**Use Cases for a Parking System**

ICT 4305: Object Oriented Methods and Programming

Michelle Agustin

University of Denver College of Professional Studies

September 21, 2025

Faculty: Nirav Shah, M.S , MBA

Director: Cathie Wilson, M.S.

Dean: Michael J. McGuire, MLS

**PART 3: REFLECTION**

Working on the use case diagram for the University Parking System was both a challenging and rewarding experience. One of the easier aspects was identifying the main actors. Since the project description already emphasized roles such as customers, the parking office, enforcement officers, and external systems like payment gateways, these stood out quickly as key participants. It also helped that I could draw from real-world experiences with parking systems, thinking about how, I as a customer, register, pay, and scan a permit at the gate gave me a strong starting point.

The difficult part was translating these actors intro precise use cases while ensuring the that the diagram was not overloaded with details. It was tempting to map every possible interaction, but I had to remember that use case diagrams are meant to highlight broader system behaviors, not every fine-grained step. Another challenge was understanding and applying the “include” and “extend” relationships correctly. At first, I thought they were interchangeable, but after reviewing UML guidelines, I learned that *includes* represents mandatory functionality like “Exit Lot” must include “Calculate Parking Fee”, while *extends* represent optional or conditional behavior like applying a discount extends the fee calculation if certain criteria are met. That distinction helped the diagram make more sense and improved its accuracy.

Several things helped me along the way. Reviewing UML references, such as Fowler’s UML Distilled (2004), clarified both notation and best practices for simplicity. Class discussions and visual examples also reinforced how to balance readability with completeness. Using diagramming tools allowed me to experiment with layouts and better visualize actor-use case connections before finalizing.

I do wish I had reviewed UML notation more thoroughly before starting. For example, I initially mixed up whether an arrow in “include” should point from the base to the included case or the other way around. Sorting this out early would have saved some rework.

For implementation decisions, I prioritized clarity over complexity. I focused on core use cases such as registering, entering/exiting lots, calculating fees, and processing payments since these are essential to the system’s functionality. Administrative tasks such as configuring lot pricing and generating reports were included but kept secondary to avoid crowding. I also deliberately positioned “Apply Discount” as an extend case rather than an independent one, because it only occurs when fees are calculated, not as a standalone interaction. This decision reflects the system’s conditional logic while keeping the diagram structured.

Overall, the assignment reinforced my understanding of how use case diagrams provide a high-level, visual roadmap of system interactions. While there were challenges, I now feel more confident applying UML principles to object-oriented system design.