appointment confirmation, match, and referral need. Ruby understood that these candidate classes may not appear in the final structural model. However, the goal, in this first iteration, was to be as thorough as possible. Before actually creating a set of CRC cards for each of these, Ruby asked everyone to try and identify whether there were any attributes, operations, and/or relationships for these potential candidate classes. Also, were any of these potential classes only actors, in which case, no class would be necessary in the structural model. After this discussion, the team created a set of CRC cards (Figure 5-A) for the following candidate classes: client, appointment, service referral, clinic service, service need, survey, service listing, referral listing, and appointment list.

FIGURE 5-A CRC Cards—Derived from the Use Case Descriptions

Front:

Class Name: Client	**ID:** 1	**Type:** Concrete Domain
Description: An individual wishing to be seen at the Patterson Superstore Health Clinic		**Associated Use Cases:** 1, 1-1, 1-2, 2,3

Responsibilities	**Collaborators**
Request Appointment	Appointment
Supply Service Need information	Service Need

Back:

Attributes:

Name	
Address	
Phone	
E-mail	
Insurance Carrier	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Appointment, Service Need, Survey, Referral

Front:

Class Name: Survey	**ID:** 2	**Type:** Concrete Domain
Description: The set of questions asked and answers provided to ascertain service need		**Associated Use Cases:** 1

Responsibilities	Collaborators
Display Question	Survey Question
Record Answers	

Back:

Attributes:

Survey Number	
Question Set	
Answer Set	

Relationships:

Generalization (a-kind-of):	
Aggregation (has-parts):	Survey Question
Other Associations:	Client, Service Need

Front:

Class Name: Service Need	**ID:** 3	**Type:** Concrete Domain
Description: Health service need of Client		**Associated Use Cases:** 1, 1-1, 1-2

Responsibilities	Collaborators
Identify Need	Clinic Service, Survey

Back:

Attributes:

Service Need Description	

Relationships:

Generalization (a-kind-of):	
Aggregation (has-parts):	
Other Associations:	Client, Survey, Appointment, Referral,
	Clinic Service, Appointment List, Referral List

FIGURE 5-A (*Continued*)

Front:

Class Name: Referral	**ID:** 4	**Type:** Concrete Domain
Description: A referral to a medical provider for services not provided by the clinic		**Associated Use Cases:** 1-2

Responsibilities	**Collaborators**

Back:

Attributes:

Provider Name

Date

Time

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Service Need, Client, Referral List

Front:

Class Name: Clinic Service	**ID:** 5	**Type:** Concrete Domain
Description: A service provided by the clinic		**Associated Use Cases:** 1, 1-1

Responsibilities	**Collaborators**
Compare	

Back:

Attributes:

Service Number

Service Name

Service Description

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Service Need

FIGURE 5-A (*Continued*)

Front:

Class Name: Appointment	**ID:** 6	**Type:** Concrete Domain
Description: A client appointment with the clinic		**Associated Use Cases:** 1-1

Responsibilities	Collaborators

Back:

Attributes:

Date	
Time	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Client, Service Need, Calendar

Front:

Class Name: Calendar	**ID:** 7	**Type:** Concrete Domain
Description: The list of all clinic appointments		**Associated Use Cases:** 1-1

Responsibilities	Collaborators
Compare	
Get Avail	

Back:

Attributes:

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts): Appointment

Other Associations:

FIGURE 5-A (*Continued*)

Front:

Class Name: Survey Question	**ID:** 8	**Type:** Concrete Domain
Description: A Survey question used to help determine needed services		**Associated Use Cases:** 1

Responsibilities	**Collaborators**

Back:

Attributes:

Question Number	
Question	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Survey

Front:

Class Name: Referral List	**ID:** 9	**Type:** Concrete Domain
Description: A list of medical providers and their services that they provide not provided by clinic		**Associated Use Cases:** 1

Responsibilities	**Collaborators**
Match Service Need	Service Need
Display Matched List	

Back:

Attributes:

Provider Name	Services Provided
Location	
Phone	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Service Need, Referral

FIGURE 5-A (*Continued*)

Review CRC Cards

After creating the set of CRC cards, the team decided that they should role-play the cards using the use case descriptions. During this exercise, the team realized that Identify Need responsibility of the Service Need class was very complex. In fact, it dealt with three use cases: Schedule Appointment (Figures 4-G and 4-H), Make Appointment (Figures 4-I and 4N), and Make Referral (Figures 4-K and 4-L). Consequently, they decomposed this responsibility along the lines of the three use cases. Second, the team realized that they had accidentally combined both the actor and class aspects of the client. For example, the Client actor requests an appointment; not the Client class. Third, the role playing raised the question as to whether the direct relationship between the Client class and the Service Need class was necessary or not. After a discussion, the team decided that at this point in the evolution of the system, the relationship should be kept. Lastly, they saw that the attributes Name and Address in the client class needed to be expanded into Last Name, First Name, Street, City, State, and Zip Code. These revisions were made, and the CRC cards that were revised are shown in Figure 5-B below.

Front:

Class Name: Client	**ID:** 1	**Type:** Concrete Domain
Description: An individual wishing to be seen at the Patterson Superstore Health Clinic		**Associated Use Cases:** 1, 1-1,1-2, 2, 3
Responsibilities		**Collaborators**

Back:

Attributes:

First Name	Country
Last Name	Zip Code
Street	Phone
City	E-mail
State	Insurance Carrier

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Appointment, Survey, Referral, Service Need

FIGURE 5-B Updated CRC Cards

Front:

Class Name: Service Need	**ID:** 3	**Type:** Concrete Domain
Description: Health service need of Client		**Associated Use Cases:** 1, 1-1, 1-2

Responsibilities	**Collaborators**
Identify Need	Clinic Service, Survey, Client
Make Appointment	Appointment, Appointment List, Client
Make Referral	Referral, Client

Back:

Attributes:

Service Need Description

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Client, Survey, Appointment, Referral, Clinic Service, Appointment List, Referral List

FIGURE 5-B (*Continued*)

The preliminary class diagram developed from the CRC cards is shown in Figure 5-C.

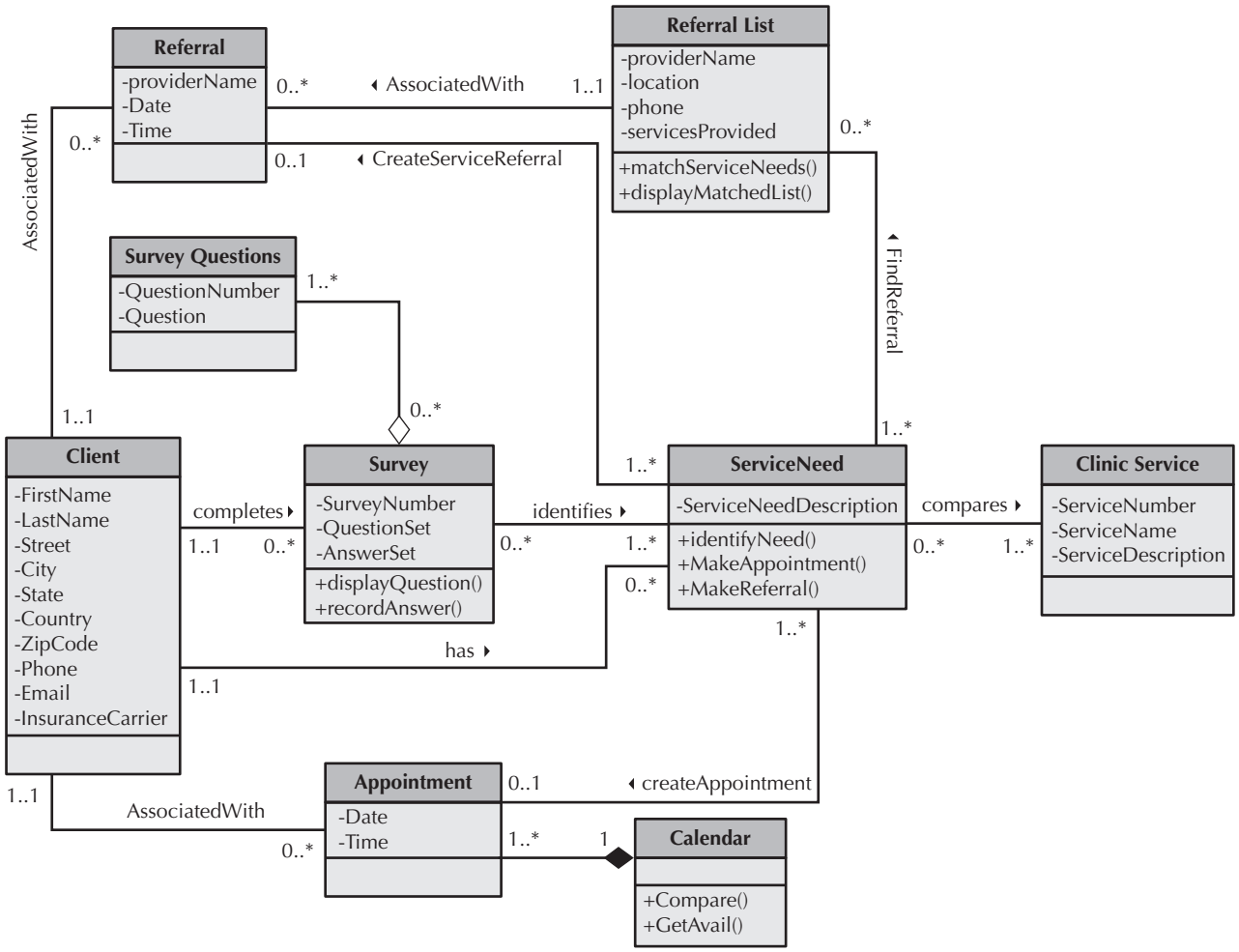


FIGURE 5-C Preliminary Class Diagram

Review Structural Model

The next step was to review the structural model for missing or unnecessary classes, attributes, operations, and relationships. At this point, since the team was still unsure about the details of the behavior aspects of the model, they decided to leave the model as is and review it further as they move into behavioral modeling. Finally, the team carefully reviewed the structural model to make sure that the CRC cards and the class diagram were in agreement. Even though they had previously reviewed the CRC cards, when comparing them to the class diagram, the team discovered that there was an association recorded on the Clinic Service CRC card that had been omitted from both the Calendar CRC card and the class diagram. So, they updated both (Figures 5-D and 5-E) to bring the structural model into agreement. Even though there was missing information in the model (e.g., operations for the Client class), the team decided that the structural model was sufficient to move on to behavioral modeling. So, pending changes that could arise with behavioral modeling, the team was satisfied with the structural model.

Front:

Class Name: Calendar	**ID:** 7	**Type:** Concrete Domain
Description: The list of all clinic appointments		**Associated Use Cases:** 1-1

Responsibilities	**Collaborators**
Compare	
Get Avail	

Back:

Attributes:

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts): Appointment

Other Associations: Service Need

FIGURE 5-D Updated Calendar CRC Card

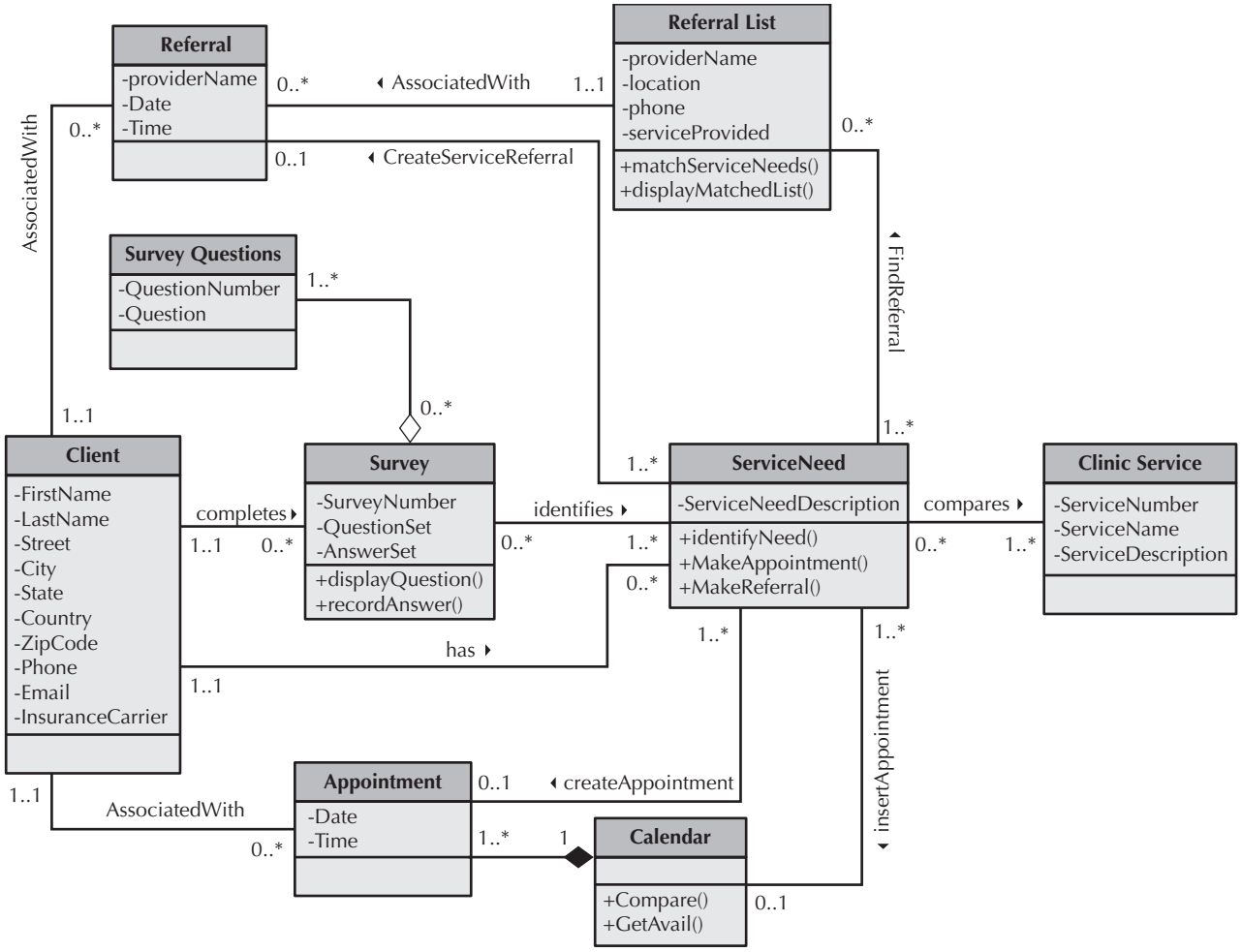


FIGURE 5-E Updated Class Diagram

CHAPTER 6: PATTERSON SUPERSTORE CASE

After developing the functional and structural models, Ruby tasked the team with developing the behavioral models for the Mobile Scheduling (Version 1) of the Integrated Health Clinic Delivery System. While the structural model depicted the static aspects of the system, the behavioral model shows the internal dynamic aspects of the system. By modeling both the static (structural) and dynamic (behavioral) aspects of a system, object-oriented systems analysis and design attempts to view the underlying problem domain in a holistic way.

The team set out to create interaction diagrams (sequence and communication diagrams), behavioral state machines, and a CRUDE matrix. In this chapter, we will only show the interaction models for the Schedule Appointment use case, a behavioral state machine for the Client class, and a CRUDE matrix based on the Schedule Appointment use case. However, please remember that the team created behavioral models for all of the use cases and classes in the evolving system.

Sequence Diagrams

Sequence diagrams are interaction diagrams that show, for a single use case, the messages that pass between objects over time. The focus of these diagrams is the *order* of messages in an interaction. The process for developing sequence diagrams is to (first) determine the context of the diagram and then to identify the actors and objects in the scenario being modeled. The next step is to set a lifeline for each object. The fourth step is to add the messages (in ordered sequence) to the diagram. Finally, the execution occurrence is added to each object and actor's lifeline.

As the team began work on the sequence diagram, they realized that there needed to be an intermediary between a client and the problem domain classes. In a non-mobile context, the

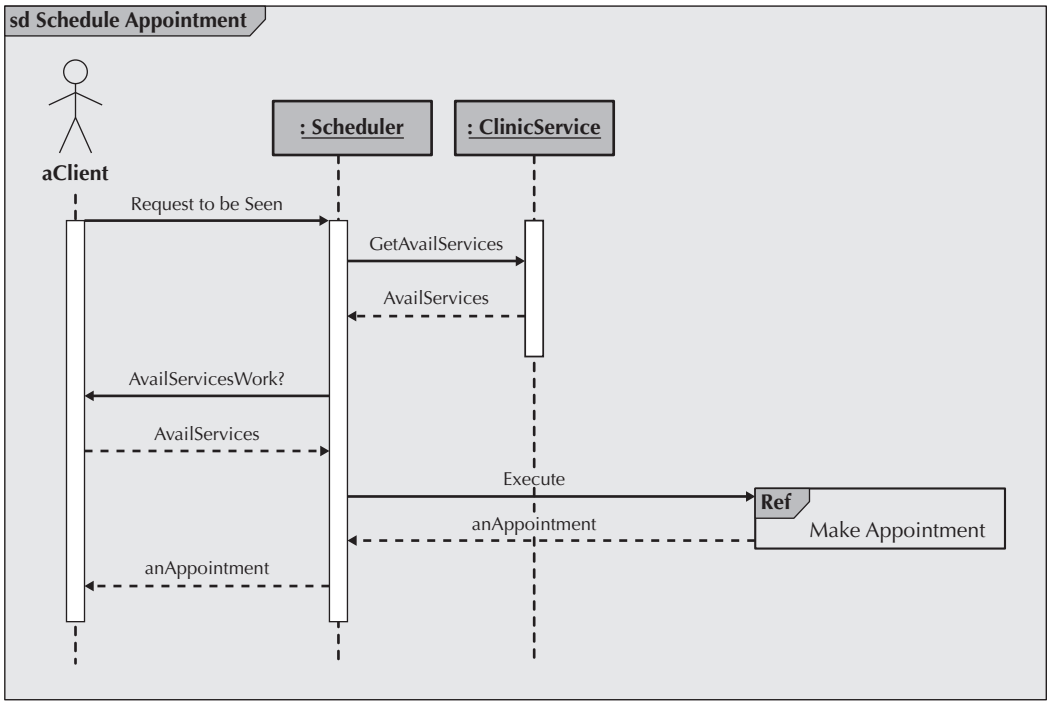


FIGURE 6-A Schedule Appointment Sequence Diagram for the client chooses service scenario

intermediary could be a receptionist. However, in a mobile context, the intermediary would have to be an actor-like class that would handle the interaction. Therefore, the team decided to add a Scheduler class to the evolving representation. This change requires modifications to the structural model (these changes will be presented in the next chapter). Also, given that the Schedule Appointment use case (Figures 4-G and 4-H) executes both the Make Appointment use case (Figures 4-I and 4-N) and the Make Referral use case (Figures 4-K and 4-L), the team had to incorporate a reference in each scenario of the Schedule Appointment use case to the appropriate use case being executed. Finally, since the current Health Clinic System already triages services needs when the client comes to the clinic in person, the Schedule Appointment use case will call the existing Triage use case to make the determination as to the services required based on the survey. Figures 6-A, 6-B, and 6-C show the sequence diagrams for the three

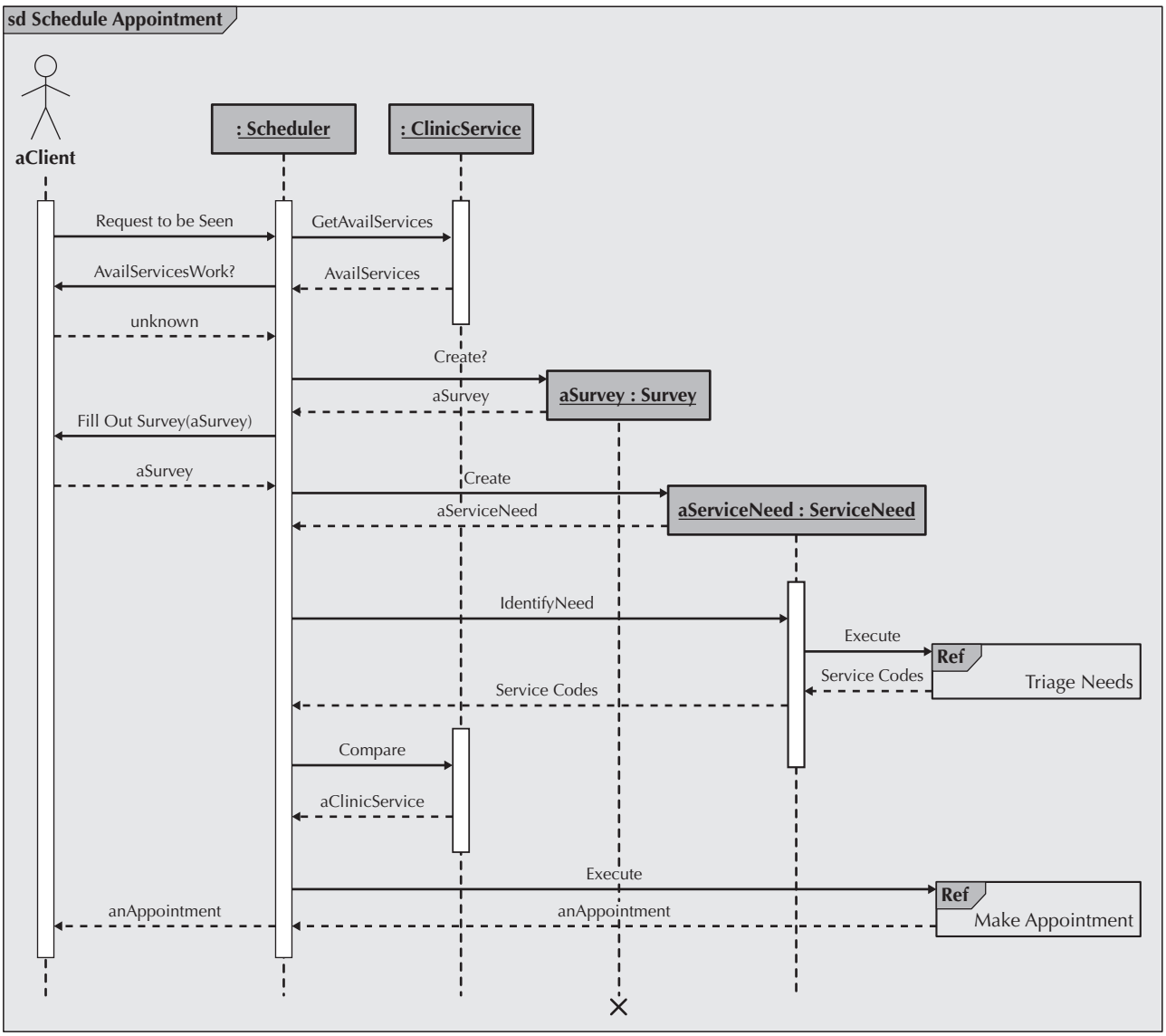


FIGURE 6-B Schedule Appointment Sequence Diagram for the filling out survey and clinic provides appropriate service scenario

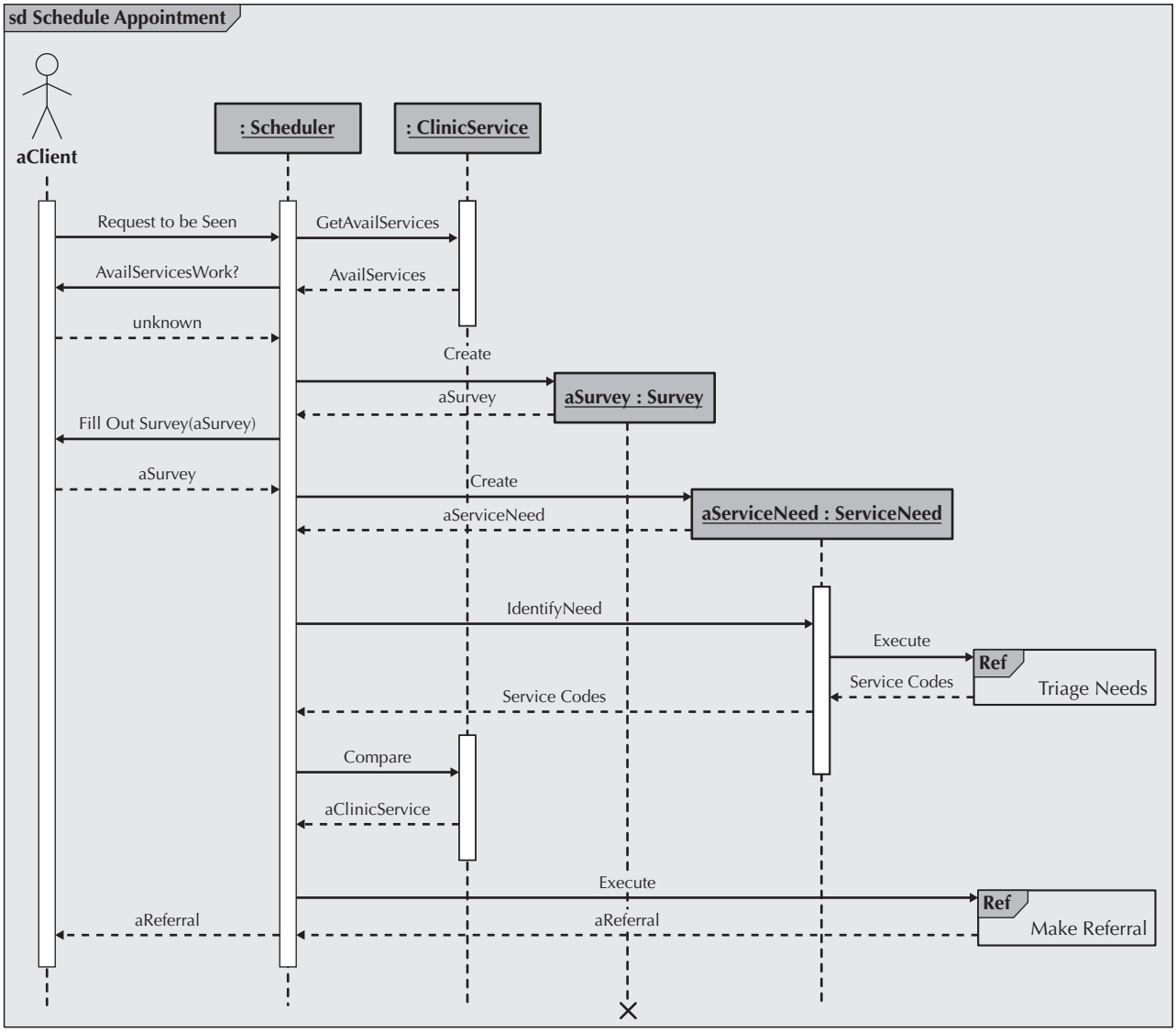


FIGURE 6-C Schedule Appointment Sequence Diagram for the filling out survey and clinic does not provide appropriate service scenario

scenarios of the Schedule Appointment use case. After drawing the diagrams, the team validated them for accuracy.

Communication Diagram

Communication diagrams also show the messages that pass between objects. While it may seem as if sequence and communication diagrams do the same thing, their focus is different and valuable due to additional information that each diagram can uncover. While sequence diagrams show the time ordering of the messages, *communication diagrams emphasize the flow of messages across objects*. The communication diagrams for the Schedule Appointment use case utilizes the same classes and has the same boundaries used in the sequence diagram.

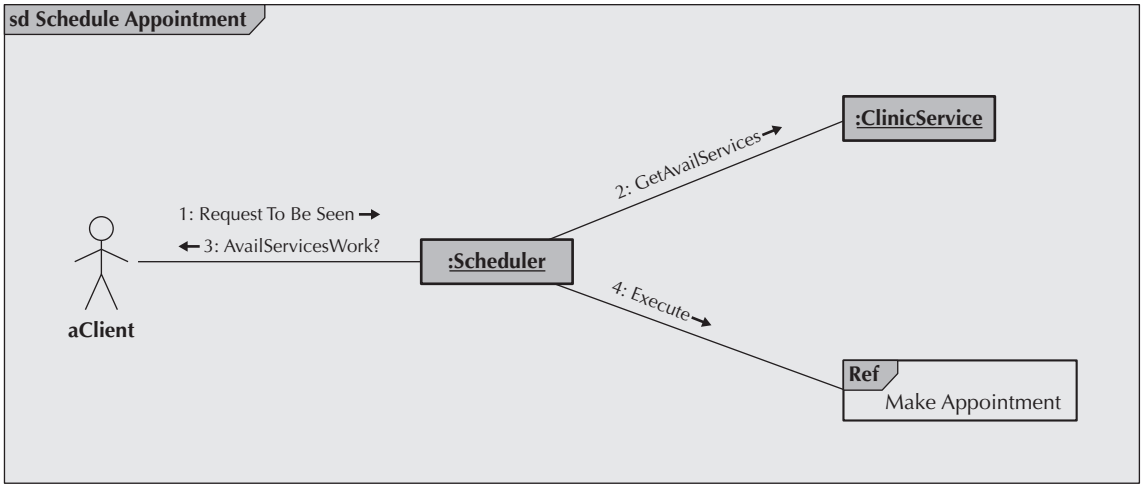


FIGURE 6-D Schedule Appointment Communication Diagram for the client chooses service scenario

However, the two diagrams look quite different. In the communication diagram, objects are positioned based on their associations with other objects in the use case and the messages are numbered. The communication diagram is validated by ensuring that it accurately portrays the Schedule Appointment use case. Figures 6-D, 6E, and 6-F show the completed communication diagrams for the same scenarios of the Schedule Appointment use case that are portrayed with the sequence diagrams in Figures 6-A, 6-B, and 6-C.

Behavioral State Machine

The third behavior diagram created by the team focuses on the changes that occur within an object. The behavioral state machine shows the transitions that an object passes through during the execution of a use case. Within the Schedule Appointment use case, an instance of the client

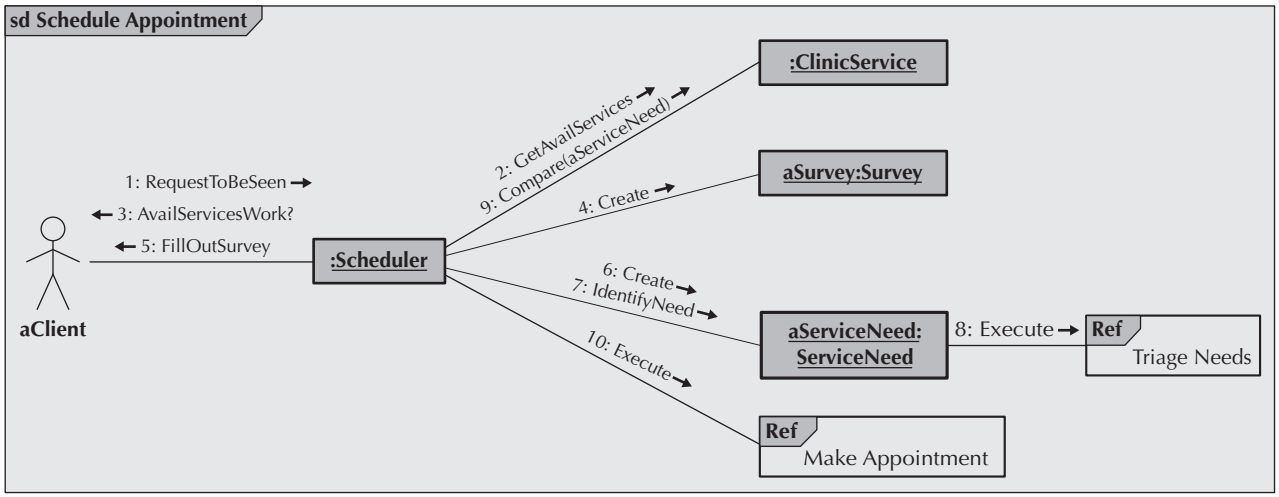


FIGURE 6-E Schedule Appointment Sequence Diagram for the filling out survey and clinic provides appropriate service scenario

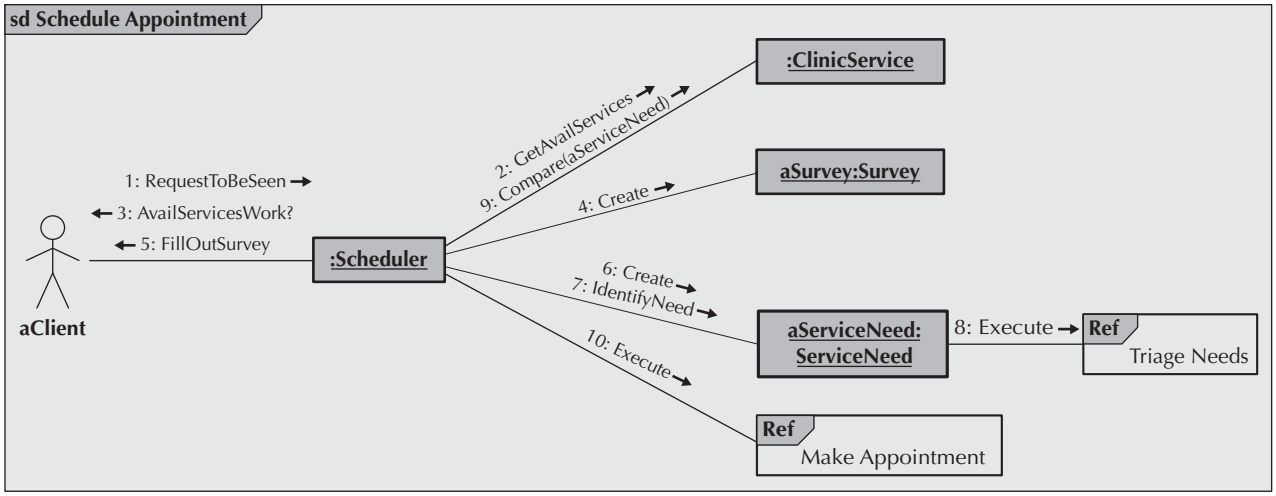


FIGURE 6-F Schedule Appointment Sequence Diagram for the filling out survey and clinic does not provide appropriate service scenario

class, aClient, transitions from inquiring, surveyed, and need defined to either referred or to be scheduled. In considering the Client class, the behavioral state machine uncovered the need for status as an additional attribute for the Client class. The behavioral state machine that the team created and validated for the client class is shown below in Figure 6-G.

CRUDE Matrix

The final step in identifying how the objects work together to collaborate in support of the use cases was to develop a CRUDE Matrix for all of Version 1. While a CRUDE Matrix is typically used to gain a system-wide view of the object in the system, this CRUDE Matrix (Figure 6-H) only depicts the operations that appear in the Schedule Appointment use case. Due to this, the majority of the cells are empty. Also, notice that the team has added the Scheduler object and included both the Client actor and the Client class. All the objects, except for Survey and Service Need objects, are instantiated elsewhere. For example, an Appointment object will be created by operations that appear in the Make Appointment use case and a Service Referral object will be created by operations associated with the Make Referral use case.

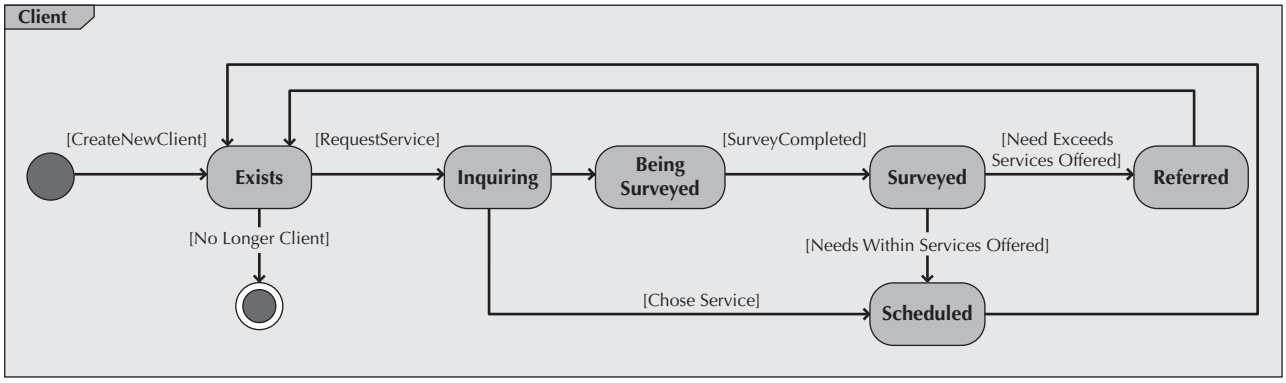


FIGURE 6-G Behavioral State Machine for the Client Class

	CLIENT ACTOR	SCHEDULER	CLIENT	SURVEY	SURVEY QUESTIONS	SERVICE NEED	CLINIC SERVICE	REFERRAL	REFERRAL LIST	APPOINTMENT	CALENDAR
CLIENT ACTOR		E		RU							
SCHEDULER	E			C		CE	RE				
CLIENT											
SURVEY					R						
SURVEY QUESTIONS											
SERVICE NEED											
CLINIC SERVICE											
REFERRAL											
REFERRAL LIST											
APPOINTMENT											
CALENDAR											

FIGURE 6-H CRUDE Matrix for Schedule Appointment use case

CHAPTER 7: PATTERSON SUPERSTORE CASE

After developing and verifying the functional, structural, and behavioral models, Ruby tasked her team with validating that the models developed in analysis agreed with each other. In other words, they needed to balance the functional, structural, and behavioral models.

Balancing the Functional, Structural, and Behavioral Models

Because the team had verified each of the analysis models independently, the newer professionals on the team questioned the need to verify and validate these models together. However, the more experienced team members assured the newbies that further validation now would ensure consistency across the analysis models and potentially save costly problems later. During this review, Ruby noticed that the current version would have a set of limitations that would need to be dealt with in future versions of the system. First, as it stands, a client can either make an appointment or receive a referral, not both. This seemed to be an acceptable position to take at this point in time, but Ruby realized that this should be fixed in future releases. Second, only current clients will be able to use the current system. A new client will have to deal with scheduling an appointment in person the first time. Given the amount of information that is required from a client to be added to the system, this seemed to be a reasonable limitation. Next, Ruby had her team begin comparing the different models.

Structural and Behavioral Balancing

As noted in the last chapter, in creating the sequence and communications diagrams, an additional class (Scheduler) was added. Also, when the team completed the communication and sequence diagrams for the Make Appointment and Make Referral use cases, they realized that both processes needed to be able to retrieve Client information. This required that the CRUDE matrix be updated to include an R in the Scheduler/Client cell that represents the Scheduler being able to read a Client object, and a relationship between the Client class and the Scheduler class is necessary. They also realized that many other cells in the CRUDE matrix needed to be updated. These changes required modifications to the class diagram, an additional CRC card for the Scheduler class, modifications to many CRC cards, and the CRUDE matrix (see Figures 7-A, 7-B, 7-C, and 7-D).

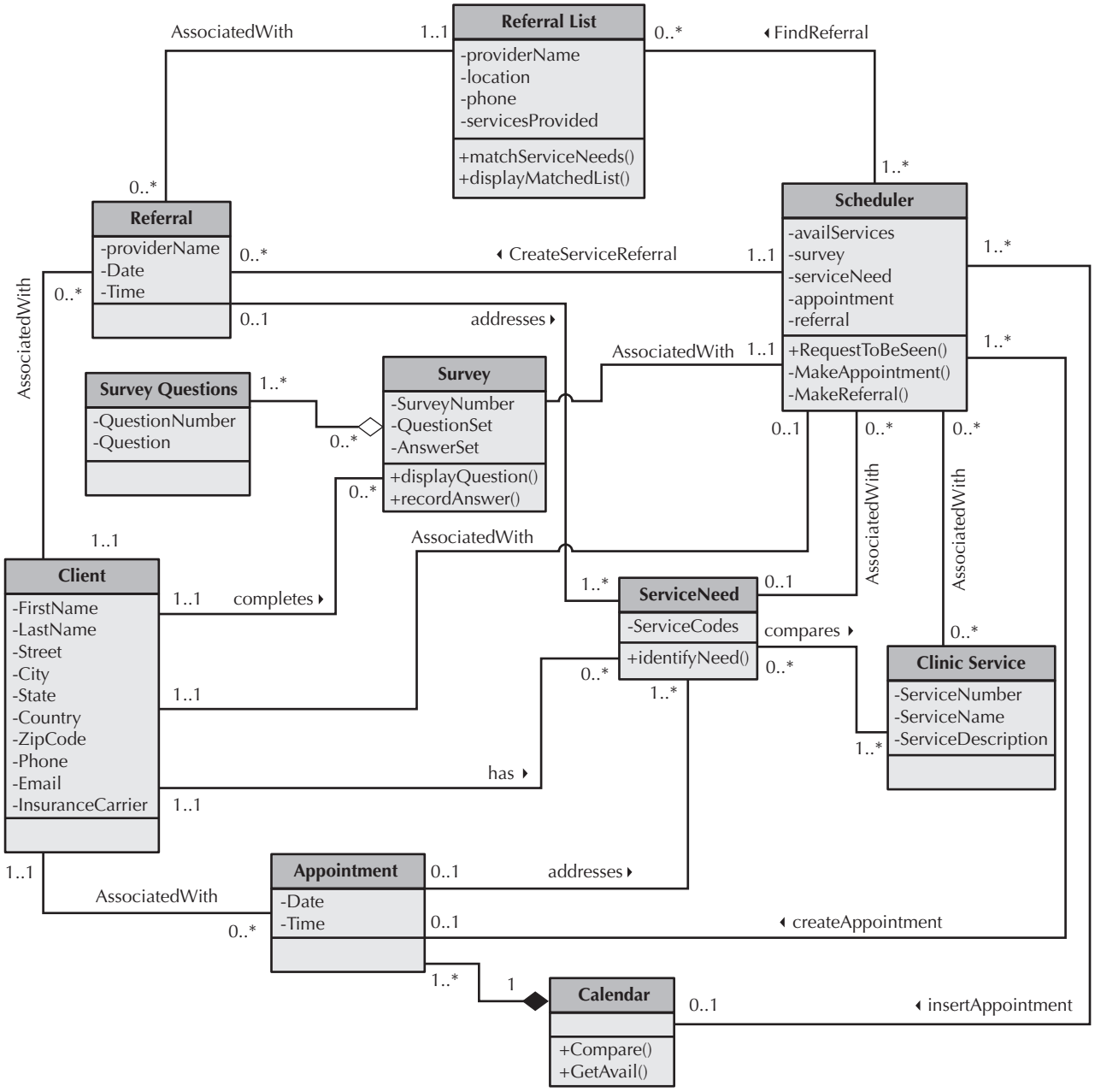


FIGURE 7-A Revised Class Diagram

FIGURE 7-B CRC Card for Scheduler Class

Front:

Class Name: Scheduler	**ID:** 10	**Type:** Concrete Domain
Description: This class acts as an intermediary between the client and the system		**Associated Use Cases:** 1, 1-1, 1-2

Responsibilities	**Collaborators**
Request to Be Seen	
Make Appointment	
Make Referral	

Back:

Attributes:

Avail Services	Appointment
Survey	Referral
Service Need	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Appointment, Service Need, Survey, Referral, Calendar, Clinic Service, Client, Referral List

FIGURE 7-C Revised CRC Cards

Front:

Class Name: Client	**ID:** 1	**Type:** Concrete Domain
Description: An individual wishing to be seen at the Patterson Superstore Health Clinic		**Associated Use Cases:** 1, 1-1, 1-2, 2,3

Responsibilities	**Collaborators**

Back:

Attributes:

First Name	Country
Last Name	Zip Code
Street	Phone
City	E-mail
State	Insurance Carrier

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Appointment, Referral, Survey, Scheduler, Service Need

Front:

Class Name: Survey	**ID:** 2	**Type:** Concrete Domain
Description: The set of questions asked and answers provided to ascertain service need		**Associated Use Cases:** 1

Responsibilities	**Collaborators**
Display Question	Survey Question
Record Answers	

Back:

Attributes:

Survey Number
Question Set
Answer Set

Relationships:

Generalization (a-kind-of):
Aggregation (has-parts): Survey Question
Other Associations: Client, Scheduler

Front:

Class Name: Service Need	**ID:** 3	**Type:** Concrete Domain
Description: Health service need of Client		**Associated Use Cases:** 1, 1-1, 1-2

Responsibilities	**Collaborators**
Identify Need	

Back:

Attributes:

Service Codes

Relationships:

Generalization (a-kind-of):
Aggregation (has-parts):
Other Associations: Scheduler, Appointment, Referral, Clinic Service, Client

FIGURE 7-C (*Continued*)

Front:

Class Name: Referral	**ID:** 4	**Type:** Concrete Domain
Description: A referral to a medical provider for services not provided by the clinic		**Associated Use Cases:** 1-2

Responsibilities	**Collaborators**

Back:

Attributes:

Provider Name	
Date	
Time	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Service Need, Client, Scheduler, Referral List

Front:

Class Name: Clinic Service	**ID:** 5	**Type:** Concrete Domain
Description: A service provided by the clinic		**Associated Use Cases:** 1, 1-1

Responsibilities	**Collaborators**

Back:

Attributes:

Service Code	
Service Name	
Service Description	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Service Need, Scheduler

FIGURE 7-C (*Continued*)

Front:

Class Name: Appointment	**ID:** 6	**Type:** Concrete Domain
Description: A client appointment with the clinic		**Associated Use Cases:** 1-1

Responsibilities	**Collaborators**

Back:

Attributes:

Date

Time

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts): Calendar

Other Associations: Client, Scheduler, Service Need

Front:

Class Name: Calendar	**ID:** 7	**Type:** Concrete Domain
Description: The list of all clinic appointments		**Associated Use Cases:** 1-1

Responsibilities	**Collaborators**
Compare	
Get Avail	

Back:

Attributes:

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts): Appointment

Other Associations: Scheduler

FIGURE 7-C (*Continued*)

Front:

Class Name: Referral List	**ID:** 9	**Type:** Concrete Domain
Description: A list of medical providers and their services that they provide not provided by clinic		**Associated Use Cases:** 1

Responsibilities	**Collaborators**
Match Service Need	Service Need
Display Matched List	

Back:

Attributes:

Provider Name	Services Provided
Location	
Phone	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Scheduler, Referral

FIGURE 7-C (*Continued*)

Functional and Behavioral Balancing

In comparing the activity diagrams and the use case descriptions (Figures 4-G, 4-H, 4-I, 4-K, 4-L, and 4-N) with the sequence and communication diagrams (Figures 6-A, 6-B, 6-C, 6-D, 6-E, and 6-F) and the updated CRUDE matrix (Figure 7-D), the team realized that nowhere had they documented that the Make Appointment and Make Referral use cases needed to retrieve the Client information. Also, the team realized that the call to the pre-existing Triage Needs use case had not been documented in the use case description and activity diagram of the Schedule Appointment use case. Consequently, the activity diagrams and use case descriptions for all three needed to be updated (Figures 7-E, 7-F, 7-G, 7-H, 7-I, and 7-J). After additional comparison and role-playing with users, and much discussion, the team determined that the sequence and communication diagrams and that the activity diagrams and the use case descriptions were in agreement.

Functional and Structural Balancing

Balancing the revised activity diagrams and use case descriptions (Figures 7-E, 7-F, 7-G, 7-H, 7-I, and 7-J) with the class diagram and related CRC cards (Figures 7-A, 7-B, and 7-C) did not uncover any additional discrepancies. As the team completed the balancing processes, they had found many inconsistencies across all of the analysis artifacts. The new professionals on the team expressed surprise and frustration that many errors were found. Ruby explained that the whole reason for balancing models was to find as many errors as possible; rather than feeling badly about these errors, she said, they should be happy about uncovering new information and correcting errors now, rather than later.

	CLIENT ACTOR	SCHEDULER	CLIENT	SURVEY	SURVEY QUESTIONS	SERVICE NEED	CLINIC SERVICE	REFERRAL	REFERRAL LIST	APPOINTMENT	CALENDAR
CLIENT ACTOR		E		RU				R		R	
SCHEDULER	E		R	C		CE	RE	C		C	RUE
CLIENT								R		R	
SURVEY					R						
SURVEY QUESTIONS											
SERVICE NEED							R				
CLINIC SERVICE											
REFERRAL						R					
REFERRAL LIST											
APPOINTMENT						R					
CALENDAR											

FIGURE 7-D Updated CRUDE Matrix

Use Case Name: Schedule Appointment	**ID:** 1	**Importance Level:** High
Primary Actor: Client	**Use Case Type:** Detail, Essential	
Stakeholders and Interests: Client wants to schedule an appointment Existing Health Clinic System provides information about clinic services		
Brief Description: This use case describes how an appointment is scheduled electronically		
Trigger: Client requests to be seen **Type:** External		
Relationships: **Association:** Client, Existing Health Clinic System **Include:** **Extend:** Make Appointment, Make Referral **Generalization:**		
Normal Flow of Events: 1. Client requests to be seen by the clinic 2. The system displays the defined service offerings list 3. Client Chooses an existing service and Executes the Make Appointment use case		
SubFlows: S-1: Determine Suitability 1. Complete service need survey questions 2. Execute pre-existing Triage Needs use case 3. Determine whether service need is within the scope of clinic's services 4. Execute Make Appointment use case		
Alternate/Exceptional Flow: 3a. Execute S-1: Determine Suitability S-1, 3a. Execute Make Referral use case		

FIGURE 7-E Updated Schedule Appointment Use Case Description

Partitioning, Packages, Factoring, Abstraction, and Refinement

As the team discussed the need for partitioning, they realized that, by using the phased approach, they had, in some respects, partitioned the system already. After reviewing the system representation for the Schedule Appointment process, they decided that this portion of the system was understandable in its current form and did not need to be partitioned. Moreover, as they reviewed the class diagrams and collaborations, they found no need to factor either by abstraction or by refinement. The newbies joked that at last the team had done something right and the entire team enjoyed this moment of fun.

Next, the team began to evolve the analysis model into a design model for the system. Up to now, only the problem domain had been studied. As development moved from WHAT the system needs to do into HOW the system will be built, environmental factors for the system are added to the evolving model. Each of the elements of the system environment (Foundation, Problem Domain, Data Management, Human Computer Interaction, and Physical Architecture) can be depicted in a layer. Figure 7-19 of the text shows a generic Package Diagram showing the layers and their dependency relationships.

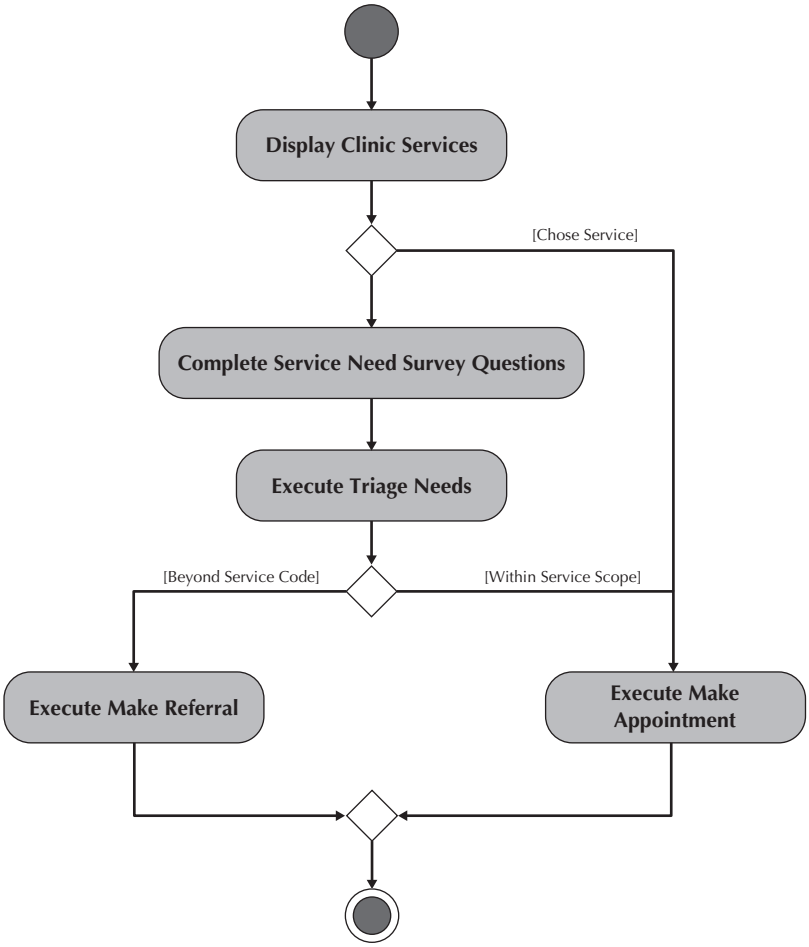


FIGURE 7-F Updated Activity Diagram for the Schedule Appointment Use Case

Design Strategies

The next step was to determine a design strategy based on three possible alternatives: custom develop in-house, purchase packaged software, or outsource the project to an external vendor. Because the Integrated Health Clinic Delivery will be one of the first such systems, no commercial systems have been developed with the features that this system requires. For this reason, purchasing packaged software is not an option. Furthermore, since this project is perceived as a means of further establishing Patterson's competitive edge in mobile service delivery, outsourcing was eliminated due to the likelihood of exporting rather than growing Patterson's expertise and advantage over less technically advanced competitors.

Custom development was chosen as the design strategy for multiple reasons. Most importantly, in-house development will enable the project to be built to the exact specifications of Patterson Superstore and allow complete control in-house. In addition, this project will utilize and increase the existing expertise within the IT department. The IT department staff designed, developed, and maintains the sophisticated prescription fulfillment system already in place at

Use Case Name: Make Appointment	**ID:** 1-1	**Importance Level:** High
Primary Actor: Client	**Use Case Type:** Detail, Essential	
Stakeholders and Interests: Client wants to schedule an appointment Administrative Staff provides wait time information, updates calendar, and sends confirmation Existing Health Clinic System Service supplies information about appointment availability		
Brief Description: This use case describes how the client chooses an appointment		
Trigger: Client wishing to schedule a clinic appointment has service needs that match clinic capability **Type:** External		
Relationships: **Association:** Client, Admin Staff, Existing Health Clinic System **Include:** **Extend:** **Generalization:**		
Normal Flow of Events: 1. Display current health clinic traffic/wait time appointment availability to client 2. Client enters appointment preference date/time 3. Client's appointment preference is checked against appointment availability 4. Matching appointment availability displayed 5. Client chooses available and desired appointment 6. Retrieve Client Information 7. Calendar is updated 8. Appointment confirmation is sent to client		
SubFlows:		
Alternate/Exceptional Flow: 5a. If no match occurs, client iterates steps 2 through 5 until satisfactory time is found		

FIGURE 7-G Updated Make Appointment Use Case Description

Patterson and can leverage that experience in creating the proposed system. Moreover, the IT department has enthusiastically moved toward RAD and Agile Methodologies and views familiarity with these methodologies as a strategic advantage. This project provides a way to further grow that expertise.

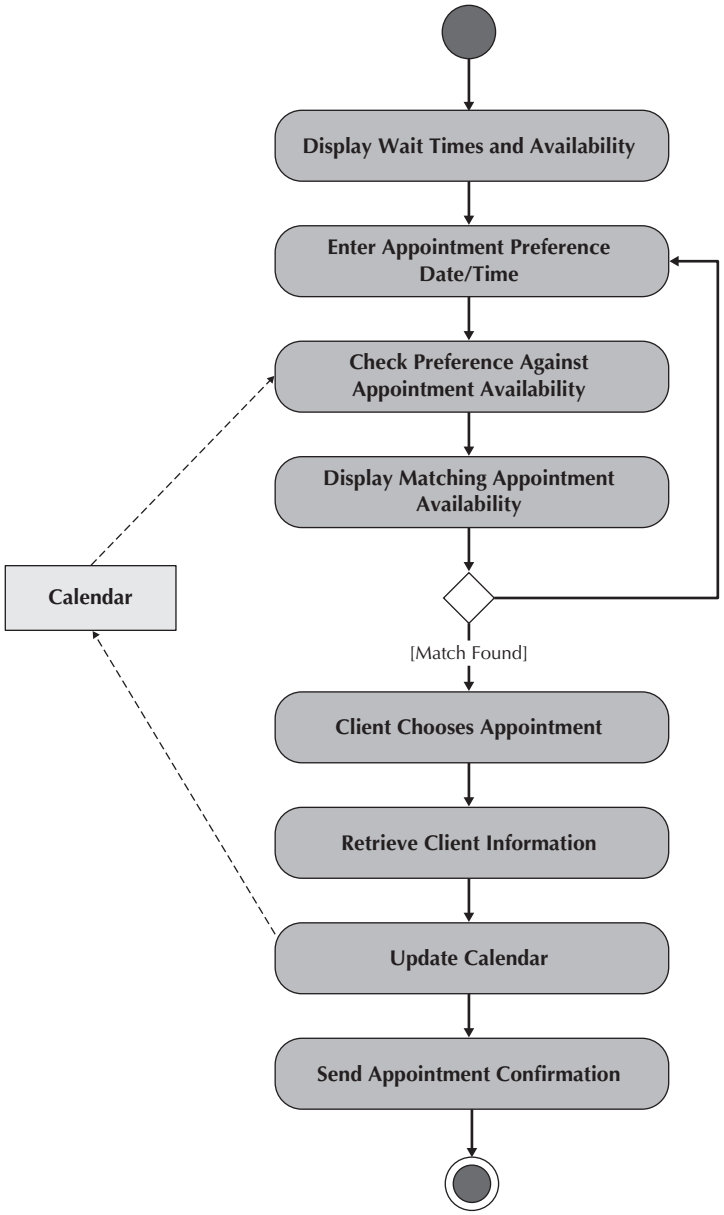


FIGURE 7-H Updated Activity Diagram for the Make Appointment Use Case

Use Case Name: Make Referral	**ID:** 1-2	**Importance Level:** High
Primary Actor: Client	**Use Case Type:** Detail, Essential	
Stakeholders and Interests: Client needs a referral Existing Health Clinic System Service provides information about referrals		
Brief Description: This use case describes how referrals are handled		
Trigger: Client need cannot be met by clinic **Type:** External		
Relationships: **Association:** Client, Existing Health Clinic System **Include:** **Extend:** **Generalization:**		
Normal Flow of Events: 1. List Referral Information 2. Compare and evaluate referral need against referral list 3. Retrieve Client information 4. Create referral		
SubFlows:		
Alternate/Exceptional Flows:		

FIGURE 7-I Revised Make Referral Use Case Description

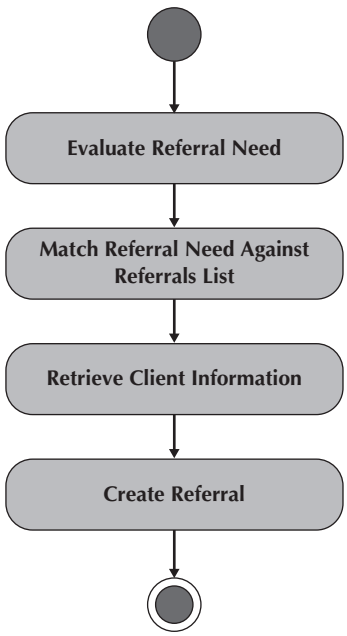


FIGURE 7-J Updated Activity Diagram for the Make Referral Use Case

CHAPTER 8: PATTERSON SUPERSTORE CASE

With balanced models and a design strategy in place, the team was eager to move into design. However, before going forward, they once again had to go back to the functional, structural, and behavioral models to ensure that the classes defined in analysis (the problem domain layer) are both sufficient and necessary. As we have said, it is vital to continually review the evolving system for accuracy. Again this is part of the incremental and iterative nature of object-oriented design and development. Ruby tasked the team members to ensure that connascence is minimized at all levels of the design. They began the detailed object design process by reviewing the class diagram for the problem domain layer (Figure 7-A). Ruby made it clear that the team should be aware of the cohesion, coupling, and connascence design criteria and review the models with those in mind.

The review of the sequence and communication diagrams (Figures 6-A, 6-B, 6-C, 6-D, 6-E, and 6-F) revealed some coupling related to the Scheduler class sending messages to other classes to perform operations on its behalf. However, all of the interaction coupling uncovered was acceptable. In reviewing for cohesion, they found that at the method level, only functional method cohesion existed; that is, each of the methods performed only a single problem-related task. Furthermore, all classes possessed ideal class cohesion. Since both the coupling and cohesion are of the "good" type, connascence was felt to be minimal.

The next step was to review the attributes and methods of the classes to ensure that no needed attribute or method was left out and to that make sure no attribute or method was included that are not needed. After careful review, the team decided that the current version of the system representation was correct.

Now that the team was completely satisfied with the current representation, the team moved on to more fully specifying the design of the problem domain. Specifically, the team detailed the constraints and algorithms by updating the CRC cards and creating contracts and method specifications.

Using Figure 8-19 in the textbook as a guide, the team updated all CRC cards to include the multiplicity, data type, and OCL (when necessary) for each attribute and the multiplicity that is shown on the class diagram with the relationships. For example, Figure 8-A portrays the updated CRC card for the Scheduler class. Notice all of the additional constraint information contained on the card.

Contracts

A contract formalizes the interactions between client and server objects. In the Schedule Appointment use case, an instance of the client actor sends a request to be seen message to the Scheduler (server) object and the Scheduler object executes a method in response to this request.

Where the CRC cards focus on modeling invariants, contracts document any preconditions and/or postconditions that must be met for the method to successfully complete. Contracts, along with the CRC cards, are used to provide the user (client) of the method a set of expectations that the server object guarantees to occur if the constraints are met. Using Figures 8-22 and 8-25 as guides, the team created the contract for the Request to be Seen method (Figure 8-B). While only this example of a contract is shown here, the team created a contract for each method that receives messages from other objects. That is, a contract would be created for each visible method.

Method Specification

Method specifications contain the exact instruction that the programmers will need to write the code that implement the method. Typically, analysts write the method specification and give it

Front:

Class Name: Scheduler	**ID:** 10	**Type:** Concrete Domain
Description: This class acts as an intermediary between the client and the system		**Associated Use Cases:** 1, 1-1, 1-2

Responsibilities	**Collaborators**
Request To Be Seen	
Identify Need	
Make Appointment	
Make Referral	

Back:

Attributes:

Avail Services (0..*) Clinic Service (avail Services = Clinic Service.ServiceCode)

survey (0..1) Survey

service Need (0..1) Service Need

appointment (0..1) Appointment

referral(0..*) Referral

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Appointment (0..1), Service Need (0..1), Survey (0..1), Referral (0..*), Clinic Service (0..*), Client (1..1), Referral List (0..*), Calendar (0..1)

FIGURE 8-A Updated CRC Card for Scheduler Class

Method Name: Request to Be Seen	**Class Name:** Scheduler	**ID:** 1
Clients (Consumers): Client Actor		
Associated Use Cases: Schedule Appointment		
Description of Responsibilities: Client requests to be seen at a Patterson Health Clinic		
Arguments Received: None		
Type of Value Returned: Either an Appointment or a Referral object		
Pre-Conditions: Client must already exist in the Patterson Health Clinic system		
Post-Conditions: None		

FIGURE 8-B Contract for Request to be Seen Method

to the programmers so that they can write the code. A method specification is written for each and every method. Using Figures 8-26, 8-29, 8-30, 8-31, and 8-32 as examples, the team created a method specification for the Request to be Seen method (Figures 8-C and 8-D). When the team completed the specification for this method, they realized that they had to modify a set of CRC cards and the class diagram again. Given that the implementation language was to be Java, they realized that the method could not return two separate types of variables: Appointment object or Referral object. Virtually all object-oriented programming languages limit a method signature to a single data type for a method's return type. Consequently, the team had to create an abstract class (Medical Engagement) that could serve as an abstraction of both types of objects: Appointment and Referral. The new CRC card for the Medical Engagement class is shown in Figure 8-E. The update CRC cards for the Appointment, Referral, and Scheduler classes are shown in Figure 8-F, and the updated class diagram is shown in Figure 8-G. As with the contract example (Figure 8-B), this is the only method specification shown for this example. However, you should realize that a method specification was created for every method in the system.

Method Name: Request to Be Seen	**Class Name:** Scheduler	**ID:** 1
Contract ID: 1	**Programmer:** Alice Smith	**Date Due:** 12-10-15
Programming Language: ❒ **Visual Basic** ❒ **Smalltalk** ❒ **C++** ☑ **Java**		
Triggers/Events: Client requests to schedule an appointment		

Arguments Received: **Data Type:**	**Notes:**

Messages Sent & Arguments Passed: ClassName. MethodName:	**Data Type:**	**Notes:**
ClientActor.AvailServicesWork?(List)	List	List Available Clinic Services
ClientActor.FillOutSurvey(Survey)	Survey	
ClinicService.GetAvailServices		Get a list of available clinic services
ClinicService.Compare(ServiceNeed)	ServiceNeed	Compare ServiceNeed to available clinic services
ServiceNeed.Create		
ServiceNeed.IdentifyNeed(Survey)	Survey	Pass along survey results to pre-existing Triage Needs use case to determine needs
Survey.Create		

Argument Returned: **Data Type:**	**Notes:**
Medical Engagement	If Patterson can address the service needs identified, an Appointment object is returned otherwise a Referral object will be returned instead.

Algorithm Specification: See Figure 8-D.
Misc. Notes:

FIGURE 8-C Method Specification for Request to be Seen Method

```
Execute the Get Avail Services method in Clinic Service object
Request Client to either select an available service from the list or select unknown (default)
If Client selects a service from the list
        Execute the Make Appointment process
Else
        Create a Survey object
        Request Client to fill out the survey
        Create a Service Need object
        Execute the identify Need method in the Service Need object
        Execute the Compare method in the Clinic Service object (pass the Service Need object
                to the Compare Method)
        If Clinic provides required service
                Execute Make Appointment process
        Else
                Execute Make Referral Process
        Endlf
Endlf
Return Medical Engagement object
```

FIGURE 8-D Algorithm Specification for Request to be Seen Method

Front:

Class Name: Medical Engagement	**ID:** 11	**Type:** Abstract Domain
Description: An abstraction of the Appointment and Referral classes		**Associated Use Cases:** 1-1, 1-2

Responsibilities	**Collaborators**

Back:

Attributes:

Date (1..1) Date	
Time (1..1) Time	

Relationships:

Generalization (a-kind-of): Appointment, Referral

Aggregation (has-parts):

Other Associations: Service Need (1..*), Client (1..1), Scheduler (1..1)

FIGURE 8-E New CRC card for Medical Engagement class

FIGURE 8-F
Updated CRC cards for Appointment, Referral, Client, and Scheduler classes

Front:

Class Name: Appointment	**ID:** 6	**Type:** Concrete Domain
Description: A client appointment with the clinic		**Associated Use Cases:** 1-1

Responsibilities	**Collaborators**

Back:

Attributes:

Relationships:

Generalization (a-kind-of): Medical Engagement

Aggregation (has-parts): Calendar (1)

Other Associations:

Front:

Class Name: Referral	**ID:** 4	**Type:** Concrete Domain
Description: A referral to a medical provider for services not provided by the clinic		**Associated Use Cases:** 1-2

Responsibilities	**Collaborators**

Back:

Attributes:

Provider Name (1..1) String

Relationships:

Generalization (a-kind-of): Medical Engagement

Aggregation (has-parts):

Other Associations: Referral List (1..1)

FIGURE 8-F
(*Continued*)

Front:

Class Name: Client	**ID:** 1	**Type:** Concrete Domain
Description: An individual wishing to be seen at the Patterson Superstore Health Clinic		**Associated Use Cases:** 1, 1-1, 1-2, 2,3

Responsibilities	Collaborators

Back:

Attributes:

FirstName (1..1) String	Country(1..1) String
Last Name (1..1) String	Zip Code (1..1) ZipCode
Street (1..1) String	Phone (1..1) Phone Number
City (1..1) String	E-mail (1..1) E-mail Address
State (1..1) String	Insurance Carrier (1..1) String

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Medical Engagement (0..*), Survey (0..*), Scheduler (0..1), Service Need (0..*)

Front:

Class Name: Scheduler	**ID:** 10	**Type:** Concrete Domain
Description: This class acts as an intermediary between the client and the system		**Associated Use Cases:** 1, 1-1, 1-2

Responsibilities	Collaborators
Request To Be Seen	
Make Appointment	
Make Referral	

Back:

Attributes:

availServices (0..*) Clinic Service	(availServices = Clinic Service.ServiceCode)
survey (0..1) Survey	
serviceNeed (0..1) Service Need	
appointment (0..1) Appointment	
referral(0..*) Referral	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Service Need (0..1), Survey (0..1), Calendar (0..1), Medical Engagement (0..*), Clinic Service (0..*), Client (1..1),Referral List (0..*)

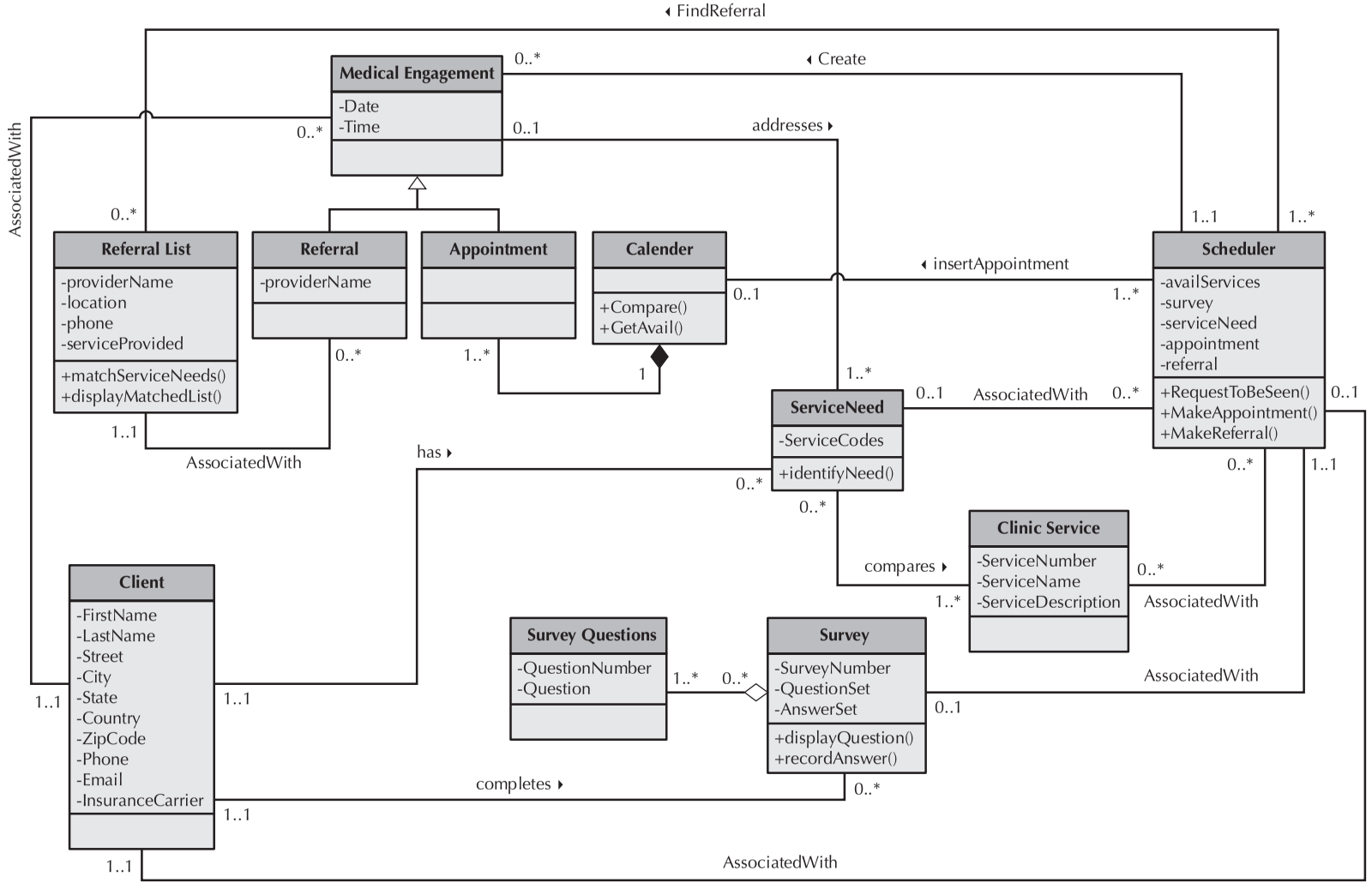


FIGURE 8-G Updated Class Diagram

CHAPTER 9: PATTERSON SUPERSTORE CASE

After developing functional, structural, and behavior models; designing contracts and method specifications; and checking for coupling, cohesion, and connascence, the team had thoroughly explored and defined the problem domain. It was now time to plan the solution domain, including data management, human–computer interaction, and physical architecture design.

To encourage parallel development, Ruby split the team into three separate groups: one for the data management layer, one for the human–computer interaction layer, and one for the physical architecture layer.

Ruby tasked Ben Joseph, a data analytics specialist, to lead the data management layer design. Ben identified four specific tasks relating to the design of the data management layer. The first task was to select the object-persistence format. Given that Patterson uses a relational database for most of its corporate data needs, Ben felt that the best solution would be to leverage this expertise and to use a relational database system (RDBMS). However, after discussions with Ruby and the development team, Ben thought that the team should consider using the object-relational extensions that the current RDBMS supported.

Given that both the existing prescription fulfillment system and the new system utilize Java, the object-relational solution would enable Patterson to leverage the data management expertise that currently existed at Patterson and support the inherent object-oriented architecture of the current design (see Figures 9-A and 9-B) better than the straight RDBMS. Also, Ben reminded the team that only concrete classes that had application data associated with them needed to be stored in the database. Consequently, the Scheduler class, which only acted as an actor or intermediary between the client actor and the system, did not need to be stored. This realization meant that every sequence diagram created that included a Scheduler object would need to be modified by adding an "X" at the bottom of the Scheduler object's lifeline to show that it went out of existence after the use case was completed (see Figure 9-C). Furthermore, since the Survey object's lifeline also had an "X" at the bottom of it (see Figure 9-C), it too did not need to be stored. Finally, Ben decided that since instances of the Survey class did not need to be stored and that instances of the Survey Question class was only associated with instances of the Survey class, instances of the Survey Question class did not need to be stored either.

After talking with Ruby and the team, Ben decided to ask Jo, Patterson's database administrator (DBA), to be part of the data management layer design team. Furthermore, being the DBA, she was very familiar with the integrated database that currently supported Patterson's other systems. Ben felt that this knowledge would be invaluable. Additionally, before becoming Patterson's DBA, Jo had worked on multiple projects for another firm that dealt with Java, mobile technology, and the object-relational extension to the RDBMS that was used by Patterson. Consequently, she was a natural and necessary addition to the team.

However, before Jo joined the team, Ruby and Ben decided to go ahead and perform the second step: mapping the problem domain classes into the object persistence format chosen. Given that the team decided to use an object-relational database management system (ORDBMS), they decided that the team should follow the schema mapping rules in Figure 9-7 and to use Figure 9-8 as an example to follow. Using Rule 1, Ben identified the Client, Service Need, Referral, Clinic Service, Appointment, Calendar, Referral List, and Medical Engagement problem domain classes that should be associated with ORDBMS tables. Using Rules 2 through 8, Ben mapped these classes to ORDBMS tables (see Figure 9-D). Notice

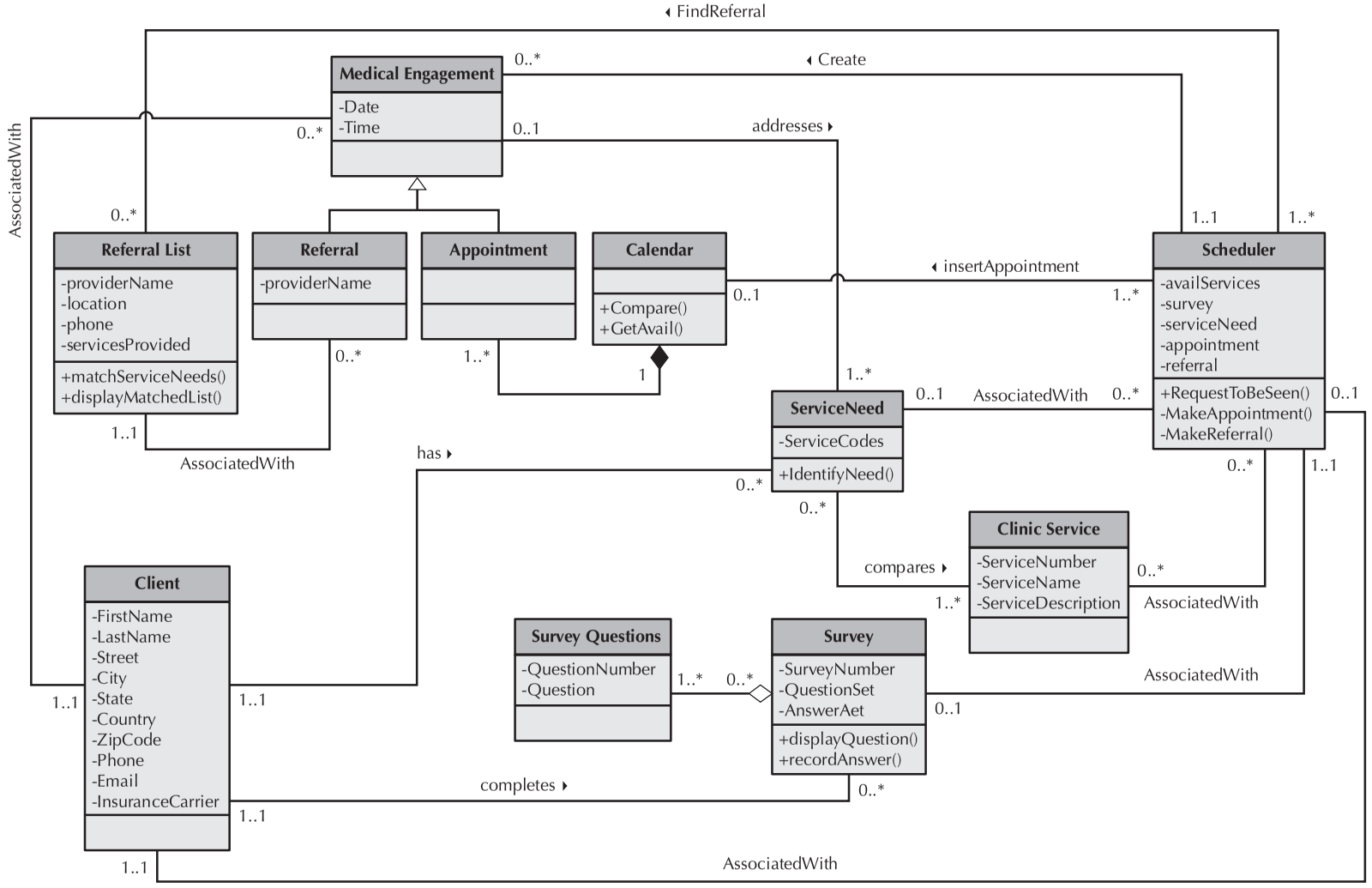


FIGURE 9-A Class Diagram

FIGURE 9-B
CRC Cards

Front:

Class Name: Client	**ID:** 1	**Type:** Concrete Domain
Description: An individual wishing to be seen at the Patterson Superstore Health Clinic		**Associated Use Cases:** 1, 1-1, 1-2, 2,3

Responsibilities	**Collaborators**

Back:

Attributes:

FirstName(1..1) String	Country(1..1) String
Last Name (1..1) String	Zip Code (1..1) ZipCode
Street (1..1) String	Phone (1..1) Phone Number
City (1..1) String	E-mail (1..1) E-mailAddress
State (1..1) String	Insurance Carrier (1..1) String

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Medical Engagement (0..*), Survey (0..*), Scheduler (0..1), ServiceNeed (0..*)

Front:

Class Name: Survey	**ID:** 2	**Type:** Concrete Domain
Description: The set of questions asked and answers provided to ascertain service need		**Associated Use Cases:** 1

Responsibilities	**Collaborators**
Display Question	Survey Question
Record Answers	

Back:

Attributes:

Survey Number_(1..1) Integer

QuestionSet (1..*) Integer

AnswerSet(1..*) String

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts): Survey Question (1..*)

Other Associations: Client (1..1), Scheduler (1..1)

FIGURE 9-B
(*Continued*)

Front:

Class Name: Service Need	**ID:** 3	**Type:** Concrete Domain
Description: Health service need of Client		**Associated Use Cases:** 1, 1-1, 1-2

Responsibilities	**Collaborators**
IdentifyNeed	

Back:

Attributes:

Service Codes (0..*) Integer

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Scheduler (0..*), Medical Engagement (0..1), Clinic Service (1..*), Client (1..1)

Front:

Class Name: Referral	**ID:** 4	**Type:** Concrete Domain
Description: A referral to a medical provider for services not provided by the clinic		**Associated Use Cases:** 1-2

Responsibilities	**Collaborators**

Back:

Attributes:

Provider Name (1..1) String

Relationships:

Generalization (a-kind-of): Medical Engagement

Aggregation (has-parts):

Other Associations: Referral List (1..1)

FIGURE 9-B
(*Continued*)

Front:

Class Name: Clinic Service	**ID:** 5	**Type:** Concrete Domain
Description: A service provided by the clinic		**Associated Use Cases:** 1, 1-1

Responsibilities	Collaborators

Back:

Attributes:

Service Code (1..1) String	
Service Name (1..1) String	
Service Description (1..1) String	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Service Need (0..*), Scheduler (0..*)

Front:

Class Name: Appointment	**ID:** 6	**Type:** Concrete Domain
Description: A client appointment with the clinic		**Associated Use Cases:** 1-1

Responsibilities	Collaborators

Back:

Attributes:

Relationships:

Generalization (a-kind-of): Medical Engagement

Aggregation (has-parts): Calendar (1)

Other Associations:

FIGURE 9-B
(*Continued*)

Front:

Class Name: Calendar	**ID:** 7	**Type:** Concrete Domain
Description: The list of all clinic appointments		**Associated Use Cases:** 1-1

Responsibilities	**Collaborators**
Compare	
Get Avail	

Back:

Attributes:

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts): Appointment (1..*)

Other Associations: Scheduler (1..*)

Front:

Class Name: Survey Question	**ID:** 8	**Type:** Concrete Domain
Description: A Survey question used to help determine needed services		**Associated Use Cases:** 1

Responsibilities	**Collaborators**

Back:

Attributes:

Question Number (1..1) Integer

Question (1..1) String

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Survey (0..*)

FIGURE 9-B
(*Continued*)

Front:

Class Name: Referral List	**ID:** 9	**Type:** Concrete Domain
Description: A list of medical providers and their services that they provide not provided by clinic		**Associated Use Cases:** 1

Responsibilities	Collaborators
Match Service Need	Service Need
Display Matched List	

Back:

Attributes:

Provider Name (1..1) String	Services Provided (1..*) String
Location (1..1) Address	
Phone (1..1) PhoneNumber	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Scheduler (1..*), Referral (0..*)

Front:

Class Name: Scheduler	**ID:** 10	**Type:** Concrete Domain
Description: This class acts as an intermediary between the client and the system		**Associated Use Cases:** 1, 1-1, 1-2

Responsibilities	Collaborators
Request To Be Seen	
Make Appointment	
Make Referral	

Back:

Attributes:

availServices (0..*) Clinic Service	(avail Services = Clinic Service. ServiceCode)
survey (0..1) Survey	
service Need (0..1) Service Need	
appointment (0..1) Appointment	
referral (0..*) Referral	

Relationships:

Generalization (a-kind-of):

Aggregation (has-parts):

Other Associations: Service Need (0..1), Survey (0..1), Calendar (0..1)
Medical Engagement (0..*), Clinic Service (0..*),
Client (1..1), Referral List (0..*)

FIGURE 9-B
(*Continued*)

Front:

Class Name: Medical Engagement	**ID:** 11	**Type:** Concrete Domain
Description: An abstraction of the Appointment and Referral classes		**Associated Use Cases:** 1-1, 1-2

Responsibilities	**Collaborators**

Back:

Attributes:

Date (1..1) Date

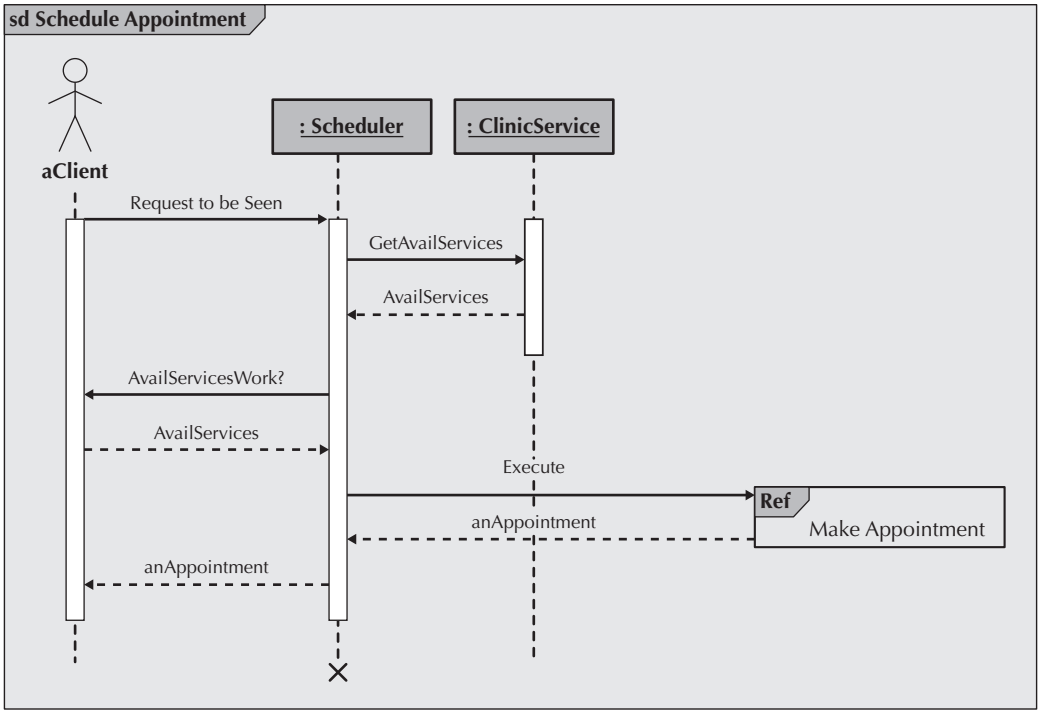
Time (1..1) Time

Relationships:

Generalization (a-kind-of): Appointment, Referral

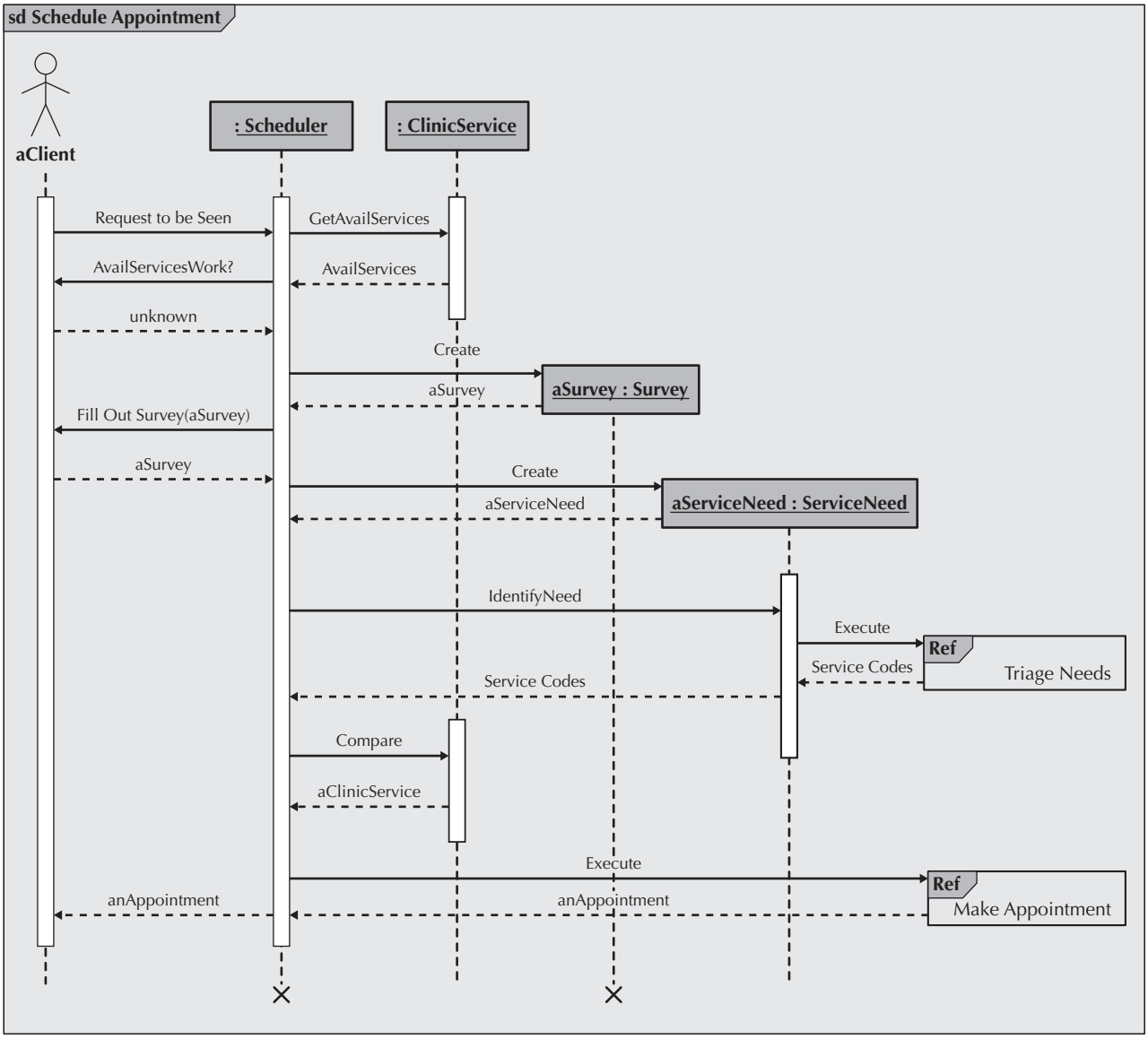
Aggregation (has-parts):

Other Associations: Service Need (1..*), Client (1..1), Scheduler (1..1)



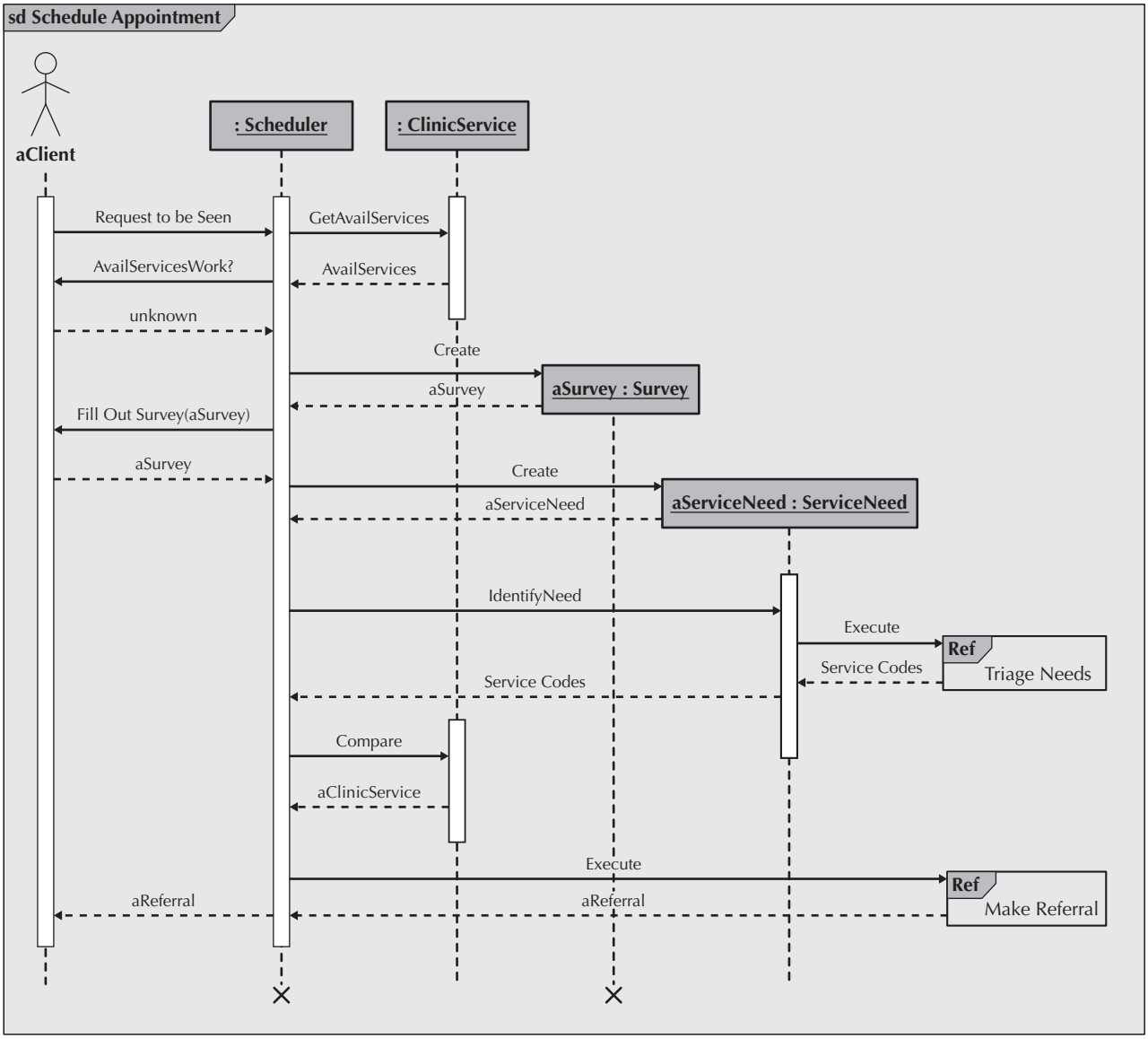
Schedule Appointment Sequence Diagram for the client chooses service scenario

FIGURE 9-C Sequence Diagrams for Schedule Appointment Use Case



Schedule Appointment Sequence Diagram for the filling out survey and clinic provides appropriate service scenario

FIGURE 9-C (*Continued*)



Schedule Appointment Sequence Diagram for the filling out survey and clinic does not provide appropriate service scenario

FIGURE 9-C (*Continued*)

that all of the Association-based relationships have been converted to attributes in the ORDBMS tables.

Next, he had to tackle the question of the inheritance between the Medical Engagement superclass and the Appointment and Referral subclasses. Ruby explained to Ben that the Medical Engagement superclass was abstracted from the Appointment and Referral classes so that the Request to be Seen method (see Figures 8-C and 8-D) of the Scheduler class could return either an instance of the Appointment or Referral classes to the Client Actor (see Figure 9-C). Given this discussion, Ben decided that instances of the Medical Engagement class did not need to be stored. So, he chose to apply Rule 9b instead of 9a. Figure 9-E shows

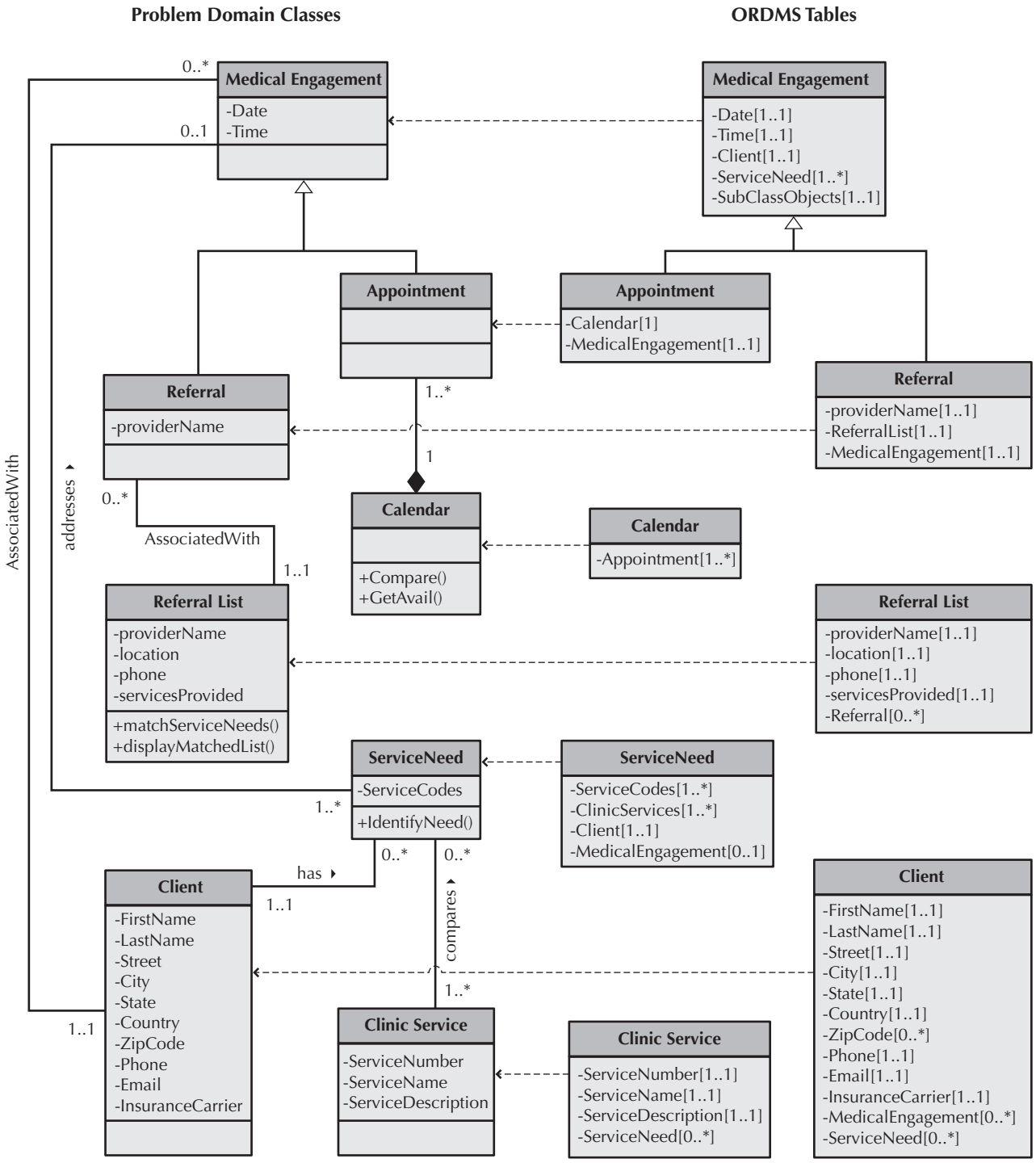


FIGURE 9-D Problem Domain class to ORDBMS Table Mappings

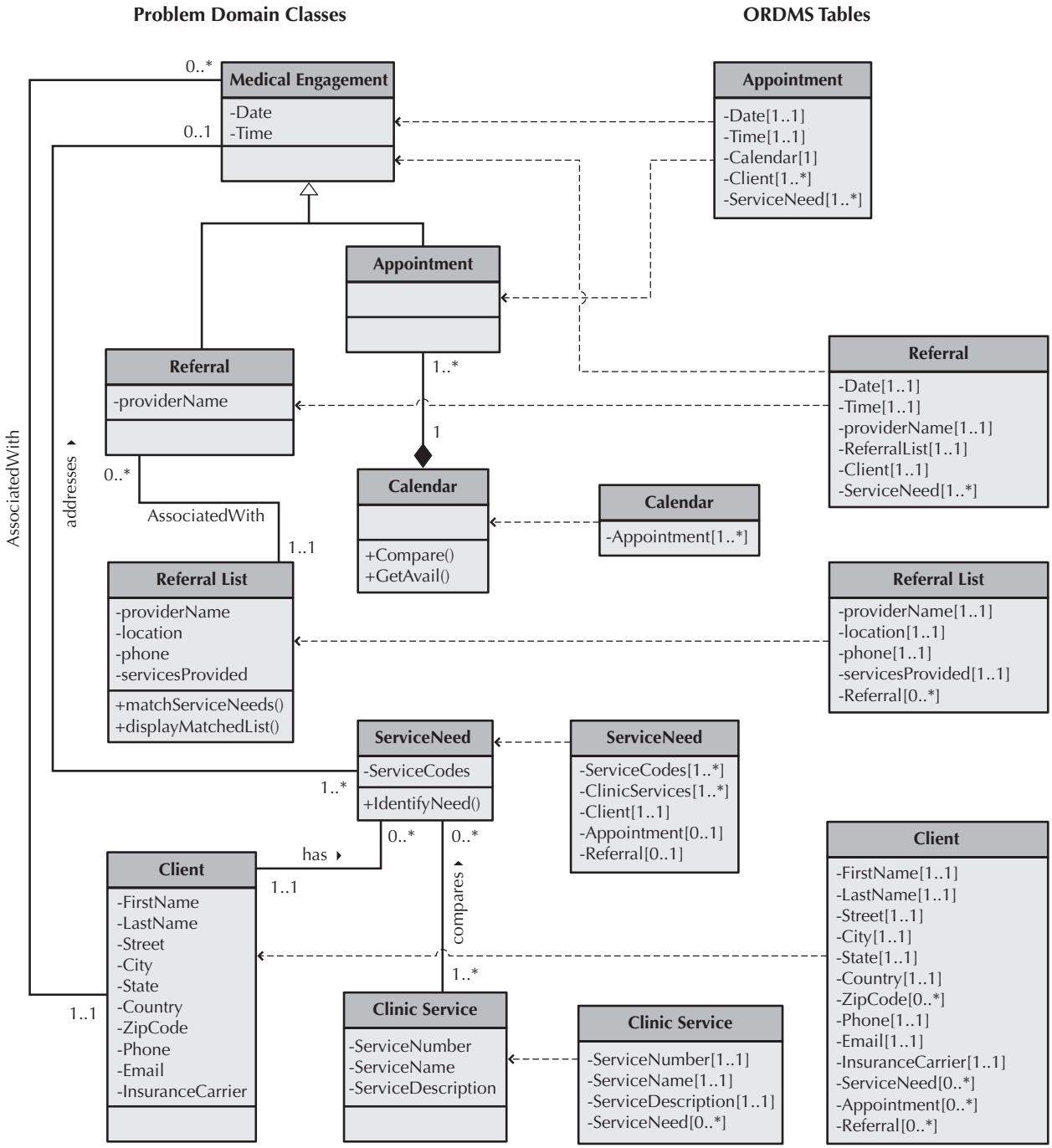
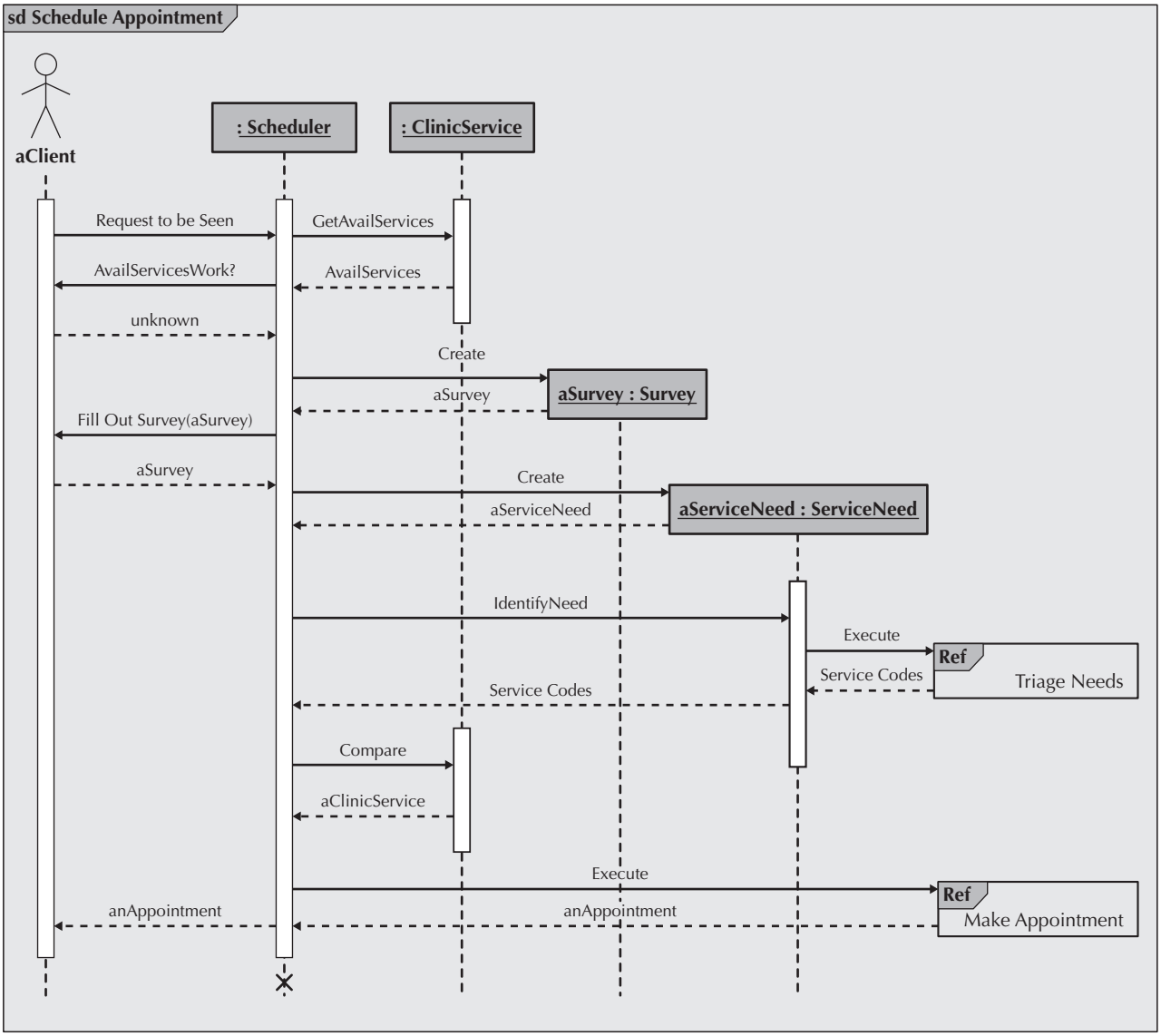


FIGURE 9-E Updated Problem Domain class to ORDBMS Table Mappings

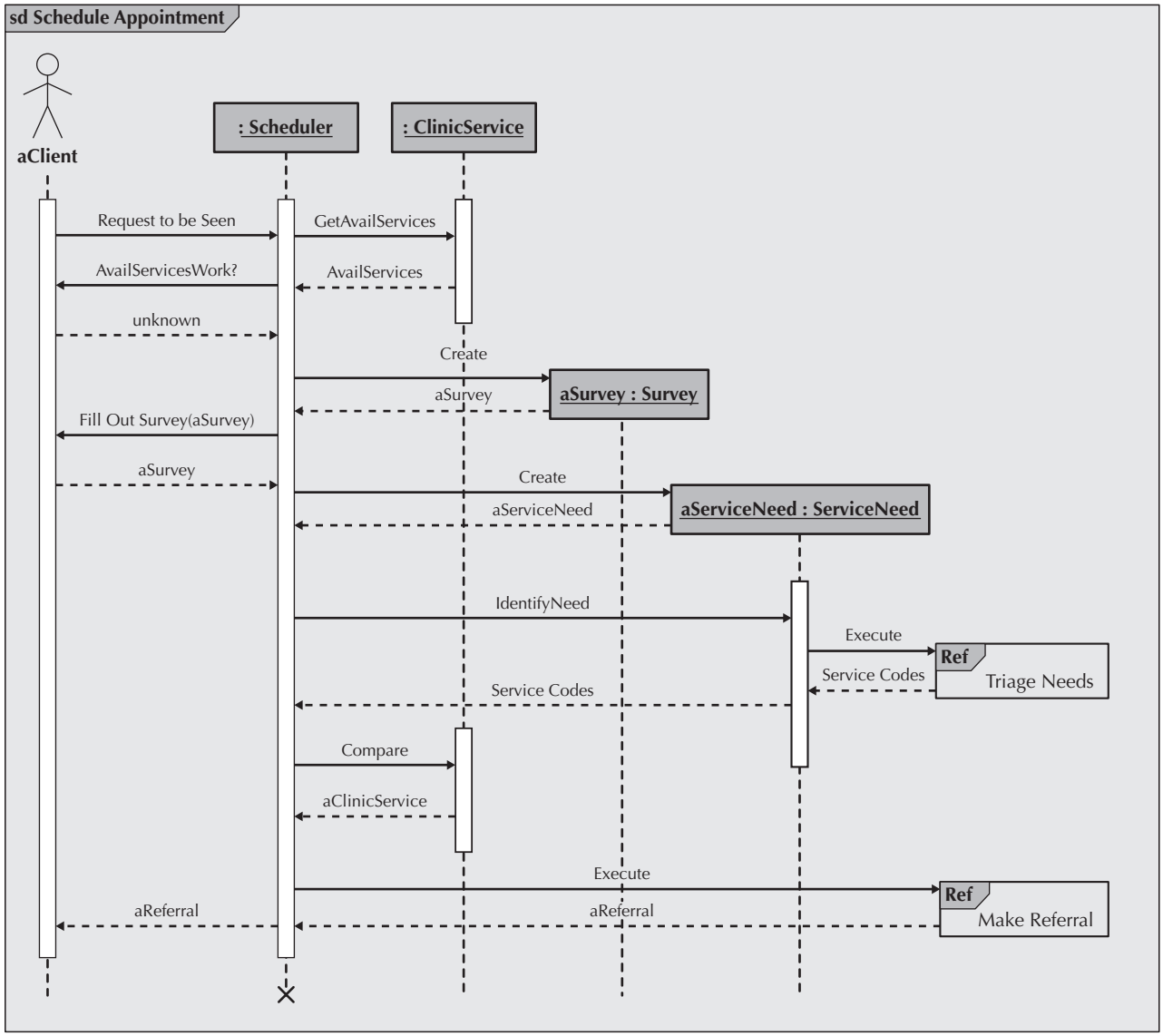
the mapping between the problem domain classes and the ORDBMS tables with the inheritance relationships and the ORDBMS table for the Medical Engagement abstract class removed. Also, notice that the Medical Engagement attribute has been changed to an Appointment and a Referral attribute in both the Client and ServiceNeed ORDBMS tables.

At the next team meeting, Ben provided both the problem domain class design (see Figures 9A, 9B, and 9C) and the updated problem domain classes to ORDBMS tables mappings (see Figure 9-E). After reviewing the figures, Jo congratulated the team for doing a very good job with the mappings. However, she did have a couple of questions that needed to be answered before the ORDBMS design could be completed. First, she questioned the decision not to store instances of the Survey class. She pointed out that in the current system, the survey results are stored. After discussing this with Ruby and Ben, the team decided that storing the results was the smarter thing to do. This decision caused the sequence diagrams for two of the scenarios of the Schedule Appointment use case to be modified again (see Figure 9-F). Also, due to the



Schedule Appointment Sequence Diagram for the filling out survey and clinic provides appropriate service scenario

FIGURE 9-F Updated Sequence Diagrams for Schedule Appointment Use Case



Schedule Appointment Sequence Diagram for the filling out survey and clinic does not provide appropriate service scenario

FIGURE 9-F (*Continued*)

repeating group in instances of the Survey class (the Question Set and Answer Set attributes), a new ORDBMS table had to be created (SurveyQ&A) and the ORDBMS design to be modified (see Figure 9-G).

Second, she pointed out that, with a little more effort, every ORDBMS table that had been designed could have been mapped into a set of RDBMS tables. In fact, the set of equivalent RDBMS tables already existed in the existing health clinic system. However, after much

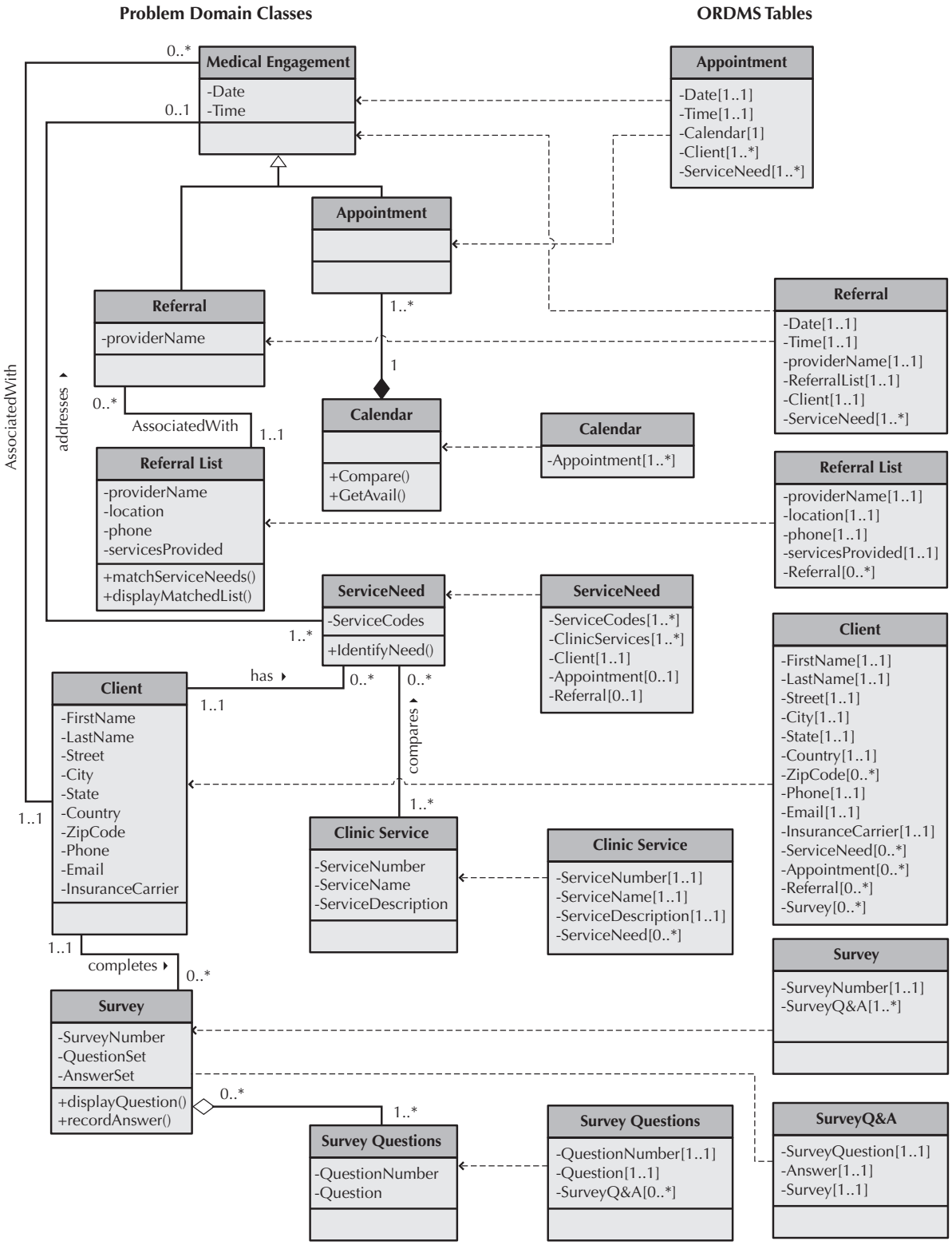


FIGURE 9-G Updated Problem Domain class to ORDBMS Table Mappings

discussion with Ruby and Ben, Jo agreed that since both the prescription fulfillment system and the Integrated Health Clinic Delivery system were strategic initiatives aimed at enhancing Patterson's advantage in the mobile market, leaving the ORDBMS-based mappings alone and using the new system as a mechanism to begin bridging the old systems from RDBMS to ORDBMS made sense. Furthermore, Jo agreed to have her database group deal with converting the old RDBMS tables to ORDBMS tables and to convert the old legacy systems to take advantage of this change. But in the interim, she suggested that her database group would first create a set of temporary ORDBMS tables that would map into the current RDBMS tables. This way, the mobile app development could be kept on track.

The third step was to determine whether there were any relevant database optimizations that should be performed on the ORDBMS tables. Given that an ORDBMS is being used for this system, the database did not require normalization. Moreover speed constraints related to joins used with RDBMS were not a concern. However, given the need for efficiency in a mobile context, the team decided to have Jo and her database group check into whether there were any additional efficiency issues related to using an ORDBMS that should be investigated.

The fourth and final step was to design the data access and manipulation (DAM) classes that act as translators between the object persistence and problem domain objects. A DAM class is created for each concrete class and ensures that changes in the object persistence format will only change the DAM object allowing the problem domain object to remain isolated from the change. Based on the problem domain to ORDBMS mappings (see Figure 9-G), the current version of the design requires nine DAM classes: Appointment-DAM, Referral-DAM, Calendar-DAM, Referral List-DAM, Service Need-DAM, Client-DAM, Clinic Service-DAM, Survey-DAM, and Survey Question-DAM. Each of these is shown in Figure 9-H. For clarity purposes, we use a class name view only (see Chapter 5).

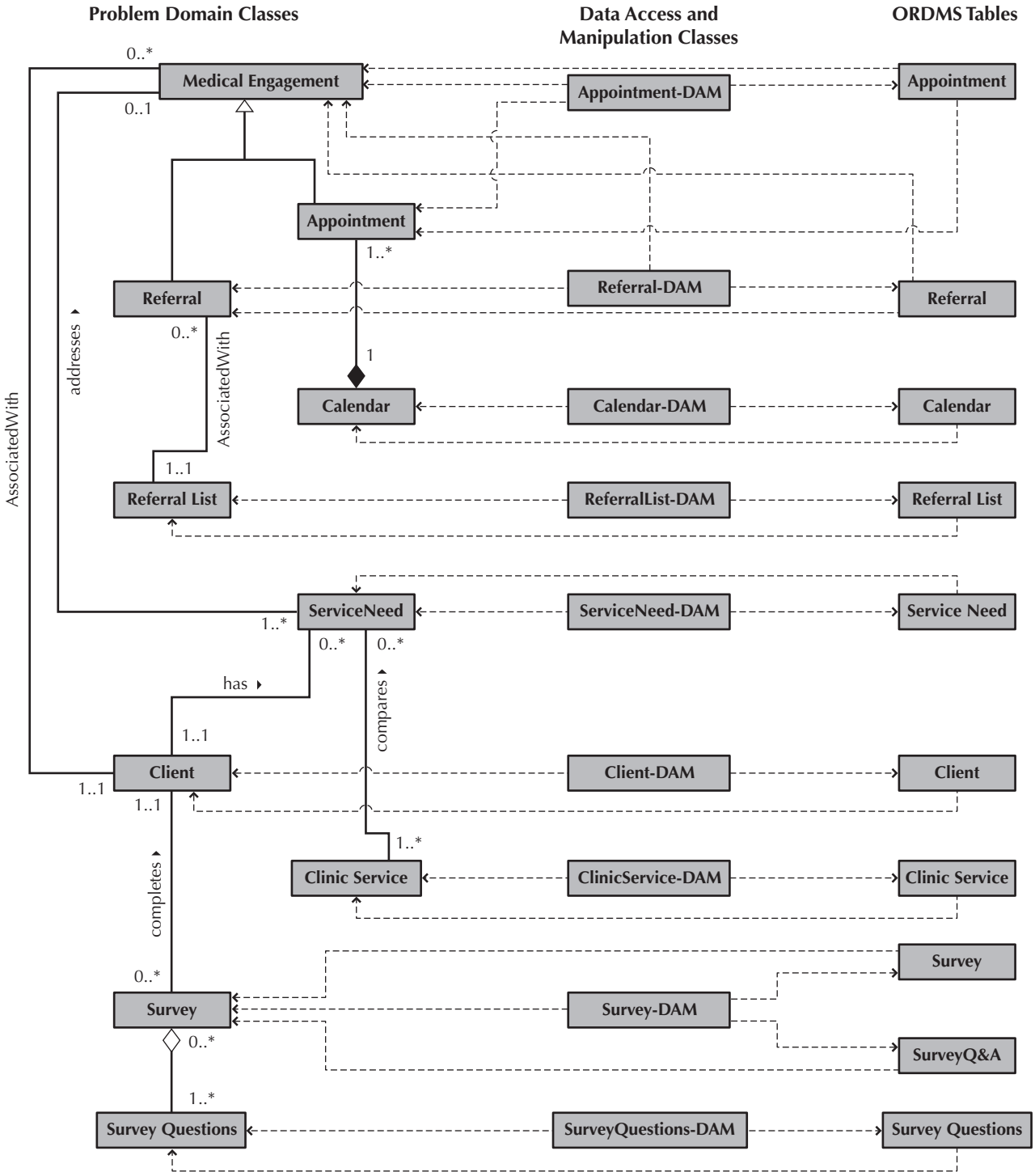


FIGURE 9-H Problem Domain class to ORDBMS Table Mappings via DAM classes

CHAPTER 10: PATTERSON SUPERSTORE CASE

From the users' perspective, the user interface is the system. For this reason, designing the user interface is particularly important. As described in the textbook, there are challenges related to designing the user interface for mobile devices. Some of these include:

- The size of the screens is small and not standard.
- Typing on the virtual and physical keyboards is difficult.
- Mobile devices are used everywhere.
- Mobile devices have unique I/O capabilities.

Furthermore, the look and feel to these devices tend to be totally based on the specific operating system being used. Consequently, the team decided to go for a simple, generic user interface that could be used across the different mobile platforms.

Since Kelly, the systems analyst, had already dealt with this issue with the mobile prescription fulfillment system, she was asked to head up the design for the HCI layer. The team, led by Kelly, began by studying the revised use-case descriptions (Figures 7-E, 7-G, and 7-I), activity diagrams (Figures 7-F, 7-H, 7-J), and sequence diagrams (Figures 9-C and 9-F). Based on this review, they created three use scenarios that covered the three use cases (see Figure 10-A).

The first use scenario, Client Selects Service, describes how the system will interact with the client when he or she selects a service that is provided by the clinic. This scenario combines aspects of the Schedule Appointment use case (Figures 7-E and 7-F) with the Make Appointment use case (Figures 7-G and 7-H). In this case, the client is able to identify the service that he or she needs from the list of available services offered.

The second use scenario, Client Fills out Survey and Patterson provides Service, also combines aspects of the Schedule Appointment use case with the Make Appointment use case. However, in this case, the client could not identify the service needed from the list of available services offered. So, the client must fill out the survey and have the system triage the client to determine which service is required. Once the appropriate service has been identified by the system, an appointment is made.

The third use scenario, Client Fills out Survey and Patterson provides Referral, combines aspects of the Schedule Appointment use case with the Make Referral use case (Figures 7-I and 7-J). Like the second scenario, the client could not identify an appropriate service and had to fill out the survey. However, in this case, since the clinic did not provide the appropriate service, the system could only identify a set of possible referrals.

Notice how the three scenarios overlap but do not replicate the exact same steps as the three use cases. If you go all the way back to the Chapter 4 installation of this case and review Figure 4-E, you should be able to identify three unique paths through that specific activity diagram that essentially combined the three use cases into one. The three unique paths are directly related to the three use scenarios documented here.

Navigation Structure Design

Using the three use scenarios, Kelly and her team began designing the structure of the navigation through the different user interface components that would be required to support the user in scheduling an appointment. To accomplish this, she created a windows navigation diagram (WND) that portrays all of the differing paths through the user interface (see Figure 10-B). Notice that at this point in time, Kelly has included a button for the Communicate Real Time and Tele-health Assessment use cases that will be developed with later versions of the system. This WND only documents the navigation through the user interface components

Use Scenario 1: Client Selects Service

1. Client requests Mobile Scheduling
2. System displays the available services
3. Client selects service
4. System displays wait times and availability list
5. Client selects preferred date and time for appointment
6. Client requests appointment at preferred date and time
7. If appointment is found
   System displays appointment information
   Else
   System repeats steps 5–7
8. Client confirms appointment
9. Systems displays confirmation message

Use Scenario 2: Client Fills out Survey and Patterson provides Service

1. Client requests Mobile Scheduling
2. System displays the available services
3. Client selects unknown
4. System displays Survey Question
5. Client selects answer
6. If Survey is not completed
   Repeat steps 4–6
7. System displays wait times and availability list
8. Client selects preferred date and time for appointment
9. Client requests appointment at preferred date and time
10. If appointment is found
    System displays appointment information
    Else
    System repeats steps 8–10
11. Client confirms appointment
12. Systems displays confirmation message

Use Scenario 3: Client Fills out Survey and Patterson provides Referral

1. Client requests Mobile Scheduling
2. System displays the available services
3. Client selects unknown
4. System displays Survey Question
5. Client selects answer
6. If Survey not completed
   Repeat steps 4–6
7. System displays a list of referrals
8. Client selects referral

FIGURE 10-A Use Scenarios

that support the Mobile Scheduling aspect of the system. When the Communicate Real Time and Tele-health Assessment use cases are developed, the WND will need to be modified to document the navigation through the additional user interface components required to support those use cases.

After developing the WND, Kelly had the team walkthrough the WND by using the three use scenarios to ensure that the WND captured all of the user paths. The first use scenario is supported by following the path that begins with pressing the Mobile Scheduling Button at the top center of the diagram. Next, the client selected a service from the dropdown list and presses the Submit button to move onto the Wait Times and Availability List window where the wait times are displayed in a scrollable list and the client can enter the preferred date and time for the appointment being scheduled. By pressing the Find Appt button, the client requests the system to create an appointment on his or her preferred date and at the preferred time. Notice the XOR arc connecting the two Press Find Appt button paths. This documents the fact that only one of the paths will be executable at a time. In this case, if the system cannot find an appropriate date and time, the system will simply ask the client to try again by changing his or her preferred date and time. This will go on until an acceptable date and time are found to be available. At that time, the system will traverse the other path and ask the client to confirm the appointment by pressing the Confirm button, which will cause the system to send a confirmation message back to the client. Finally, the Client presses the Done button to return to the Main Menu of the system.

The second use scenario is supported by following the path that begins with pressing the Mobile Scheduling Button at the top center of the diagram. Next, the client selected unknown from the dropdown list and presses the Submit button to move onto the Survey interface component. The client will answer each question by selecting an answer from the dropdown list associated with each question. The client will either move on to the next question, by pressing the Next button, or he or she will move on to the Wait Times and Availability List window, by pressing the Done button. The Done button will only become available once the entire set of questions have been answered. Notice the XOR arc again associated with the Done button. In this case, the path that is executed is based on the system determining that the clinic did indeed provide the relevant service. This is why the use scenario then follows the same path that the first use scenario followed in creating an appointment.

The third use scenario is supported by following the path that begins with pressing the Mobile Scheduling Button at the top center of the diagram. Next, the client selected unknown from the dropdown list and presses the Submit button to move onto the Survey interface component. The client will answer each question by selecting an answer from the dropdown list associated with each question. The client will either move on to the next question, by pressing the Next button, or he or she will move on to the Referral List MsgBox. In this case, the system determined that Patterson could not provide the relevant services and instead provides a set of possible referrals. The client will select a relevant service from the dropdown list and press the Done button. At which time the system will send the referral information to the client and return them to the Main Menu of the system.

Once the team verified that the WND did indeed represent the structure of the designed navigation through the user interface, the team began to design the initial user interface. Using the established interface standards that were developed for the mobile description fulfillment system, the team set to work.

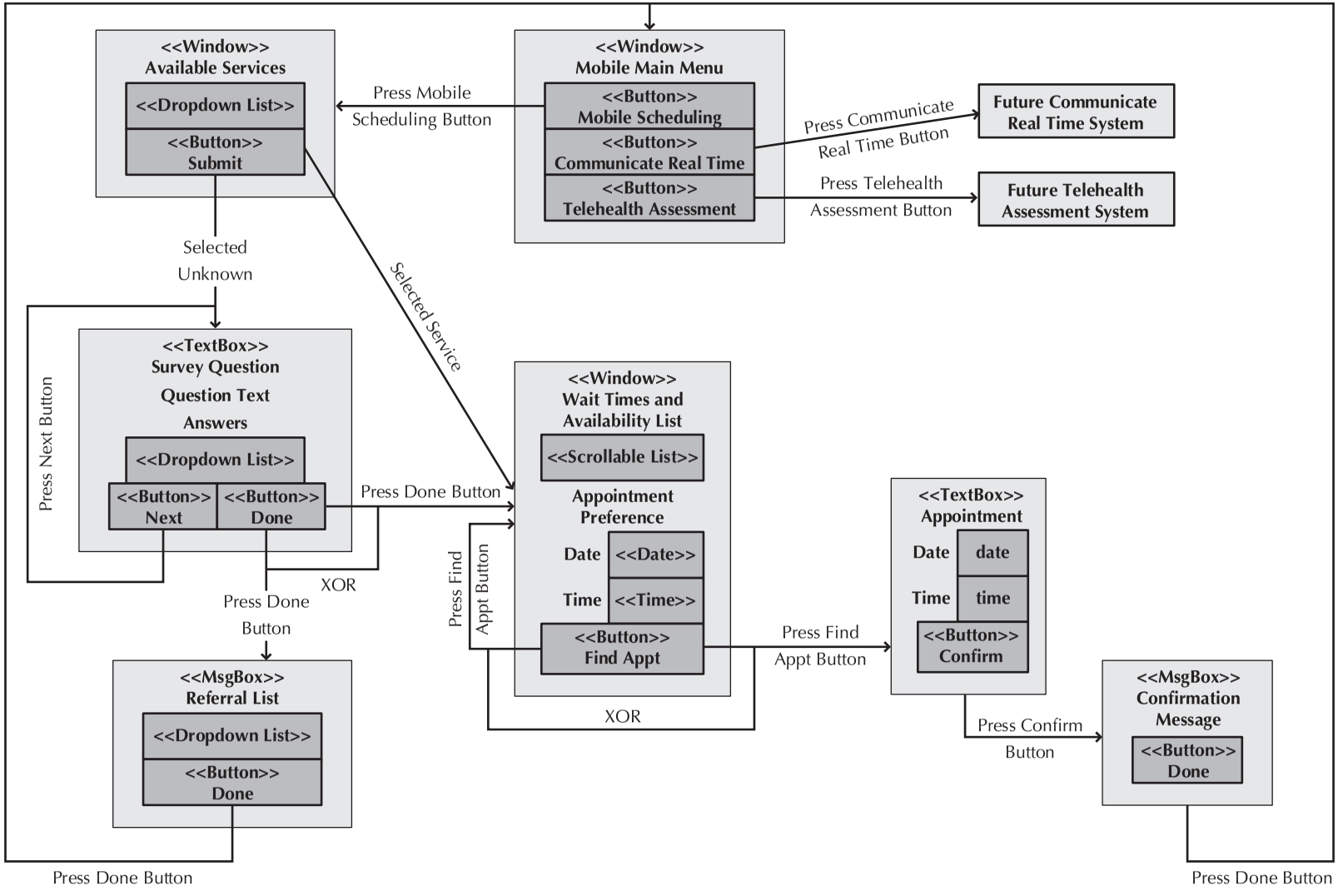


FIGURE 10-B WND for Integrated Health Clinic Delivery System (Phase 1)

Interface Design Prototyping

First, the team mocked up a set of windows layout diagrams as a way to design the look and feel of the mobile interface. Figure 10-C portrays the different diagrams using the same general layout as Figure 10-B (the WND). By laying out both diagrams in the same way, it was very easy for Kelly to be able to "see" the navigation that would take place between the different user interfaces.

The Main Menu diagram is located at the top center of Figure 10-C. It comprises three buttons, one for each major function that the system will gradually support. The Available Services interface shows an example list of services. Notice that Service 3 is selected (the dashed oval around it). When the user presses the Submit button, the Wait Times and Available interface will be displayed (located in the middle of the figure). In this case, notice the Try Again message that appears at the bottom of the interface. This message will only appear if the Appointment Preference Date and Time is not available; otherwise, when the Client presses the Find Appt button, the Appointment interface would be displayed. Comparing the Survey Question interface to the WND, it is very easy to see from where each interface component came. Also, notice that the Answer 2 was selected (again portrayed by the dashed oval around it).

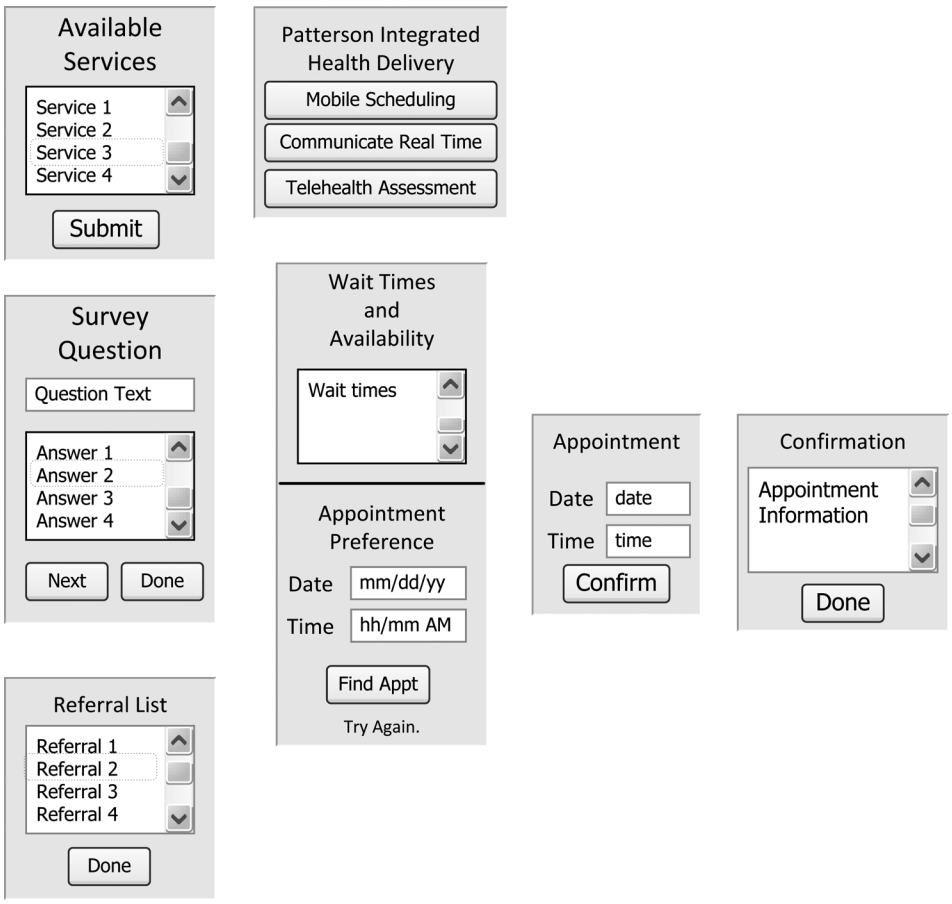


FIGURE 10-C Windows Layout Diagram for Integrated Health Clinic Delivery System (Phase 1)

After developing the windows layout diagrams, Kelly had the team walkthrough the diagrams by using the three use scenarios to ensure that the look and feel of the interfaces would support the scenarios and the three use cases on which the scenarios were based. Once the team verified that the diagrams did provide the layout for an effective user interface, the team began the process of documenting the navigation design.

Navigation Design Documentation

The next step was to document the navigation design by transitioning the essential use cases to real use cases. The real use cases for the Schedule Appointment, Make Appointment, and Make Referral are shown in Figure 10-D. Notice the changes between the essential and real use case descriptions. First, the use case type was changed. Second, the texts of the Normal Flow of

Use Case Name: Schedule Appointment	**ID:** 1	**Importance Level:** High
Primary Actor: Client	**Use Case Type:** Detail, Real	
Stakeholders and Interests: Client wants to schedule an appointment Existing Health Clinic System provides information about clinic services		
Brief Description: This use case describes how an appointment is scheduled electronically		
Trigger: Client requests to be seen **Type:** External		
Relationships: **Association:** Client, Existing Health Clinic System **Include:** **Extend:** Make Appointment, Make Referral **Generalization:**		
Normal Flow of Events: 1. Client presses Mobile Scheduling button 2. The system displays the available service offerings list 3. Client selects an existing service and presses Submit button that Executes the Make Appointment use case		
SubFlows: S-1: Determine Suitability 1. System displays survey question 2. Client selects question answer 3. Client presses Done button 4. System execute pre-existing Triage Needs use case 5. System determines whether service need is within scope of clinic's services 6. Execute Make Appointment use case		
Alternate/Exceptional Flow: 3a. Client selects Unknown as service option and presses Submit button that Executes the S-1: Determine SuitabilitySubFlow S-1, 3a. Client presses Next Button S-1, 3b. If Done button is not active, client iterates steps 1 through 3 until it becomes active 6a. If services required are outside of clinic's capabilities, Execute Make Referral use case		

FIGURE 10-D Real Use Cases

Use Case Name: Make Appointment	**ID:** 1-1	**Importance Level:** High
Primary Actor: Client	**Use Case Type:** Detail, Real	

Stakeholders and Interests: Client wants to schedule an appointment
Administrative Staff provides wait time information, updates calendar, and sends confirmation
Existing Health Clinic System Service supplies information about appointment availability

Brief Description: This use case describes how the client chooses an appointment

Trigger: Client wishing to schedule a clinic appointment has service needs that match clinic capability
Type: External

Relationships:
Association: Client, Admin Staff, Existing Health Clinic System
Include:
Extend:
Generalization:

Normal Flow of Events:
1. System displays current wait time appointment availability to client
2. Client enters appointment preference date/time
3. Client presses Find Appt button
4. Matching appointment availability displayed
5. Client chooses available and desired appointment by pressing Confirm button
6. System retrieves client information
7. System updates the Calendar
8. Appointment confirmation message is sent to client
9. Client presses Done button to return to Main Menu

SubFlows:

Alternate/Exceptional Flow:
4a. If no match occurs, client iterates steps 2 through 4 until satisfactory time is found

FIGURE 10-D (*Continued*)

Use Case Name: Make Referral	**ID:** 1-2	**Importance Level:** High
Primary Actor: Client	**Use Case Type:** Detail, Real	
Stakeholders and Interests: Client needs a referral Existing Health Clinic System Service provides information about referrals		
Brief Description: This use case describes how referrals how handled		
Trigger: Client need cannot be met by clinic **Type:** External		
Relationships: **Association:** Client, Existing Health Clinic System **Include:** **Extend:** **Generalization:**		
Normal Flow of Events: 1. System displays list of Referrals 2. Client selects referral from list 3. Client presses Done button to complete transaction and return to Main Menu 4. System retrieves client information 5. System creates referral and sends to client		
SubFlows:		
Alternate/Exceptional Flow:		

FIGURE 10-D (*Continued*)

Events, SubFlows, and Alternative/Exceptional Flows have changed from a high-level description to a very specific set of directions as to how the navigation through the various interfaces (Figure 10-C) will occur. For example, the first Normal Flow of Events step for the Schedule Appointment essential use case stated:

> Client requests to be seen by the clinic.

But, when it is converted to a real use case, it becomes:

> Client presses Mobile Scheduling button.

Interface Evaluation

The final step was to evaluate the user interface design. To accomplish this, Kelly had each team member independently "execute" the three use scenarios and the three use cases by performing a walkthrough using the use scenarios (Figure 10-A), the windows layout diagrams (Figure 10-C), and the real use case descriptions (Figure 10-D). Upon completing this, Kelly and the team felt as if they had designed a very usable mobile user interface for the Mobile Scheduling phase of the system.

CHAPTER 11: PATTERSON SUPERSTORE CASE

The physical architecture layer includes the hardware, software, and network environment of the system. Sam Wilson, the infrastructure analyst, was tasked with the design of the architecture layer. Sam's job as infrastructure analyst is to ensure that the system conforms to the infrastructure standards at Patterson and that the Patterson infrastructure can support the new system. Since the prescription order notification and auto refill system, already in place, is available for both mobile and desktop users, the infrastructure needed for Phase 1 of the Integrated Health Clinic Delivery System was already in place. Figure 11-A shows how the hardware will be deployed to support the systems, while Figure 11-B depicts how the software layers will be deployed onto the different pieces of hardware.

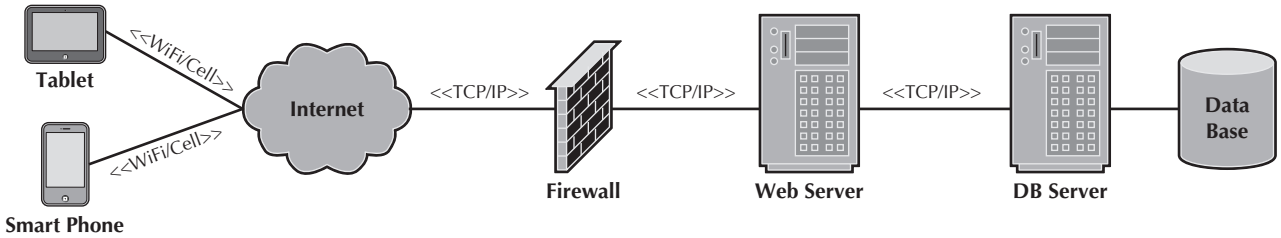


FIGURE 11-A Hardware-Oriented Deployment Diagram for Phase I Mobile Scheduling

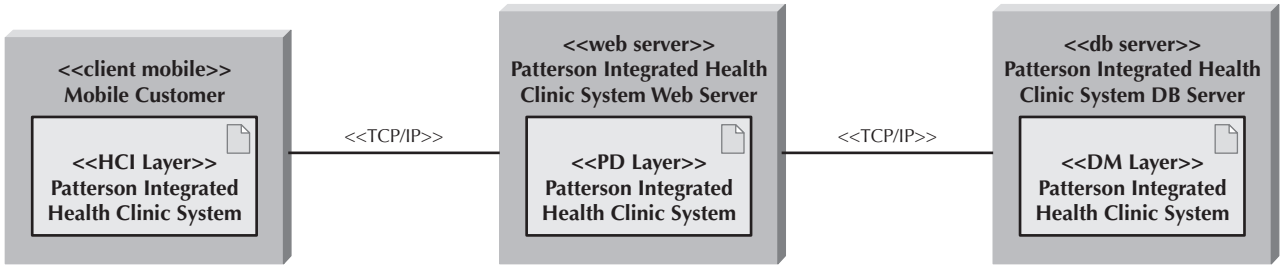


FIGURE 11-B Software-Oriented Deployment Diagram for Phase I Mobile Scheduling

CHAPTER 12: PATTERSON SUPERSTORE CASE

Programming, which takes place during construction, is typically the most expensive and time-consuming aspect of systems development. While the analysts were not involved with programming activities, they were instead busy developing user documentation and designing test plans for the Mobile Scheduling phase of the Integrated Health Clinic Delivery System.

Documentation

The team had been documenting their activities throughout the SDLC. System documentation helps the technical staff understand the system and enables them to build and maintain the system. Now it was time to create user documentation designed to help the user operate the system.

User documentation includes user manuals, training manuals, and online help systems. Because clients will access the system via mobile devices and desktop computers, the user documentation needs to be available from these devices. For this reason, online documentation was the primary documentation created. However, it should be noted that brief, printed instructional fliers would be placed at the pilot clinic and at each clinic just prior to the site going live with the Mobile Scheduling System. These printed materials will serve as both an announcement of the mobile system and instructional information on how to access mobile scheduling.

Online reference documents were created to instruct the user on how to perform specific tasks, such as accessing, completing, and submitting the service need survey; entering appointment preferences; and choosing from available appointment dates and times. These reference documents were designed to be available both from the help button and through context sensitive mechanisms. One challenge for the group was designing context-sensitive help on touch devices. This was a challenge since hovering or mouse-overs do not exist with touch screens. After researching this issue, the team found varying solutions based on screen size but were unable to find a single solution that would work both on small and larger screen devices.

In addition, the team created an online tutorial that users could access from the home page for the mobile scheduling system. While most users would probably not utilize the tutorial, the team wanted to provide a tutorial for those novice technical user who desired and would benefit from a step-by-step tutorial.

The last user documentation that the team created was a procedural manual for use within the clinic to guide employee users through the changed business procedures brought by the mobile scheduling system.

Testing

Throughout the SDLC, the team had tested the developing system, including the functional, structural, and behavioral models. They had also checked for consistency across the models. Further testing continued with the development of unit, integration, system, and acceptance tests.

Unit testing was done through black-box texting of the encapsulated classes, which had been tested throughout the analysis and design phases. Now invariants on the CRC cards, class diagrams, and the pre- and post-conditions in contracts were tested.

Integration test plans were developed to assure that the system displayed correctly on all possible mobile/web interfaces. Integration testing also assessed the proper interaction of classes as well as the correct exchange of data within the system.

The goal of system testing is to ensure that all requirements are met for all components of the system. Systems testing examines the ability of the system to meet both functional and nonfunctional requirements. Performance, security, and usability tests were developed and performed.

Particular attention was placed on performance testing of the mobile/web interface components. The goal was to determine how many clients could access the system without incurring

delays in response time. Security of all aspects of the system was tested to guarantee that HIPPA privacy requirements were met. Lastly, usability testing was conducted with the clients who participated in the initial JAD sessions during the analysis phase. Usability testing assessed how well the user interface supports the use cases. Usability tested whether the functional requirements were met by the system as well as the ease of use of the system.

Acceptance testing was done in two parts. First, alpha testing was conducted using test data during staff training sessions. A series of tests and activities were designed to both train the clinic staff and test all aspects of the system with data. After alpha testing was successfully completed, beta testing was conducted at the busiest clinic, which had been chosen as a pilot site. Because the clients who had participated in the initial JAD were from this clinic, these clients were invited to be the first to use the system with live data. After successful beta testing with this limited number of users, the pilot site would go live for all users.

CHAPTER 13: PATTERSON SUPERSTORE CASE

In this final segment of the Patterson Superstore Case, we see how the new system is put into production for use by the Clinic mobile users. Ruby and Max oversaw most of the activities, including conversions preparation, employee and client training, and both the system and project team review.

Conversion

During planning, it was determined that the Integrated Health Clinic Delivery System would be developed in three versions. Mobile scheduling, the first version, was to be deployed as a pilot at the busiest clinic. After running successfully at the pilot site for three months, the system will go live at other sites. While the location conversion strategy was pilot, the conversion style is parallel; both the new and existing systems run concurrently. It should be noted that mobile scheduling was developed as an enhancement rather than a replacement for the on-site scheduling procedures currently in use.

Change Management

Change management is not a significant problem with this system. Because appointments can still be made by phone or on a walk-in basis, only those clients who wish to utilize the new system will need to do so. These clients will be the more technical users who currently use the prescription system and who have requested mobile services for appointment scheduling. The more reluctant adopters can wait until they are ready to use the new system or can continue to use the on-site scheduling procedures.

Training

For those users ready to adopt mobile scheduling, the tutorials and reference materials, discussed in the last chapter, are available online. At the pilot site, the small number of clients who participated in the initial JAD sessions had also been involved with usability testing and had developed some familiarity with the system through that experience. These clients were invited to be the first to use the system with live data.

Post-Implementation Activities

After the implementation, support was transitioned to the operations group; one additional mobile support expert was added to the group to work in conjunction with the operations staff already supporting the mobile prescription fulfillment system.

Also, two reviews were conducted. The project team review focused on the performance of the development team. The review uncovered lessons learned that will be important as development of the system moves on to Version 2 and Version 3. For example, the team realized that they would have benefited if they had included Ben and Jo earlier in the development. Their knowledge of both data analytics and the underlying database would have been very useful earlier in the process. Furthermore, given the focus of Version 2 and Version 3, their involvement will be crucial if these versions are to be successful. The systems review was conducted after two months of operation. While the mobile scheduling was running relatively smoothly with no major problems, Max was somewhat disappointed that usage was not higher at the Pilot site. Ruby suggested that, given the smooth rollout at the pilot site, more promotional activities could be utilized to enhance visibility and usage at future sites.