# Team 7 - Warehouse Robot Operator Usability Study Report

Matt Bekhterev

Trevor Glennon

Jason Li

Shuge Luo

Zibo Shang

June 2nd, 2022

## Evaluation and Analysis Results (3~4 pages)

*a)* **Topic**:

We have chosen to implement **a warehouse robot delivery system by designing** and using our DSL (Domain Specific Language). Users would have access to a number of instructions/ commands that will allow them to move the robot around, transport items and order new items to be delivered. This should **streamline the warehouse workers' working process by achieving the warehouse delivery automation.**

b) **Evaluation Goals**:

1. Is our DSL design **easy to learn and understand**? Is it easier to adapt than the coding language like Java?
2. Is our DSL design **easy to use** on a daily basis? How **effective** is the DSL for performing daily tasks?
3. Is there a **sufficient amount of feedback** after each action is completed for users?

*c)* **Summary of evaluation**:

For our evaluation, we had **two participants** in total. Each participant came from a non-CS background, although all of them had experience with symbol coding before, such as Excel function or highschool CS electives. They all had access to a laptop or computer for our online evaluation.

We chose to perform **an observational study** of the participants with participants **saying out loud their thoughts** while performing tasks and also a small **interview**, so that we can understand their thought processes and their performed actions better. The developed function description sheet allowed users to reference all the available functions that would help to guide their coding process.

Each evaluation session took around 15 minutes to complete for each participant.

*d)* **Reference of functions and summary of examples:**

The github document "Warehouse Manager Documentation.pdf" is given to the tested users for reference and understanding the DSL.

*e)* **Task and Sample Solution:**

1. Read through the reference sheet("Warehouse Manager Documentation.pdf" ), ask any questions you have, and tell me what you think after seeing and learning the language.
2. Create an order "customerOrder2", which includes three coffees and one watermelon, then assume all the items are in stock, grab all the things, bring them to the front house, and fulfill the order.
3. Now, you do not know if the items in the order are in stock or not, so modify task 2's answer to check for items' availability and if in stock, grab all the items; if not in stock, refill the item and then grab it to fulfill the order.
4. Add 5 pizzas to your order.

5. Follow up interview: How do you feel using this language for the tasks? How do you feel using this language for daily use in warehouses in the real world?

Task 2's solution:

```
create order customerOrder2 with 3 Coffee, 1 Watermelon;
create products producta; //the name given have no constraints
pickUp 3 Coffee to products producta;
pickUp 1 Watermelon Coffee to products producta;
goTo fronthouse;
fulfill customerOrder2 with products producta;
```

Task 3's solution:

```
create order customerOrder2 with 3 Coffee, 1 Watermelon;
create products producta;   //the name given have no constraints
// the easiest solution
ifNot check availability of products 3 Coffee{
        restock 3 Coffee;
}
pickUp 3 Coffee to products producta;

// the complicated solution
if check availability of products 1 Watermelon{
        pickUp 1 Watermelon to products producta;
}
ifNot check availability of products 1 Watermelon{
        restock 1 Watermelon;
        pickUp 1 Watermelon to products producta;
}
goTo fronthouse;
fulfill customerOrder2 with products producta;
```

Task 4's solution:

```
Add 5 Pizza to order customerOrder2
```

*f)* **Each task's result for two tested users:**

**Task 1:**

**User a:** The reference sheet is easy to understand and learn. All the names given are intuitive, except the “create products name”. From the use of the sentence, he can know it is for creating a product basket, but maybe changing “products” to something like “productsGrabbed” or “productBasket” could be better. But in general, all the grammar is similar to English, and he can imagine using it in a real world warehouse.

**User b:** He asked what is meant by **“**if check availability of products 3 Coffee{...}” and “ifNot check availability of products 3 Coffee{...}” because he understood and interpreted the language more in a natural language style and order, so he was confused in “why do I need to **not** check the availability of a product”. After explanation, the if and ifNot statement is clear to him. All the other function names and sentences are intuitive, and can be understood and remembered after reading through the reference sheet.

**Task 2:**

**User a:** The task is simple with the reference sheet given. He tried to combine the two add products to order commands into one, and later found out it cannot be done like that, and changed to the right solution.

**User b:** The logic of the language is simple for completing the task, simply “create order->create product bucket-> add to bucket -> go to front house and finish the order”. He can remember and write most of the commands without referencing the sheet, but made an error in his answer: for creation, “products” with “s” and “order” without “s”.

**Task 3:**

**User a:** The if and ifnot statement is easy to use and understand. Also tried to combine the if and ifnot together, and then found out those statements cannot be combined. Then, he tried to find the easiest way to write the commands, and succeeded.

**User b:** He followed the logic given in the question, first giving the “if…” and then “ifnot …”, and felt like the two commands are similar and redundant. After telling him that there could be simpler solutions, he figured it out after some thinking.

**Task 4:**

**User a:** The task is really easy by following the reference.

**User b:** The task is easy, but he made a minor mistake of adding plurals “pizzas” instead of using the correct version “pizza”. Although he checked with the reference sheet, he did not notice that plural is not required until we pointed it out.

**Task 5:**

**User a:** He can imagine using it in a real world setting, and “it is easy to learn and understand, so after maybe a few hours of using the language, I can use it smoothly without the help of a reference sheet”. In general, he likes the language design, especially all the names given to the commands are simple and describe exactly the action required, but it could be more simplified, like combining picking up items in two one command, which would save more time and become more efficient in daily use.

**User b:** For daily use, there could definitely have more functions, like deleting an order. There could be more consistency in terms of the plural because it is easy to mis-remember something and make mistakes, and those mistakes are hard to find out. The language can

have a better design in terms of the if and ifnot because for a senior worker who has no experience with programming language, the logic is not intuitive, so the language can be hard to learn. But once you learn it, it is easy to use in real world settings.

**Result and Conclusions :**

The user study in general went out smoothly, both users completed tasks successfully within the time limit (15 min). Thanks to the "Warehouse Manager Documentation", they had no problem with writing statements like creating orders, creating products, adding items and goto locations.

The language's strengths are:

1. The name is **intuitive, accurate and easy to understand** with no extra explanations required, like "pickUp", "restock", "add".
2. The language is **easy to learn** as the logic is similar to natural languages.
3. The language is **easy to use** on a daily basis in the real world because the language **can be remembered easily,** and the commands needed for completing tasks are **logical**.
4. The **feedback** given after commands entered are **sufficient** and users are not lost or not knowing what to do next.
5. In general, **users can imagine them using the language on a daily basis with no difficulties.**

The language can be improved by:

1. Commands **can be simplified and more efficient by combining** because they are the same, but with different products, like picking up items and checking availability in the if statement.
2. The **if and ifnot** statement could be improved by having a **more natural-language-like expression** as the current command can be confusing to someone who has no experience with programming.
3. Details: the commands could be **more consistent** in terms of the **plurals**. The "create products productname" could **have a better name**, e.g. "productBucket".