DESIGNING PILL DISPENSERS FOR MILD COGNITIVE IMPAIRMENT – AN APPLICATION OF USE CASE UML

Jessie Liu[1]

[1]Georgia Institute of Technology, Woodruff School of Mechanical Engineering

ABSTRACT

Unified modeling language (UML) is a standardized visual syntax used to model systems, first created for software design. Here, the application of a use case UML diagram towards design processes is explored instead. A prior project regarding collaborative research and design for people living with mild cognitive impairment is used as a basis to demonstrate the potential application of use case UML in a design process. As a part of this project, interviews were conducted in collaborative workshops with someone with MCI and their care partner. A design focus is then narrowed down from these collaborative interviews. With the information provided by the workshop participants, detailed use cases are then formulated and written into use case diagrams and their accompanying fully-dressed use case tables to demonstrate the applications and advantages of using use case UML in a design process. Use case UML allows designers to fully represent every scenario of their system, catching potential fringe cases and failure modes to create a truly robust design.

Keywords: mild cognitive impairment (MCI), use case unified modeling language

1. INTRODUCTION

Mild cognitive impairment (MCI) affects anywhere from 3% - 19% of people aged 65 and above. This impairment comes with difficulties associated with recent short-term memory, executive function, and potentially daily functional tasks. While not all cases of mild cognitive impairment lead to dementia, there is still significant risk in its development [1].

As a part of a collaboration between an Industrial Design Special Topics course at the Georgia Institute of Technology and Emory University's Brain Health Center and Georgia Institute of Technology's Cognitive Empowerment Program (CEP), two interactive workshops were held to interview a member of the CEP program with MCI and their care partner – someone they lived with without MCI that assisted in their everyday routine. Through these interviews, the design topic was narrowed down to an automatic pill dispenser. While a design and prototype was created in the class, this paper explores the application of use case UML using this project.

1.1 Project Background

The workshops and data collected were completed as a part of ID 4833: Special Topics Collaborative at the Georgia Institute of Technology. This class focused on collaborative design processes. Collaborative design involves the demographic the product is being designed for in order to obtain first-hand knowledge of their needs and wishes. In particular, the class focused on interactive visual interview methods, such as games and activities. By providing visual cues and interactive items during interviews, the interviewee's memory can be stimulated to provide more information otherwise unavailable.

Another aspect of this class was co-design: looping in the customer as a part of the design and prototyping process to ensure the finished product better matches any and all customer requirements. By employing these two design tactics, a wholistic understanding of the customer needs is achieved, allowing for a fully informed and detailed application of use case UML and consequently better design.

1.2 Unified Modeling Language (UML)

Unified Modeling Language (UML) is a modeling language developed in 1995 to help standardize the way people represent models in software systems. Adopted as an official ISO standard in 2005 (ISO/IEC 19501:2005), UML is a useful tool to help model systems in design spaces as well. It includes standardized graphic notation to represent systems of a model, its views, systems, and subsystems.

UML representations allow for both a static and dynamic views of the system. The static view, also known as the structural view, shows the basic relationship between users, operations, objects, etc. and the way they interact with each other. The dynamic view, on the other hand, allows for representations of potential changes in the state of objects, the behavior of the system, and other more complex interactions between actors.

There are numerous kinds of diagrams encompassed within UML, the use case diagram being one of them. Other diagrams, shown in Figure 1 [2], can be broken down into two types: behavioral and structural diagrams. Structural diagrams, like object diagrams, focus more on grouping similar classes, objects, components, etc. together to model the layout of a system. Behavior diagrams, like use case diagrams, focus more on the interactions between these actors, activities, use cases, etc.

While there are many interesting and useful applications of UML for different scenarios, this paper focuses on applying use case UML to the design process [3].

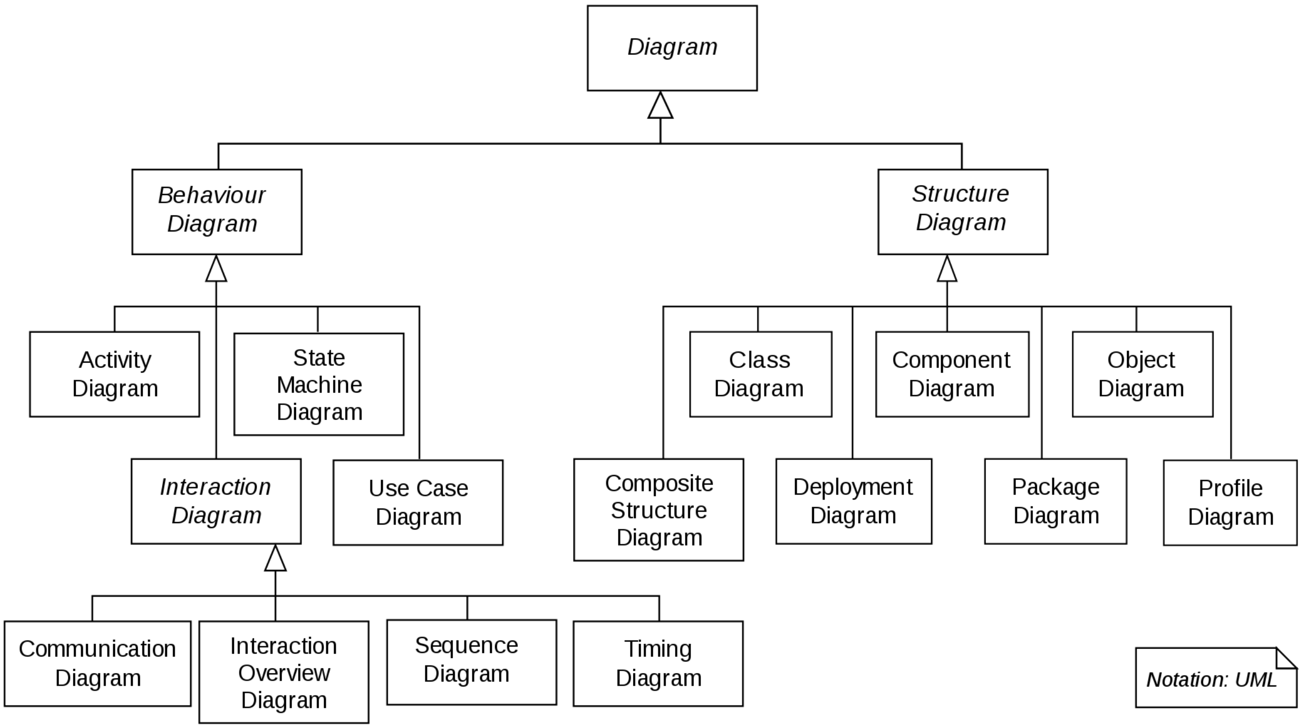

FIGURE 1: DIFFERENT DIAGRAM REPRESENTATIONS IN UNIFIED MODELING LANGUAGE [2]

1.3 Use Case UML

Use case diagrams are a way to model how users interact with a system. By representing systems, top-level functions within that system, sub-functions for different use cases, and different users, designers can get a thorough understanding of how their product or system will interface with its users. This also allows them to thoroughly account for any failure modes and therefore create a robust design.

Use case diagrams have a standardized syntax established by UML. Actors, or users, are represented by stick figures. They're labeled and placed outside the system. The system that is being designed is represented by a box. Inside this box, use cases, written inside bubbles, are placed. These represent the various functions or possibilities the designed system has when interacting with the actors. Using dotted lines and arrows, all labeled, these use cases can be linked together to isolate and break down functionality into different scenarios [3]. Figure 2 shows an example use case diagram modeled for an online shopping system. Note the various actors and how their use cases interact with each other via include and extend mechanisms.

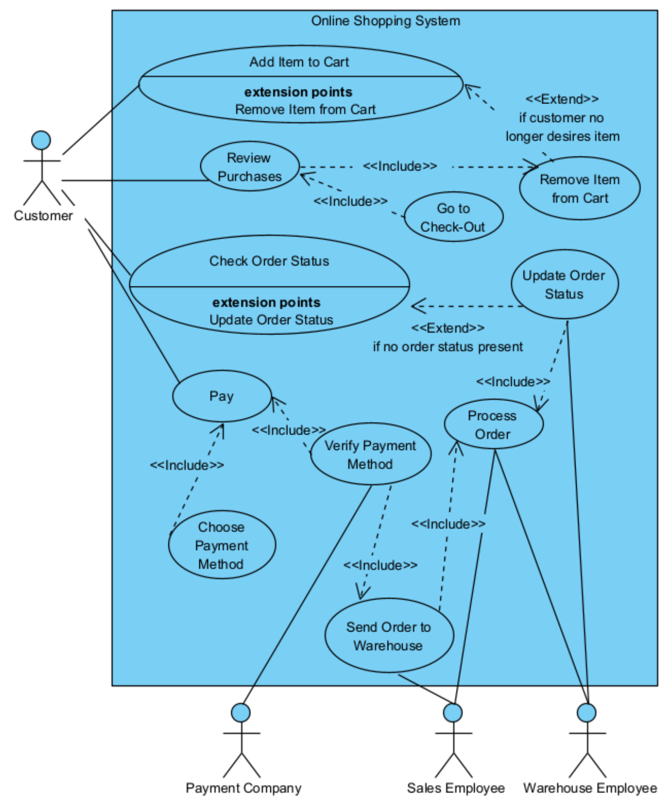

FIGURE 2: EXAMPLE USE CASE DIAGRAM FOR AN ONLINE SHOPPING SYSTEM [4]

TABLE 1: EXAMPLE OF A FULLY-DRESSED USE-CASE TABLE FOR AN ONLINE SHOPPING SYSTEM [4]

System	Online Shopping System
Name	Add Item to Cart
Actors	Prime: Customer
Other Stakeholders	ParaMedia Company
Description	Customer adds an item they wish to purchase to their shopping cart
Main Success Scenario	1. Customer finds item they wish to purchase within system 2. Customer selects type and quantity of product 3. Customer selects "Add to Cart" to add desired item to cart 4. Item(s) are added to the Shopping Cart
Alternative 2a	In Step 2, if the user does not specify a type or quantity, the first listed type at a quantity of 1 is added to the cart.
Alternative 3a	In Step 3, if the user adds an undesired item, they can proceed to their cart/check-out to remove items.
Precondition	There are items in the system in stock to be purchased.
Post Condition	There are item(s) in the customer's shopping cart.
Assumptions	User has a device with access to the internet to access the online shopping system.
Special Requirements	N/A

In addition to the use case diagram itself, a table for each use case is created to describe the functions to a fully-dressed level [5]. These fully-dressed use case tables allow systems to be represented in full detail, with every potential failure mode and scenario accounted for. An example table is included in Table 1. As shown in this example table, the fully-dressed use case table allows for further details such as what the use case entails, what must happen before and after the use case occurs, as well as assumptions. A fully-dressed table is created for every top-level use case in the system to fully capture the scope of the system.

2. MOTIVATIONS

Illustrating the applications of use case UML can help teach designers a new modeling skill to facilitate their design process. In particular, coupling this application with this prior project done with people with MCI and CEP creates an interesting display of how different design methods can supplement and build off of one another.

The Industrial Design Special Topics course was designed to practice collaborative design methods, especially focused on finding customer needs. This directly feeds into applications of use case UML to represent the culmination of those design processes.

3. MATERIALS AND METHODS

Two workshops were held with CEP members to obtain data. Each group was paired with one member for the duration of the project, so data included in this paper was collected from two individuals: the CEP member with MCI (whom we will call A) and their care partner (whom we will call B).

3.1 Workshop 1

The first workshop was focused on obtaining data on the participants' daily routine and the difficulty levels of those routine tasks.

The first activity focused on various rooms in the house. Activities involving memory can be used to help people with MCI exercise their short-term memory. Thus, matching tiles were provided with various rooms on them for A. They were placed face down, and A was directed to flip them over in pairs of two in order to find two matching tiles. B helped during this activity with both flipping the cards and offering hints for tile locations. When a match was made, questions were asked to probe for difficulties surrounding each area of their home. This activity resulted in more qualitative data that informed the activities done in Workshop 2.

After finishing this activity, the tiles were repurposed to create a house layout to gain a better understanding of the environment in which they encountered difficulties. This also allowed additional data for areas that aren't rooms to be collected, such as stairs and entryways. An image of the resulting layout can be seen in Figure 3.

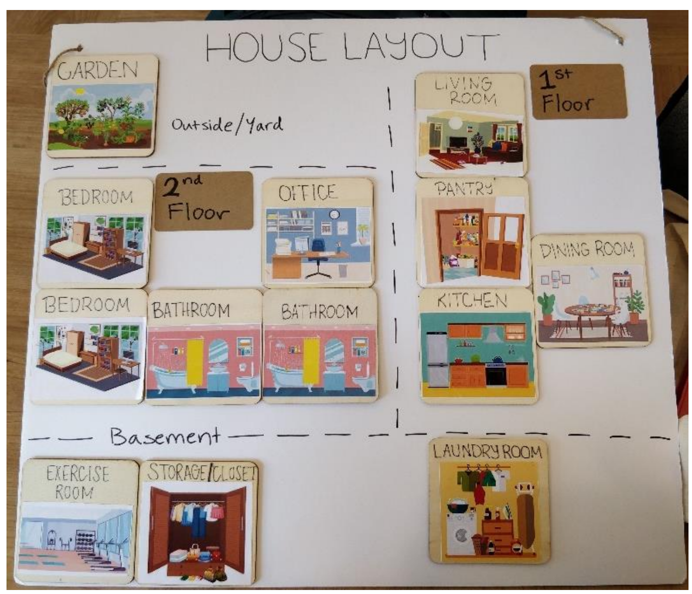

FIGURE 3: HOUSE LAYOUT AND TILES FROM ACTIVITY 1 OF WORKSHOP 1

Activity two involved ranking various household tasks on a difficulty scale. This helped gather data on difficult tasks rather than rooms in order to narrow down a design scope. The participants were asked to walk through their routine, placing tasks as they described them on the difficulty scale shown in Figure 4.

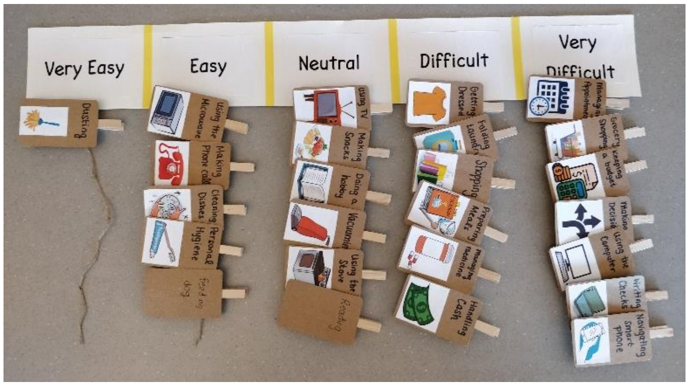

FIGURE 4: ACTIVITY 2 OF WORKSHOP 1

3.2 Workshop 2

Workshop two was formulated based on the results from workshop one. This workshop was focused more on their morning routine with the goal to truly narrow down which area the participants wished to improve and co-design some features and characteristics for a prototype. The activities were also refined to be more friendly to A and their abilities after getting to know them and their struggles with MCI during the first workshop. The shape of the tiles was changed to be easier to pick up for A, and the tiles themselves were labeled the activities they had described in their routine during workshop one.

First, a journey map was made of the participants' morning routine. They placed a task, marked whether A, B, or both of them together completed that task, and how that task made A feel (good, neutral, or bad). These emotions and who completed the task were marked by stickers on the tiles.

Next, the tiles were placed on a coordinate system, displayed in Figure 5. By ranking how these activities made A feel and their importance to them, the scope of the design project is further narrowed down. The result of this activity is shown in Figure 6.

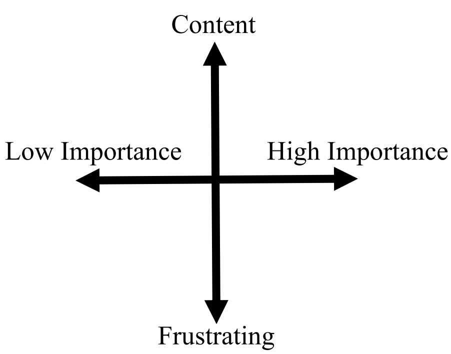

FIGURE 5: WORKSHOP 2 ACTIVITY 2 COORDINATE SYSTEM

FIGURE 6: WORKSHOP 2 ACTIVITY 2 RESULTS

The next activity, shown in Figure 7, was to ask A which activities they desired to do on their own. This narrows down the scope of which areas are higher priority in giving A independence and empowerment, the goal of CEP. At the end of this activity, there was enough data to narrow the design topic. The task with high importance, frustration, and desire for independence was the key task to design for.

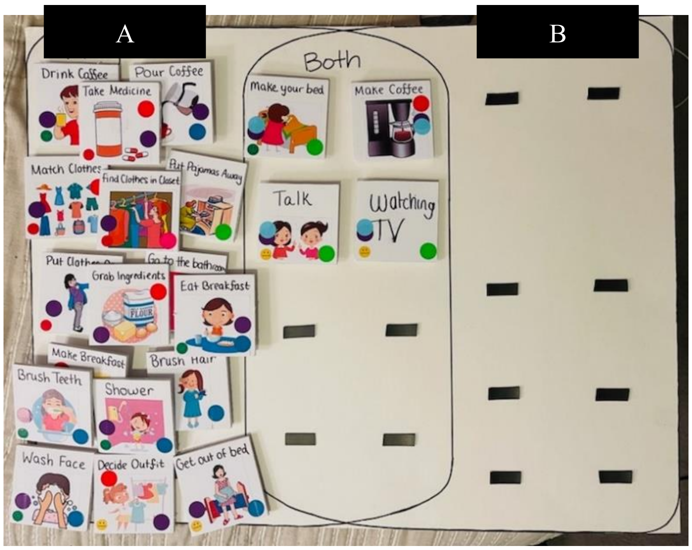

FIGURE 7: WORKSHOP 2 ACTIVITY 3 VENN-DIAGRAM

Finally, the last activity was co-design. Together with the participants, various potential solutions were discussed. With this information they provided, desired use cases can be formulated for use case UML, facilitating design. This is discussed further later in the paper.

4. WORKSHOP RESULTS

The key findings from workshop one, illustrated Table 2 and Figure 8, pointed towards two key difficulty categories: their wardrobe/getting dressed in the morning and information tracking/processing challenges. MCI creates difficulties making decisions and remembering where clothes lie in the closet, leading to difficulties getting dressed independently in the morning for A. It also creates difficulties in short term memory, making tasks such as taking medicine, managing appointments, and reading difficult.

Another problem area discovered through workshop one was the bathroom. Due to old-age mobility issues, the bathroom was a dangerous place for A to be. Due to the goal of the workshops being to focus on empowerment for MCI, as well as preference input from the participants, the decision was made to focus on the closet and getting ready for the second workshop.

TABLE 2: WORKSHOP 1 ACTIVITY 2 RESULTS

Difficulty Level	Tasks
Very Easy	Dusting
Easy	Using Microwave Making Phone Calls Personal Hygiene Feeding the Dog Cleaning Dishes
Neutral	Using Television Using Stove Doing a Hobby Making Snacks Vacuuming Reading
Difficult	Managing Medicine Preparing Meals Getting Dressed Folding Laundry Handling Cash Shopping
Very Difficult	Managing appointments Navigating Smart Phone Using Computer Grocery Shopping Keeping a Budget Making Decisions Writing Checks

However, during the second workshop, factoring in A's ratings on importance and desire for independence, the data pointed towards focusing on taking medicine as the main design focus. Due to the memory problems surrounding MCI, A has difficulty remembering to take medication and which medications to take. When put in a pill box, A has difficulty knowing the correct day and time (AM/PM) to take the medication from. Thus, currently, B sorts A's medication into a pill box weekly and transfers each serving's contents into a medicine cup left on the bathroom counter for A.

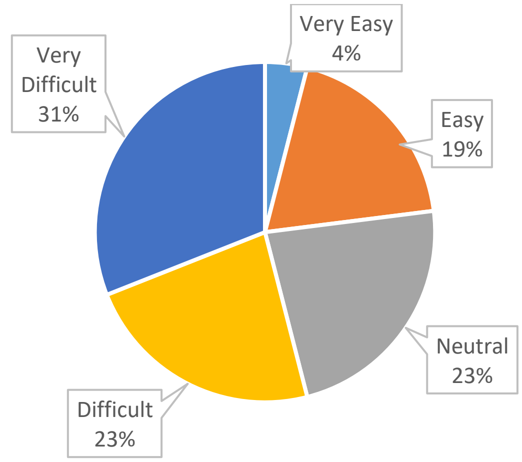

FIGURE 8: DIFFICULTY BREAKDOWN OF VARIOUS TASKS FROM WORKSHOP 1 ACTIVITY 2

Discussion during the co-design activity revealed that A liked having their medicine given to them by B using small medicine cups. They also said that weekly set-up of the medicine could be entrusted to B, and that by simply giving A the ability to take medicine daily on their own would be a big step towards independence and empowerment.

Thus, the design focus was reached to create a pill box that helps remind and automate some of the medicine taking process for A. To facilitate this, alarm clock functionality was also added in. A picture of the resulting prototype is shown in Figure 9 to help facilitate understanding of the system for further discussion of the application of use case UML.

FIGURE 9: PROTOTYPE PILL DISPENSER

The system dispenses medicine by pushing out a medicine cup holding the correct dosage at the correct times. The user can then take the medicine cup, consume the medication, and store the cup back in the system.

5. APPLYING USE CASE UML

5.1 Creating the Use Case Diagram

First, the top-level functions and use cases are placed on the use case diagram. This gives a starting point for modeling the system. This very basic use case diagram can be seen in Figure 10. The top-level functions of this system are:

1. Set Current Date & Time
2. Choose 12H or 24H Time
3. Set Dispense Times
4. Load Medication
5. Alarm Sounds/Lights Up
6. Press Dispense Status Button
7. Alarm & Lights Turn Off
8. Take Medicine.

There are two actors interacting with this system: the user with MCI (A) and their care partner (B). The care partner is responsible for the set-up tasks associated with the system: setting the current time, whether to display the time in 12H or 24H format, which times to alert the user to dispense medicine, and to load the medication into the system itself.

The user with MCI interacts with this system only when medication needs to be taken. The system alerts the user with sound and lights when medicine needs to be taken; the user triggers the dispensing, which turns the alarms off, takes the pill cup, takes the medication, then resets the system.

Using these top-level functions, fully-dressed tables can be created to further explore the scenarios within the system. While some functions not listed in Figure 10 may seem like top level functions, they are excluded because of the include and extend interactions.

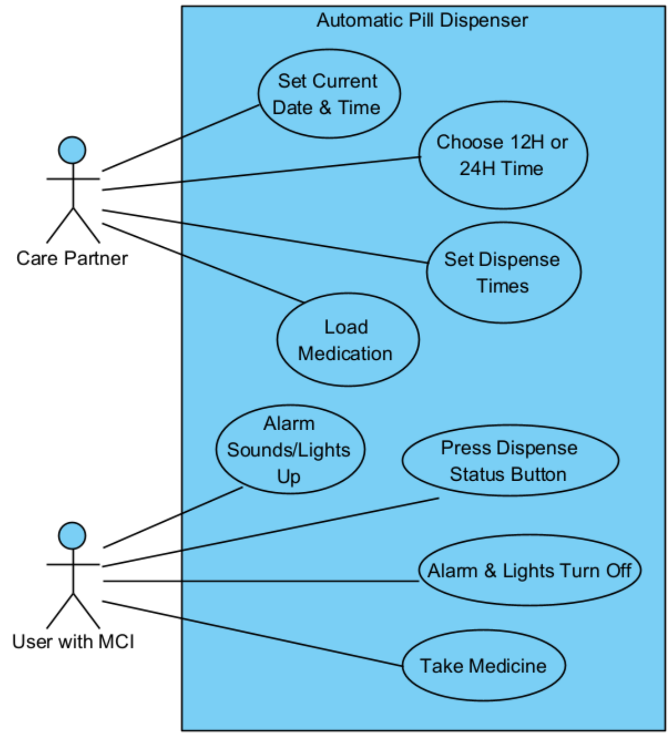

Figure 10: **TOP LEVEL FUNCTIONS IN THE USE CASE DIAGRAM**

The first fully-dressed use case table is for the "Set Dispense Time" use case. This use case is very similar to both "Set Current Date & Time" and "Choose 12H or 24H Time," so only one fully-dressed table is shown. Using this fully-dressed table, shown in Table 3, potential fringe cases such as the care partner not entering a time into the system can be accounted for. Fully-dressed use cases are a helpful tool for designers to guide them through every possible scenario to account for fringe cases such as these.

TABLE 3: FULLY-DRESSED USE CASE FOR "SET DISPENSE TIMES"

System	Automatic Pill Dispenser Box for those with Mild Cognitive Impairment
Name	Set Dispense Times
Actors	Prime: Care Partner
Other Stakeholders	User with MCI
Description	The care partner sets what times the user with MCI will take pills in the morning and evening.
Main Success Scenario	1. Care partner uses buttons to navigate to set up menu. 2. System prompts AM time set-up 3. Care partner enters AM alarm time 4. System prompts PM time set-up 5. Care partner enters PM alarm time
Alternative 3a	In Step 3, if the care partner does not enter a time, AM alarm time is automatically set to 8 AM.
Alternative 5a	In Step 5, if the care partner does not enter a time, PM alarm time is automatically set to 8 PM.
Precondition	The system has been plugged in and is receiving power.
Post Condition	Alarm time for medicine taking is set.
Assumptions	The care partner is able to navigate a basic menu with buttons.
Special Requirements	N/A

TABLE 4: FULLY-DRESSED USE CASE FOR "LOAD MEDICATION"

System	Automatic Pill Dispenser Box for those with Mild Cognitive Impairment
Name	Load Medication
Actors	Prime: Care Partner
Other Stakeholders	User with MCI
Description	The care partner loads each serving of pills into their corresponding cup at the beginning of each week.
Main Success Scenario	1. Care partner pushes button to choose which time to load (for AM or PM). 2. Care partner presses the dispense button 3. All cups for chosen AM/PM eject. 4. Care partner loads medication. 5. Care partner pushes dispense button to close all compartments.
Alternative 5a	In Step 4, if the care partner doesn't push the button to close the compartments, they will automatically close after 15 minutes.
Precondition	The system has been plugged in and is receiving power.
Post Condition	All medication for the week is loaded in the system.
Assumptions	The care partner is able to navigate a basic menu with buttons and has the required medications.
Special Requirements	N/A

The last use case explored is "Load Medication." The care partner achieves this by navigating the menu and pushing the dispense button to eject all the cups for either the morning or evening to load the medication servings for the week. This use case table can be found in Table 4.

Next, a fully-dressed use case table can be made for the "Take Medication" use case, shown in Table 5. This use case's primary actor is the user with MCI. Here, the user is prompted via auditory and visual cues to press the dispense button at the right time. The right time's cup is then ejected for the user to take. Having a fully-dressed table allows the designer to specify assumptions in use cases such as this where the system's functions are contingent on the user being able to take either visual or auditory cues from the system.

Using the main success scenario steps, a fully-dressed use case diagram can be created. The fully-dressed use case diagram for this pill box system can be found in Figure 11. Similar use cases can be consolidated into one bubble and linked to other use cases using include and extend interactions. This fully dressed use case diagram adds the following use cases to the system:

1. Navigate Menu
2. Dispenser Closes
3. Dispenser Opens
4. Remove Pill Cup
5. Return Pill Cup

Include interactions are used for use cases that are similar behaviors occurring at various points in the system. Setting current, 12H/24H, and dispense times all require navigating the

TABLE 5: FULLY-DRESSED USE CASE FOR "TAKE MEDICATION"

System	Automatic Pill Dispenser Box for those with Mild Cognitive Impairment
Name	Take Medication
Actors	Prime: User with MCI
Other Stakeholders	Care Partner
Description	User with MCI retrieves and takes their medication at the right times.
Main Success Scenario	1. Alarm sounds and lights up to alert user. 2. User presses dispense button. 3. Dispenser opens and ejects cup and the alarm turns off. 4. User removes cup and takes medicine. 5. User returns cup. 6. User presses dispense button again to close dispenser.
Alternative 2a	In Step 2, if the user does not dispense the cup, the alarm will not turn off.
Alternative 4a	In Step 4, if the user does not remove the cup to take medicine, the alarm will turn on again to prompt return.
Alternative 6a	In Step 6, if the user does not trigger the dispenser to close, it will automatically close after 15 minutes.
Precondition	The medicine has been pre-loaded by the care partner.
Post Condition	The user with MCI has taken the correct medication.
Assumptions	The user with MCI is able to take either visual or auditory cues from the alarm.
Special Requirements	N/A

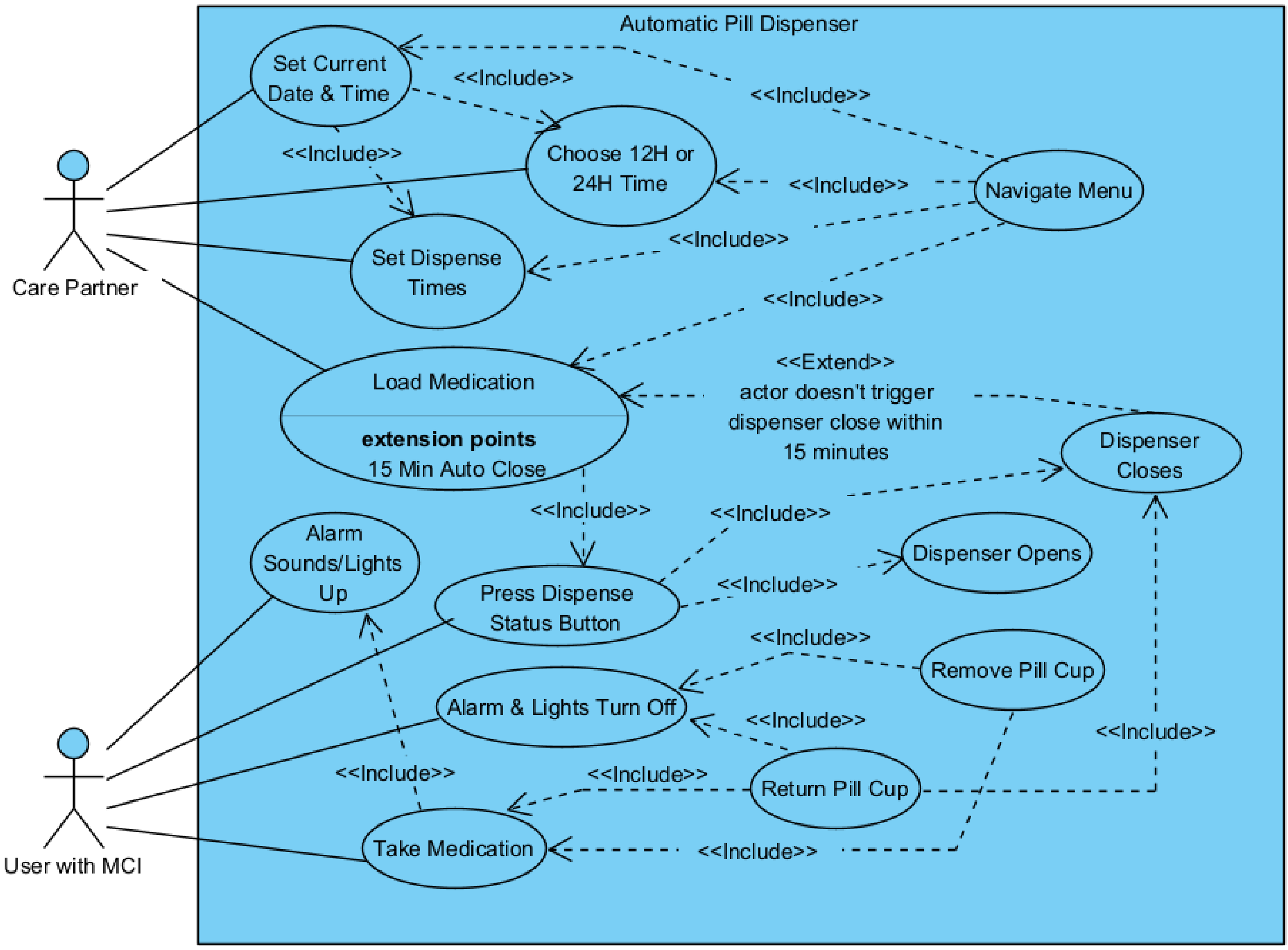

FIGURE 11: FULLY-DRESSED AUTOMATIC PILL DISPENSER USE CASE DIAGRAM

menu. To reflect this, the “Navigate Menu” use case is simply included within the set time use cases using a dotted line labeled “<<Include>>.” This interaction is used for many use cases such as the dispenser opening and closing and removing and returning the pill cup.

Extend interactions are used for use cases that may not happen during every interaction with the system. In this system, the dispenser only closes on its own if an actor does not press the dispense button to close it within 15 minutes. This interaction is shown as a dotted arrow as well, marked instead with “<<Extend>>” and labeled with the extension condition.

While fully-dressed tables can be made for every use case, the majority of the system has been represented here with these three as an example and proof of concept. Creating additional fully-dressed tables is a very similar procedure to what has already been done for these top-level functions.

5.2 Advantages of Use Case UML

Creating fully-dressed tables allows designers to walk through each scenario step-by-step similar to mapping out a process flow diagram. This makes it an invaluable tool for designers to be able to imagine and see how a user would potentially interact with the system they are designing. By doing this walk-through themselves, designers can potentially avoid forgetting to design for necessary functions.

Creating these main success scenarios also offers the opportunity to flesh on alternatives. By putting this in the design process, designers can identify fringe cases and failure modes and accommodate for them accordingly.

Finally, creating fully-dressed use case tables helps to highlight similar functions between use cases. Highlighting these various include and extend interactions is a vital part of the design process as it means these sub-systems interact and are intertwined with each other, providing more opportunities for potential failure modes and complexities. If all these functions happened sequentially and fully isolated, designers would not need to ensure that they function perfectly in tandem.

For example, if the dispenser on the pill-box only needed to open and close whenever the button was pressed, the system would be fairly isolated with little room for error. However, the dispenser interacts with a timer, a calendar, a buzzer, lights, and multiple actors. Mapping out each of the functions that requires the dispenser to operate properly lets the designer have a wholistic view of the system requirements.

6. CONCLUSION

Use case UML is an incredibly useful tool for designers to implement into their design process to model the interactions of their system. It allows designers to systematically review their design, account for fringe cases, and identify frequent interactions between multiple use cases. Further exploration can be conducted towards implementing this into more design processes and evaluating the resulting prototype functionality rather than this application case study towards an existing project.

ACKNOWLEDGEMENTS

I thank Alyson Riben, my partner in this project during ID 4833, as well as Dr. Leila Aflatoony, the instructor with whom we completed this project and facilitated the collaboration with CEP. I also thank Jennifer DuBose, the director at CEP, for allowing us to collaborate with the program.

I thank the participants in this research from CEP, who must remain anonymous but without whom the project could not have happened.

Finally, I thank Dr. Jianxin "Roger" Jiao for his valuable instruction during ME 6101 that made this paper possible.

REFERENCES

[1] S. Gauthier *et al.*, "Mild cognitive impairment," *Lancet*, vol. 367, no. 9518, pp. 1262–1270, 2006, doi: 10.1016/S0140-6736(06)68881-8.

[2] "Unified Modeling Language," *Wikipedia*, 2005. .

[3] J. R. Jiao and Y. Liu, "Tutorial : Analysis and Modeling with UML," *Georgia Institute of Technology*. pp. 1–31, 2017.

[4] J. Liu, "ME 6101 Assignment 2 - Use Case UML," *Georgia Institute of Technology*, no. September. pp. 1–6, 2021.

[5] "Requirements and Use cases," *California State University, San Bernardino*, 2012. .