



Laurea Magistrale in informatica - Università di Salerno  
Corso di *Ingegneria Gestione ed Evoluzione del Software*

# Strategic Robots Integration Test Design

Versione	1.0
Data	08/09/2021
Destinatario	Prof. Andrea De Lucia Dott. Fabiano Pecorelli Dott. Emanuele Iannone Dott. Manuel De Stefano
Presentato da	Antonio Martella



## Sommario

1.	Introduzione .....	3
2.	Dettagli per il testing di unità .....	4
2.1	Features da testare.....	4
2.2	Features da non testare .....	4
2.3	Strategia.....	4
2.4	Pass/fail criteria .....	5
2.5	Criteri di sospensione e ripristino.....	5
3.	Unit Test Case .....	6
3.1	UTC_Casella .....	6
3.2	UTC_Scenario.....	7
3.3	UTC_ControllerInterattivo.....	7



## 1. Introduzione

---

Lo scopo di questo documento è quello di pianificare il test di integrazione del sistema Strategic Robots. Verranno riportate le strategie adottate per il testing e le funzionalità testate.

Nel test si integrano le varie componenti del sistema e si verifica che il software risulti funzionante.

L'approccio utilizzato sarà di tipo Bottom-up, verranno prima eseguiti i test di unità.

Successivamente, sarà possibile procedere al test di sistema



## 2. Dettagli per il testing di unità

---

### 2.1 Features da testare

Durante questa fase verranno ricercate le condizioni di fallimento. Le classi che verranno sottoposte al testing di unità sono le seguenti:

- Casella.java
- Scenario.java
- ControlloreInterattivo.java

### 2.2 Features da non testare

Non sarà testata la classe ControlloreInterattivo.java in quanto fa riferimento ad una modalità che verrà disabilitata. Le altre classi rimanenti saranno testate tramite il testing di sistema.

### 2.3 Strategia

La strategia usata per il testing è la strategia Black-Box, che si concentra sul comportamento Input/Output ignorando la struttura interna della componente.

Per minimizzare il numero di test case, i possibili input verranno partizionati attraverso il metodo del Category Partition, solo però alcune funzionalità saranno testate con test di unità.

L'approccio utilizzato per eseguire il test di integrazione sarà di tipo Bottom-up.

Il framework di supporto utilizzato è JUnit.



## 2.4 Pass/fail criteria

Nel caso in cui dovessero riscontrarsi errori durante la fase di testing, si procederà con la correzione dei fault intervenendo direttamente sulle porzioni di codice che generano il problema, ed iterando nuovamente con la fase di unit testing verificando che la correzione non abbia impattato altre componenti.

## 2.5 Criteri di sospensione e ripristino

- Criteri di sospensione

Comprendono tutti quei casi critici di quando gli errori hanno un impatto dannoso sul progresso dell'attività di testing. Esempi possono essere:

- o Fallimento di funzionalità interne
- o Problemi relativi all'ambiente di sviluppo del testing

- Criteri di ripristino

La ripresa del sistema avviene solo quando tali errori vengono risolti, ripartendo dal test case che ha causato l'errore.



## 3. Unit Test Case

### 3.1 UTC\_Casella

Nome della classe da testare	Casella.java
Nome della classe test	ITC_Casella.java
ID Unit Test Case	Metodo
ITC_Casella_01	testGetTipoScenario()
ITC_Casella_02	testIsDrawOggetto()
ITC_Casella_03	testIsDrawEnergia()
ITC_Casella_04	testIsRiempiCasella()
ITC_Casella_05	testGetColCas()
ITC_Casella_06	testDisegnaOggetto()
ITC_Casella_07	testDisegnaBarraEnergia()
ITC_Casella_08	testColoraCasella()
ITC_Casella_09	testDeColoraCasella()
ITC_Casella_10	testSvuotaCasella()



### 3.2 UTC\_Scenario

Nome della classe da testare	<b>Scenario.java</b>
Nome della classe test	<b>ITC_Scenario.java</b>
ID Unit Test Case	<b>Metodo</b>
ITC_Scenario_01	testGetMaxRighe()
ITC_Scenario_02	testGetMaxColonne()
ITC_Scenario_03	testGetGriglia()
ITC_Scenario_04	testGetArrayListOstacoli()
ITC_Scenario_05	testGetArrayListBancoRifornimenti()
ITC_Scenario_06	testmModificaGrigliaCancella()

### 3.3 UTC\_ControllerInterattivo

Nome della classe da testare	<b>ControllerInterattivo.java</b>
Nome della classe test	<b>ITC_ControllerInterattivo.java</b>
ID Unit Test Case	<b>Metodo</b>
ITC_ControllerInterattivo_01	testGetstato()
ITC_ControllerInterattivo_02	testGetStatoMossa()
ITC_ControllerInterattivo_03	testGetRobotCombattenteArray()
ITC_ControllerInterattivo_04	testGetRobotLavoratoreArray()
ITC_ControllerInterattivo_05	testGetInsiemeRobotCombattenteArray()
ITC_ControllerInterattivo_06	testGetInsiemeRobotLavoratoreArray()
ITC_ControllerInterattivo_07	testAggiungiRobot()
ITC_ControllerInterattivo_08	testModificaStatoMossa()