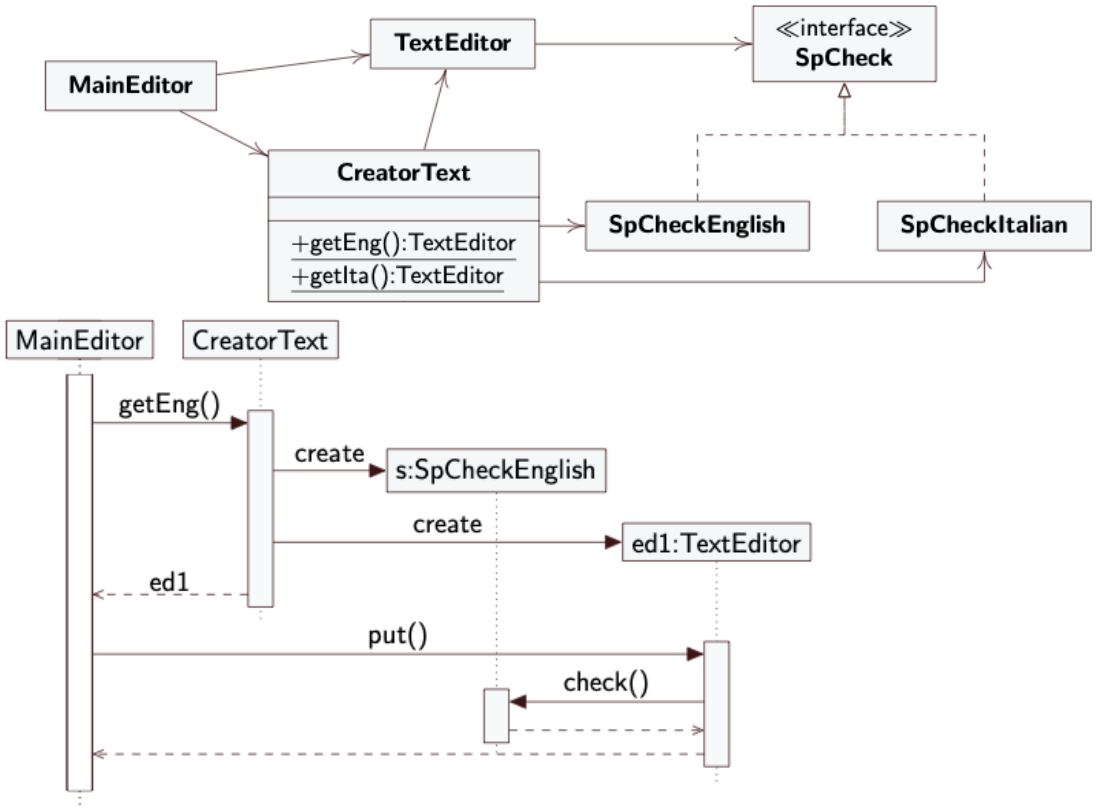


```
// Codice Java che mostra l'uso di dependency injection e object pool con il pat

/** SpCheck svolge il ruolo di Product */

public interface SpCheck {
   /** Controlla che la parola usata appartenga al dizionario */
   public boolean check(String word);
}


/** SpCheckEnglish svolge il ruolo di ConcreteProduct */

public class SpCheckEnglish implements SpCheck {

   @Override
   public boolean check(String word) {
      // TODO: Dovrebbe usare il dizionario inglese e le regole per stemming,
      // troncamento, etc.
      return word.equals("hello");
   }

}


/** SpCheckItalian svolge il ruolo di ConcreteProduct */

public class SpCheckItalian implements SpCheck {

   @Override
   public boolean check(String word) {
```

```
      // TODO: Dovrebbe usare il dizionario italiano e le regole per stemming,
      // troncamento, etc.
      return word.equals("ciao");
   }
}


/**
 * TextEditor conosce l'interfaccia SpCheck e non le sue implementazioni. La
 * dependency injection permette il disaccoppiamento fra TextEditor e la
 * sottoclasse di SpCheck.
 */

public class TextEditor {
   private SpCheck speller;

   public TextEditor(SpCheck sp) {
      speller = sp;
   }

   /** inserisce una parola e controlla lo spelling */
   public void put(String word) {
      if (speller.check(word)) System.out.println("lingua usata OK :)");
      else System.out.println("lingua usata NON OK :(");
   }
}


import java.util.ArrayList;
import java.util.List;

/**
 * CreatorText svolge il ruolo di ConcreteCreator e usa la dependency injection
 * per la classe TextEditor. Vantaggio: TextEditor non conosce il tipo
 * dell'istanza usata, e si ha un unico posto per istanziare TextEditor e la
 * sottoclasse di SpCheck.
 **/

public class CreatorText {
   private static List< TextEditor > pool = new ArrayList< >();
   public static TextEditor getEng() {
      System.out.println("Crea un TextEditor con English");
      return new TextEditor(new SpCheckEnglish());
   }

   public static TextEditor getIta() {
      System.out.println("Crea un TextEditor con Italian");
      return new TextEditor(new SpCheckItalian());
   }

   // TODO: Aggiornare il diagramma UML delle classi
   /** ritorna, se esiste, una istanza dal pool */
```

```
    public static TextEditor getFromPool() {
        if (!pool.isEmpty()) return pool.remove(0);
        return new TextEditor(new SpCheckEnglish());
    }

    // TODO: Aggiornare il diagramma UML delle classi
    /** rilascia l'istanza */
    public static void release(TextEditor t) {
        pool.add(t);
    }
}


// classe con il metodo main
public class MainEditor {
    public static void main(String args[]) {
        TextEditor ed1 = CreatorText.getEng();
        System.out.println("Inserisce hello su Editor con spelling check inglese")
        ed1.put("hello");

        TextEditor ed2 = CreatorText.getIta();
        System.out.println("Inserisce ciao su Editor con spelling check italiano")
        ed2.put("ciao");

        System.out.println("Inserisce hello su Editor con spelling check italiano"
        ed2.put("hello");

        TextEditor ed3 = CreatorText.getFromPool();
        ed3.put("word");
        CreatorText.release(ed3);
    }
}
```