Code report:

One of the problem that existed in the first version of my code was that machine was accepting any answer for each question if the input was in grammar. For example the answer “Thursday” was acceptable answer for question “Who are you meeting with?”. To resolve this problem I created the function “isInGrammar_spec”, that not only checks if the object is in grammar, but also if the object has correct category. For example “person” is an only correct category for answer for question “Who are you meeting with?”.

At first “checkgrammar” was an independent state. I was trying to move to it after each state that was asking an question. It was very hard though to implement some artificial history method. I was trying at first in each state with question to implement variable “next_question”. That was supposed to help the program to move between “checkgrammar” stage and other question stages. It didn’t work though. Now I know I could use “type:history” stages. In the end I just created “checkgrammar” as a substate in each main question state.

One of the trivial problems was that my machine wasn’t recognising am and pm hours. Then I realised that in “isInGrammar_spec” function I was using “.toLowerCase()” method. I just had to change all PM and AM in my grammar to lowercase.

I also realise that my machine can recognise answer incorrectly with its ASR system. So when in my code machine is summing up the details of the meeting, user can confirm or not the appointment. After negative answer(“no”) user can choose, in “repeat” state, if they want to try to create an appointment again. Now even if ASR doesn’t recognise something correctly, user can repeat the process.