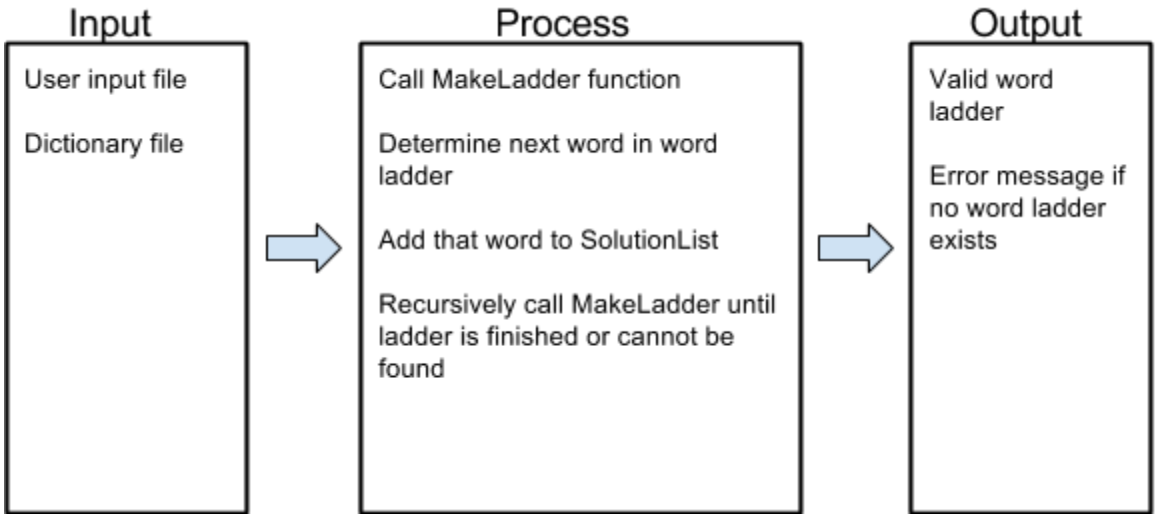
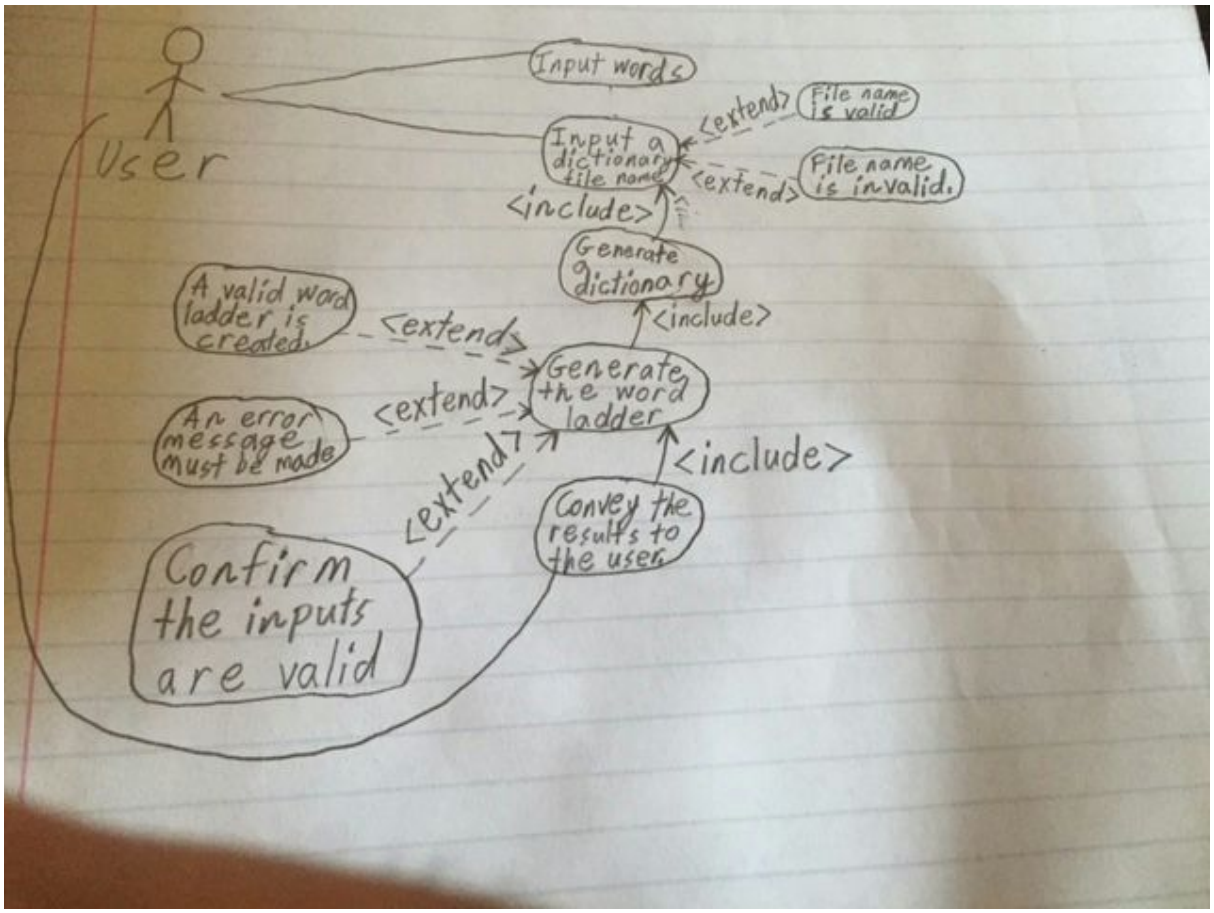
Mukundan Kuthalam, Caroline Yao
UTEID: mk33274, chy253
EE422C - Assignment 4

Lab 4 Analysis and Design

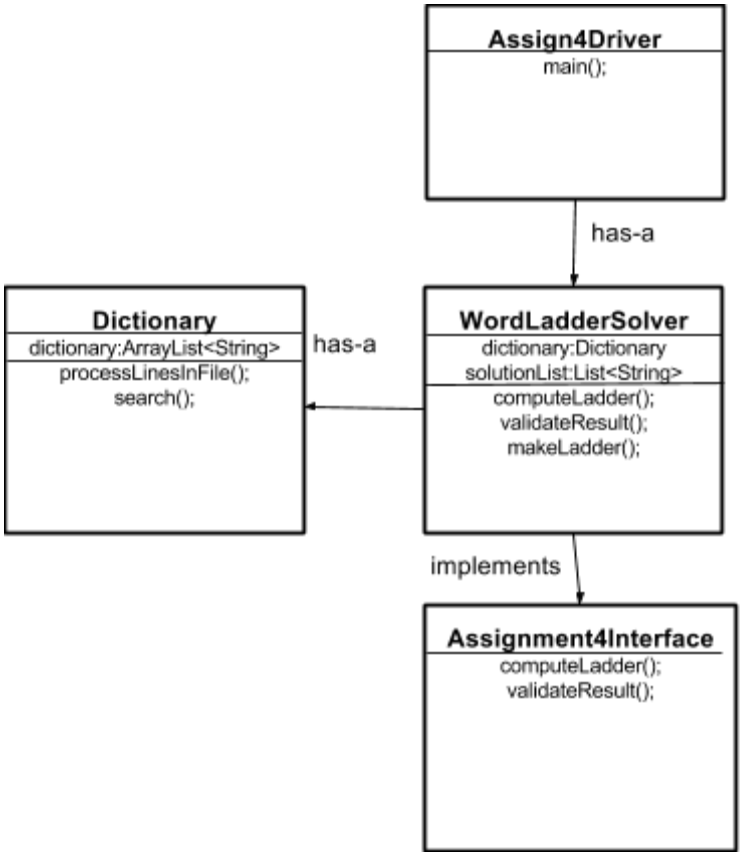
1. A System IPO diagram



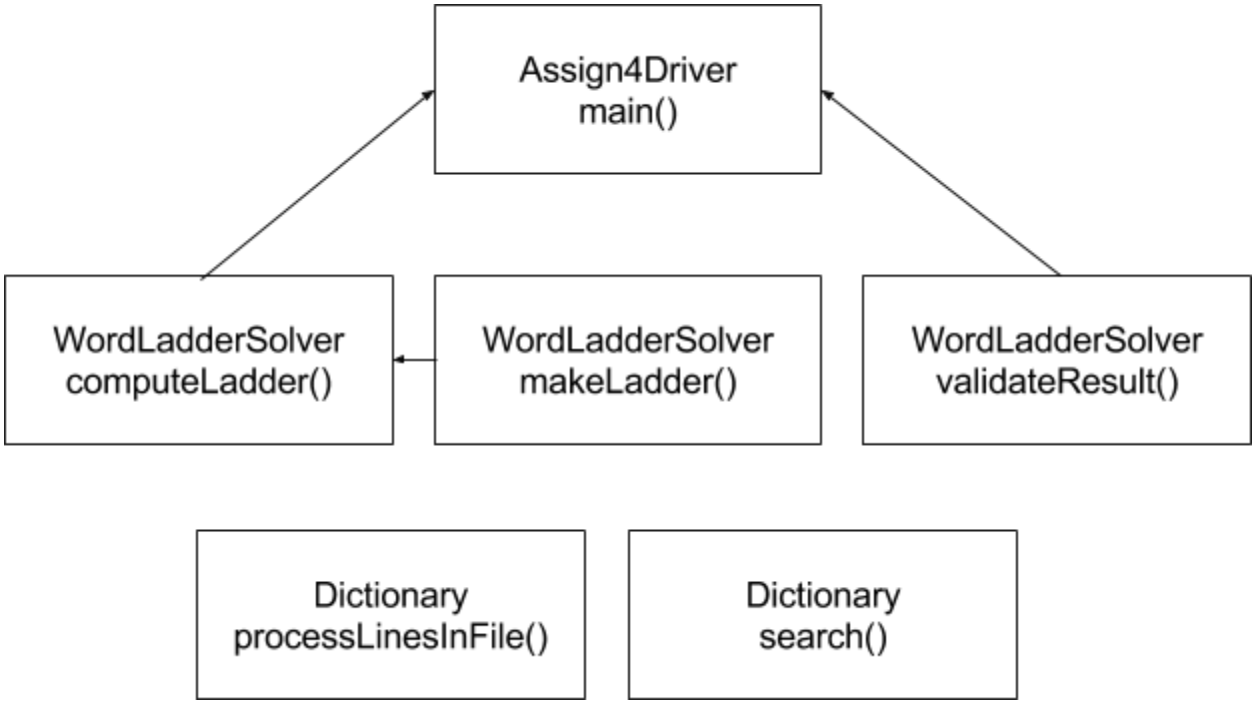
2. A Use-case diagram



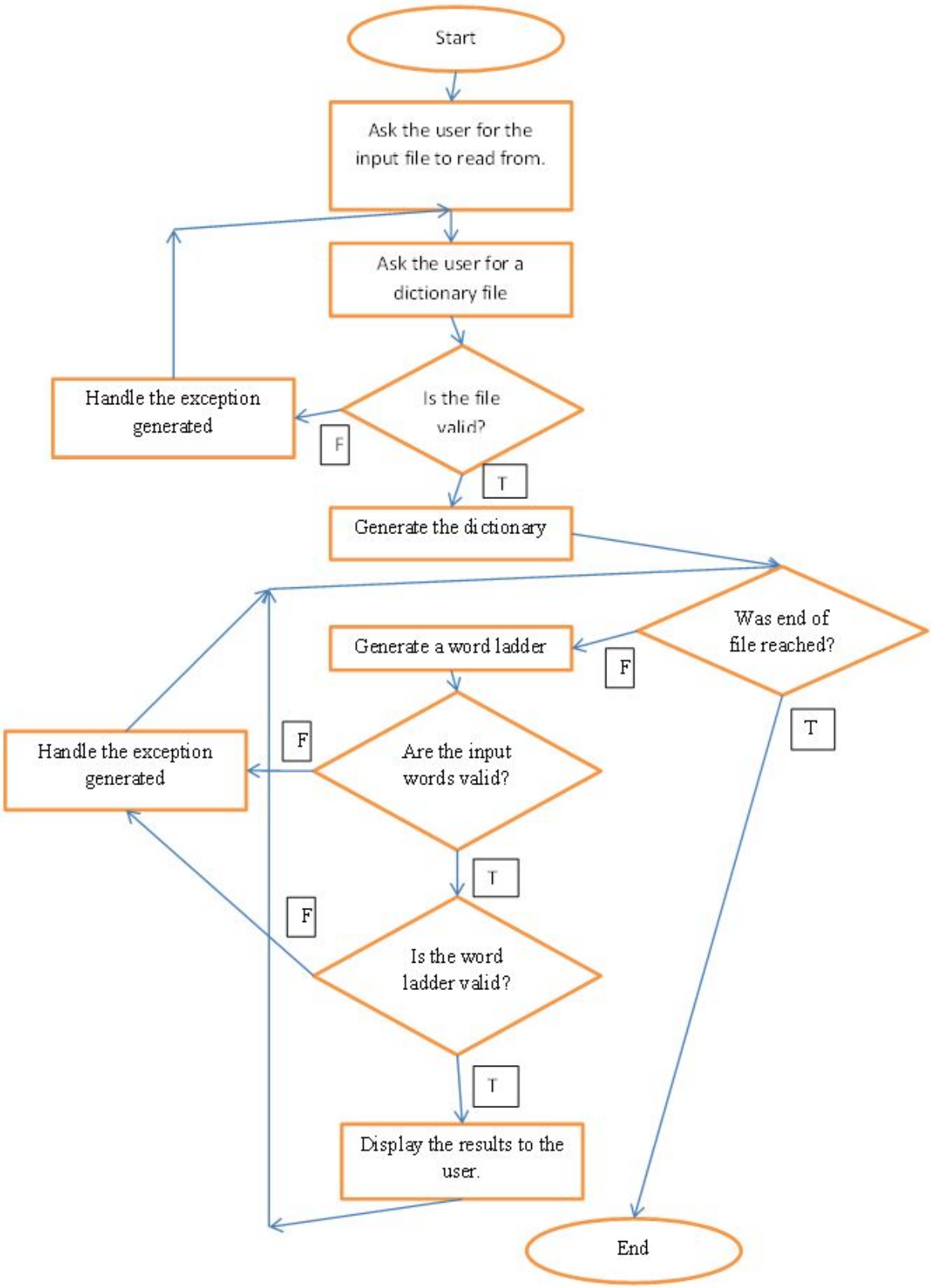
3. A UML model of the needed classes and their relationships



4. A functional block diagram showing the calling relationships between methods (library methods are not included)



5. The algorithm needed for the driver logic (main method)


Start
Ask the user for the input file to read from.
Ask the user for a dictionary file
Is the file valid?
F
Handle the exception generated
T
Generate the dictionary
Was end of file reached?
F
Generate a word ladder
T
Are the input words valid?
F
Handle the exception generated
T
Is the word ladder valid?
F
T
Display the results to the user.
End

6. A paragraph describing the rationale behind your design. This would include: a) How does your OOD reflect the interaction and behavior of the real-world objects that it models b) What alternatives did you consider? What were the advantages/disadvantages of each alternative both from a programming perspective and a user perspective? c) What are some expansions or possible flexibilities that your design offers for future enhancements? d) How does your design adhere to principles of good design: OOD, cohesion, coupling, info hiding, etc?

One important thing to note about the objects that compose this program's design is that all these objects can be reused for different purposes. That is, the Driver can naturally execute the program however the user wishes. The WordLadderSolver class is something that performs operations on words and phrases, but not any data type, so it makes sense for it to be its own class. Now, the real question is why the Dictionary and Assignment4Interface need to be separate from other classes. The Dictionary class could simply be an ArrayList inside the WordLadderSolver. However, this means that a coder would have to touch the class that looks for the ladder, instead of adjust Dictionary, which is all they would want to do. So it makes sense for Dictionary to be its own class, as it alone could need to be modified (what if you had a dictionary of French words and not English? This would mean Dictionary operations could work differently). The Assignment4Interface offers a couple of methods for creating a word ladder. However, the key here is that it does not say what the word ladder must be made up of. In this case you want to just use String objects, but if you want to create a ladder for words and characters separately using the same method signatures, it makes sense to make a interface that is implemented by two different classes. That is why Assignment4Interface is an interface - so that it can be used by other classes in the future. For these reasons, the team decided to design the code as detailed above, so that any part of the Assignment 4 code can be used elsewhere.