

Language Detection Web Application – Duck Soft Works § Co.

US 4 – As a user, I want to cancel a task that is currently being processed

1. Requirements Engineering

1.1. Customer Specifications and Clarifications

Q: “What does it mean for the task to be “expressly canceled”?”

R: “It means that the user “expressly” cancels the task (by clicking a cancel task button on the application frontend).”

1.2. Acceptance Criteria

A task can only be canceled by a user with user privileges.

Cancel method can proceed and change the status of a task to “Canceled” if the task is still with a “Processing” status.

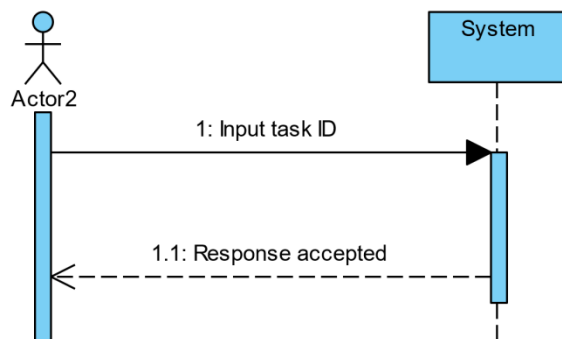
1.3. Found out Dependencies

To fulfill the cancellation method, one task with status “Processing” must exist.

1.4. Input and Output Data

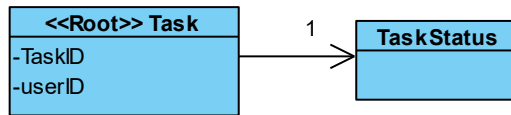
In order to cancel a task, the user must provide the task id.

1.5. System Sequence Diagram (SSD)



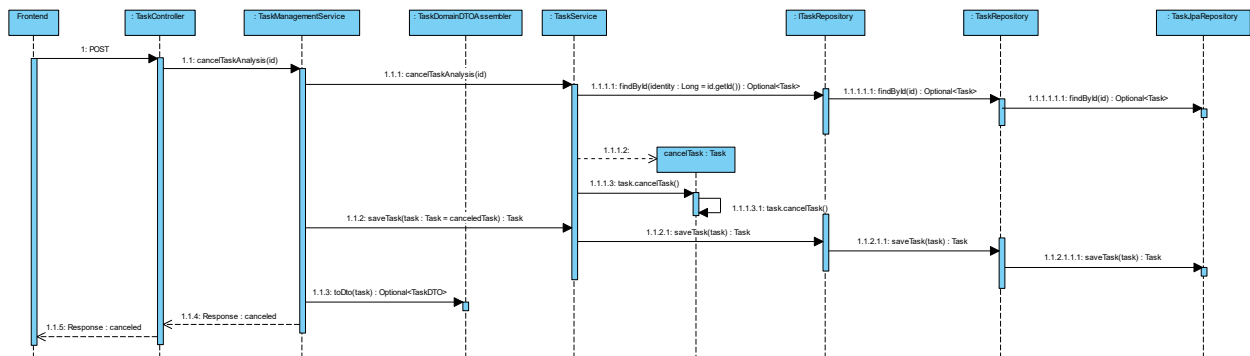
2. OO Analysis

2.1. Relevant Domain Model Excerpt

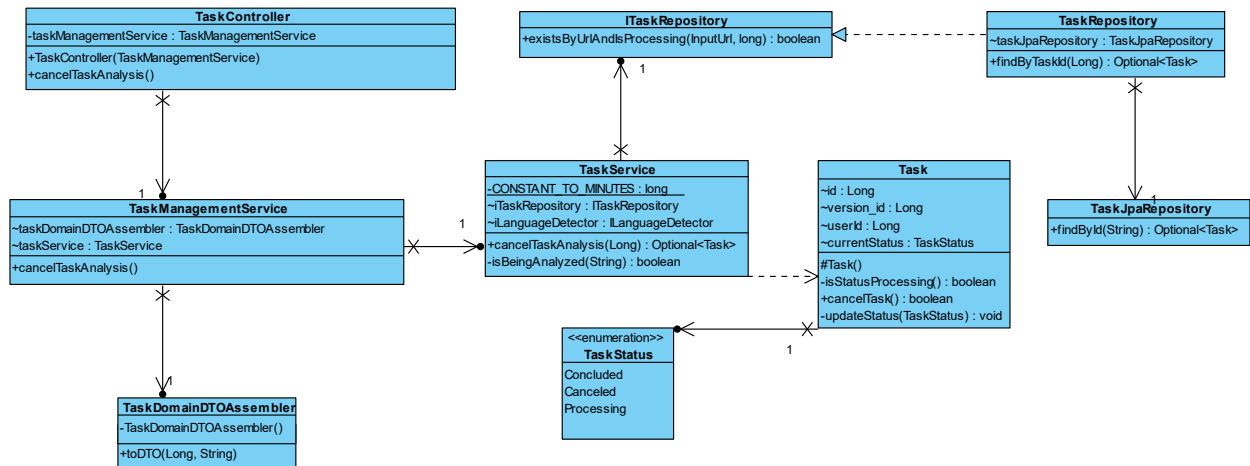


3. Design - User Story Realization

3.1. Sequence Diagram (SD)



3.2. Class Diagram (CD)



4. Tests

```
@Test
void
ensureCancelTaskAnalysisIsCanceledWhenTaskExistsAndTaskStatusIsProcessing(
) throws MalformedURLException,
        NoSuchMethodException, InvocationTargetException,
InstantiationException, IllegalAccessException {
    //arrange
    String url = "https://www.w3.org/TR/PNG/iso_8859-1.txt";
    int inputTimeOut = 1;
    String inputCategory = "Sports";
    Category category = new Category(inputCategory);
    Timeout timeOut = new Timeout(inputTimeOut);
    InputUrl inputUrl = new InputUrl(url);

    Constructor<Task> taskConstructor =
Task.class.getDeclaredConstructor(InputUrl.class, Timeout.class,
        Category.class, Long.class);
    taskConstructor.setAccessible(true);
    Task task = taskConstructor.newInstance(inputUrl, timeOut, category, 1L);
    Optional<Task> opSavedTask = Optional.of(task);

    try(MockedStatic<UserDetailsDomainService> utils =
Mockito.mockStatic(UserDetailsDomainService.class)) {
        utils.when(() -> iTaskRepository.findByTaskIdAndUserId(any(),
any())) .thenReturn(opSavedTask);
        when(iTaskRepository.saveTask(opSavedTask.get())) .thenReturn
(opSavedTask.get());

        //Act / Assert
        Assertions.assertEquals(taskService.cancelTaskAnalysis(1L),
opSavedTask);
    }
}
```

5. Construction (Implementation)

As we faced several problems when trying to cancel an ongoing thread, construction of this user story was not easy. After trying to interrupt or kill one thread with several methodologies without success, we chose to just change the task state from **Processing** to **Canceled** directly through a save in the database. For this, we use a method in Task that changes the Task Status from “Processing” to “Cancel”.

6. Integration and Demo

To integrate this with the rest of the application, before modifying the object's state to **Canceled**, we verify the Task ID, the user ID and the Status.

In the Frontend, a user creates a Task and may navigate to the “Cancel Task” web page to try to cancel it. If it is still in “Processing”, the user may insert the task ID to proceed to the cancelation.




Ducked Language Detection

Cancel

Your Processing Tasks

Task ID	Date	Text Category	URL	Text Language	Task Status	Time
9	09:41:37 28-06-2022	Philosophy	file:/C:/Users/inesc/Desktop/texto_longo.txt		Processing	1



Ducked Language Detection

Cancel

The task with ID 9 was successfully canceled! ✕

Your Processing Tasks

Task ID	Date	Text Category	URL	Text Language	Task Status	Time
No tasks are being processed at the moment!						

6. Observations

We are aware that this implementation is not ideal, however, given the amount problems faced about the inability to interrupt the asynchronous process of language analysis, we chose to solve it the way previously explained.

We are aware that there is a performance issue as the request to cancel the analysis does not interrupt the process and, therefore, continues to use the machine's resources.