Software Design Documents

Prenotazione Spesa On-Line

EperMercato

Progetto Software Engineering di

BRUNETTI BORGHI EMANUELE

1 Introduction

2 System Overview

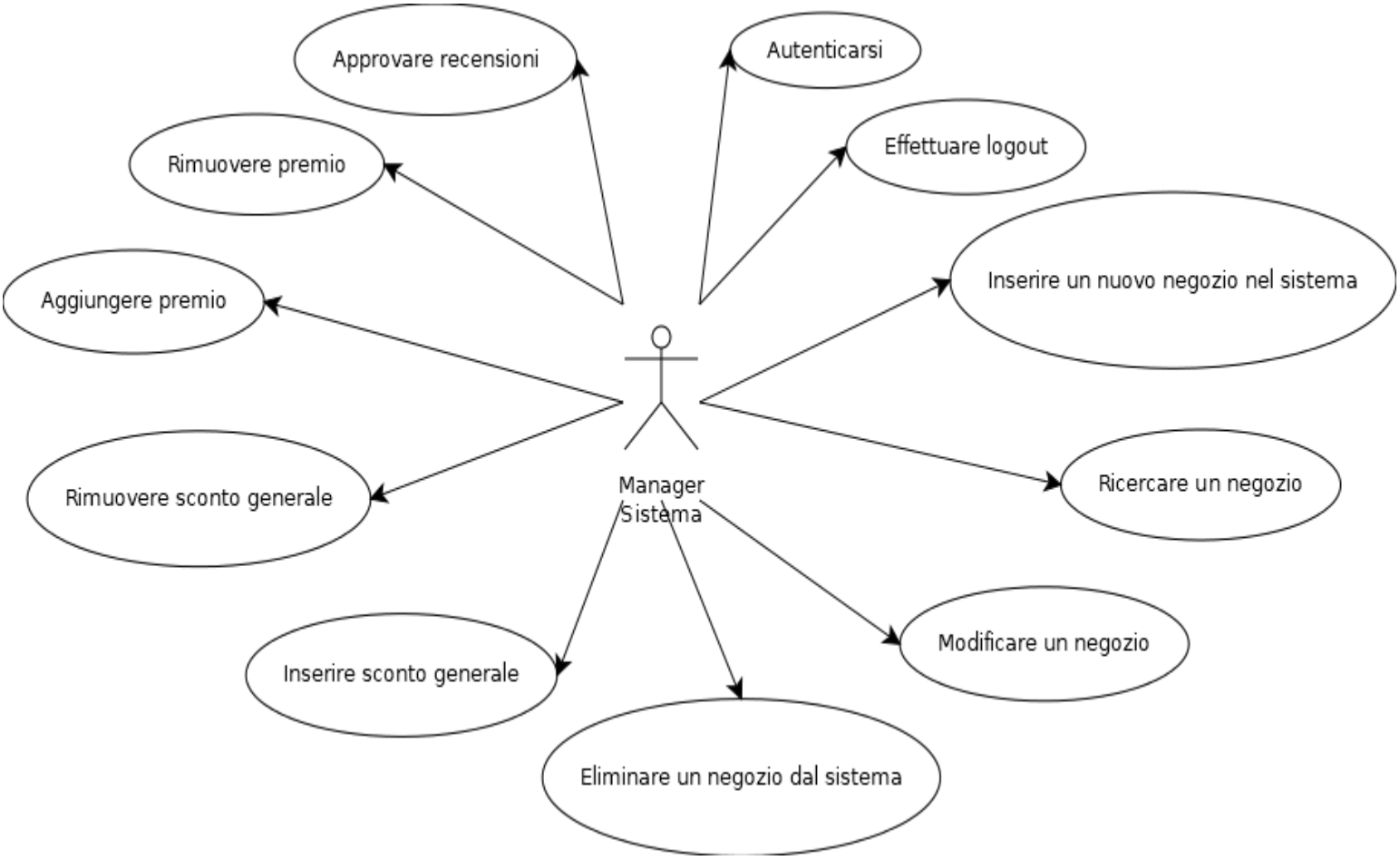
Lo scopo del progetto è quello di descrivere tramite un insieme di modelli UML il Software per un "Sistema di *Gestione Spesa Multinegozio"*, in riferimento al documento "srs_p_13_BrunettiBorghiEmanuele.odt". In questa prima versione, rispetto al sistema descritto nel documento di riferimento, si eseguono le seguenti semplificazioni:

I. Si considererà soltanto il lato del Manager del sistema con tutti i suoi obbiettivi, quindi si implementa l'App Manager del sistema
II. Non si implementerà l'interfaccia grafica

3 Models

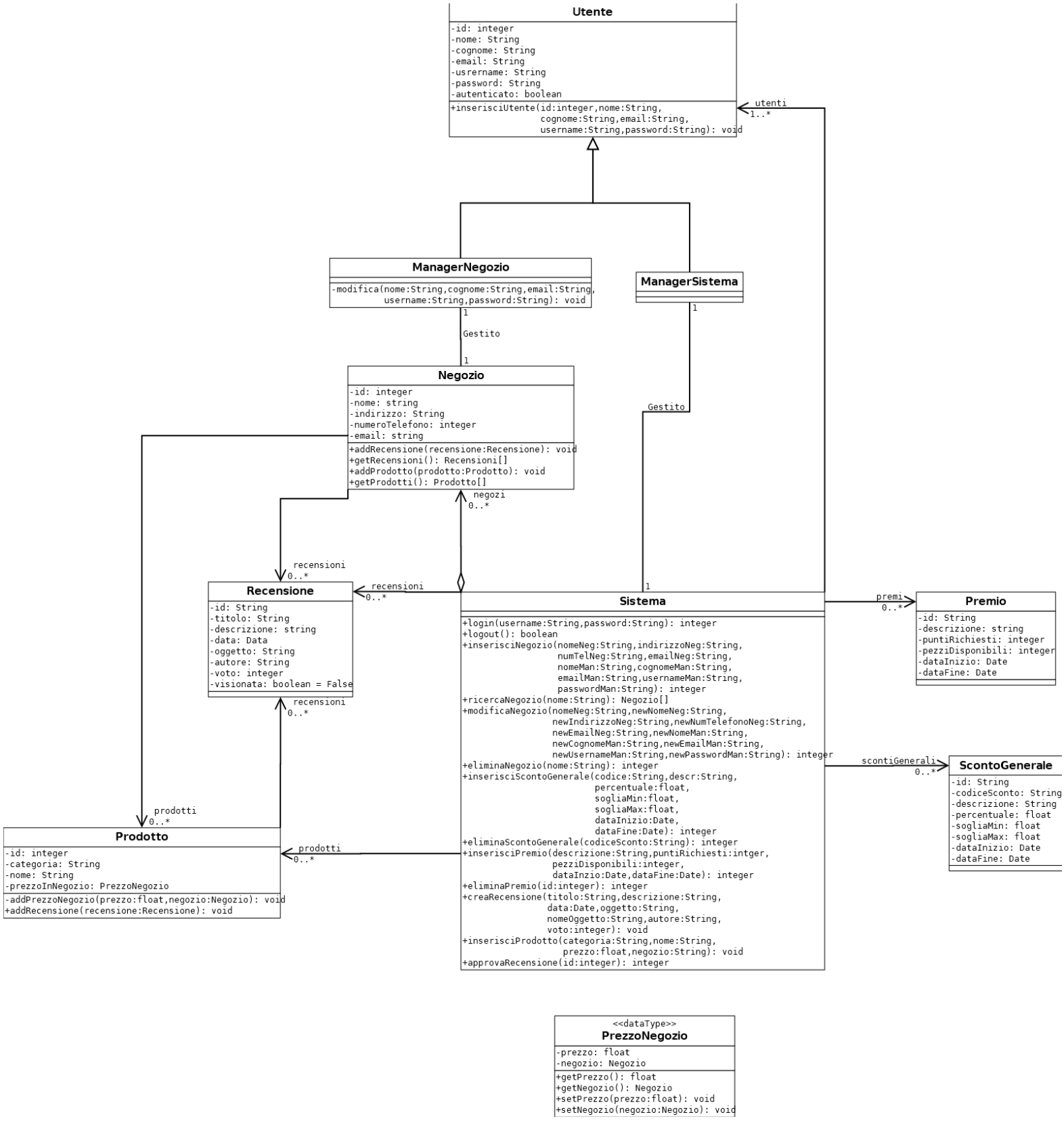
3.1 Static Models

3.1.1 Use Cases Diagram

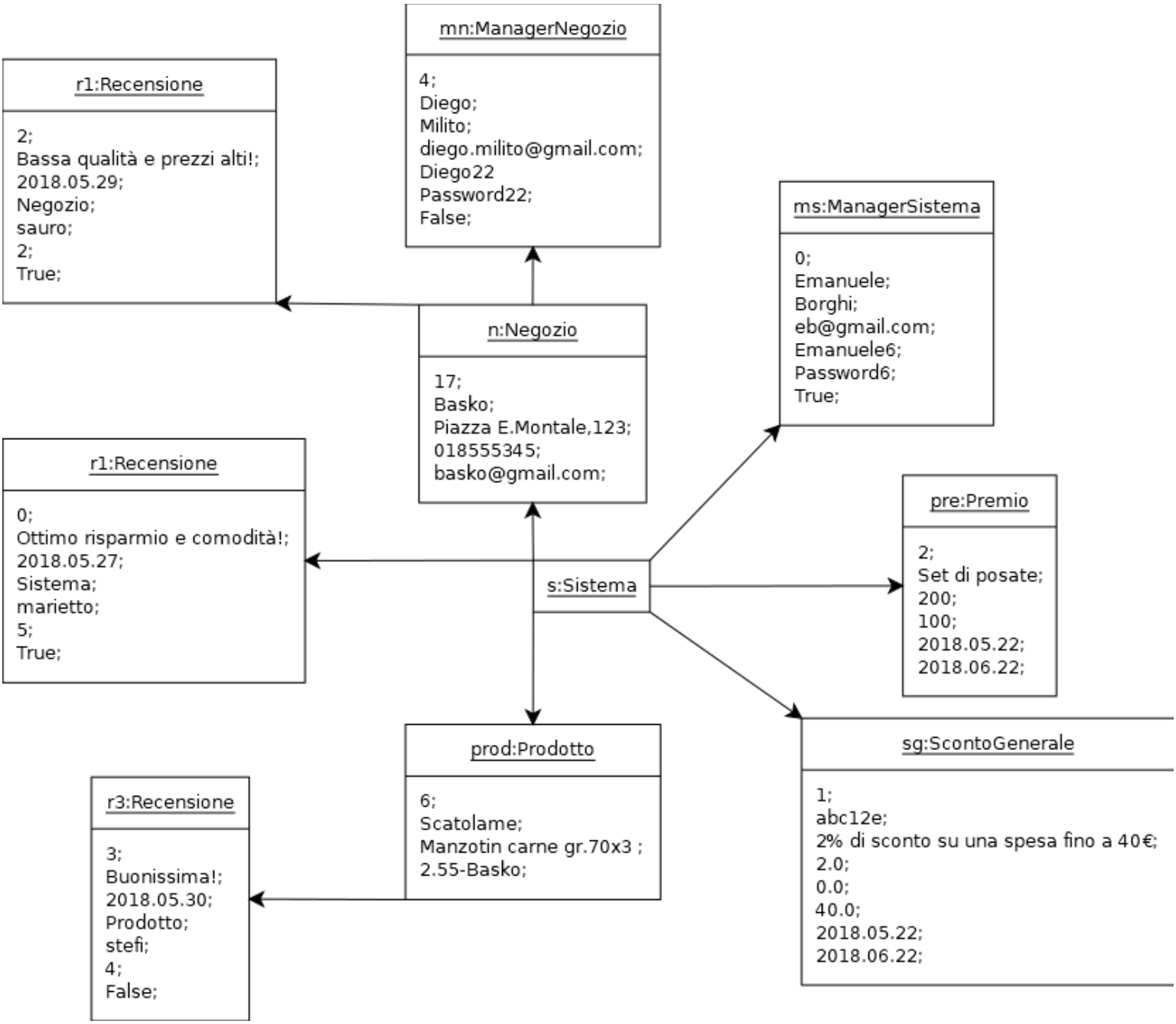


.

3.1.2 Class Diagram



3.1.3 Object Diagram



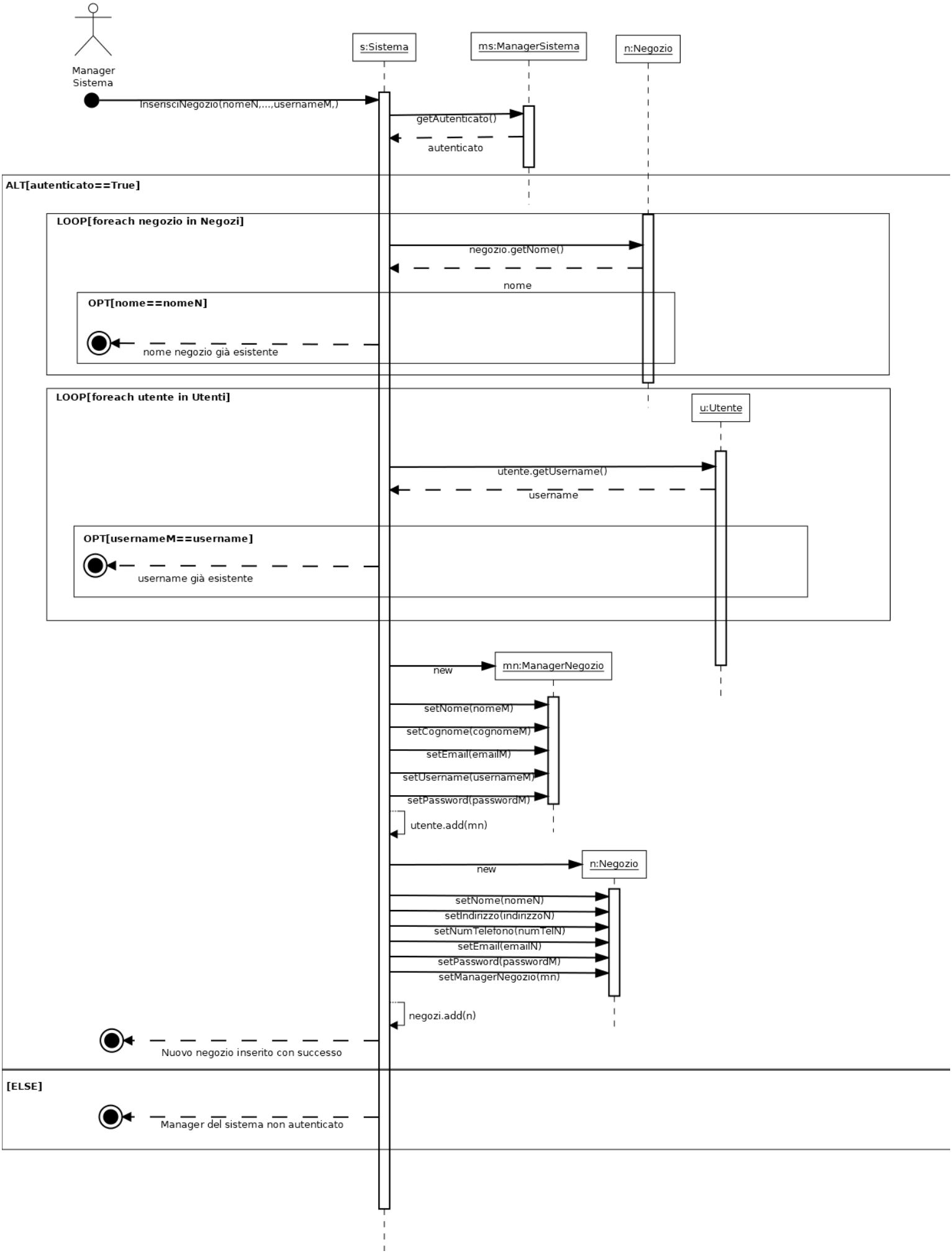
3.2 Dynamic Models

3.2.1 Sequence Diagrams

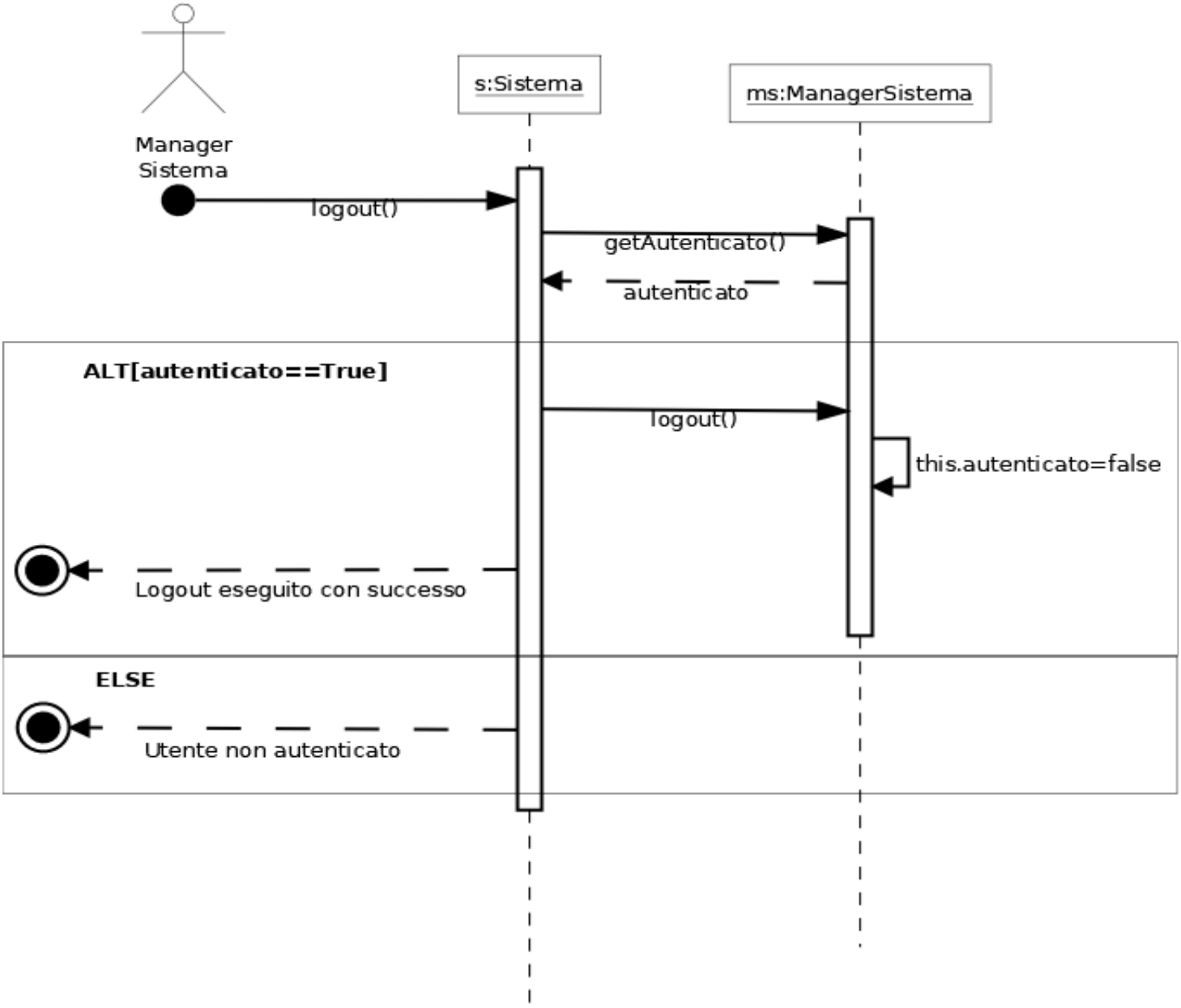
-UC: Inserire un nuovo negozio nel sistema

Ho considerato, in accordo con il document srs, che il Manager del negozio deve gestire un solo negozio e che un Negozio deve essere gestito da un solo manager.

La funzione iniziale è:
InserisciNegozio(nomeN,indirizzoN,numeTelN,emailN,nomeM,cognomeM,emailM,usernameM,passwordM)

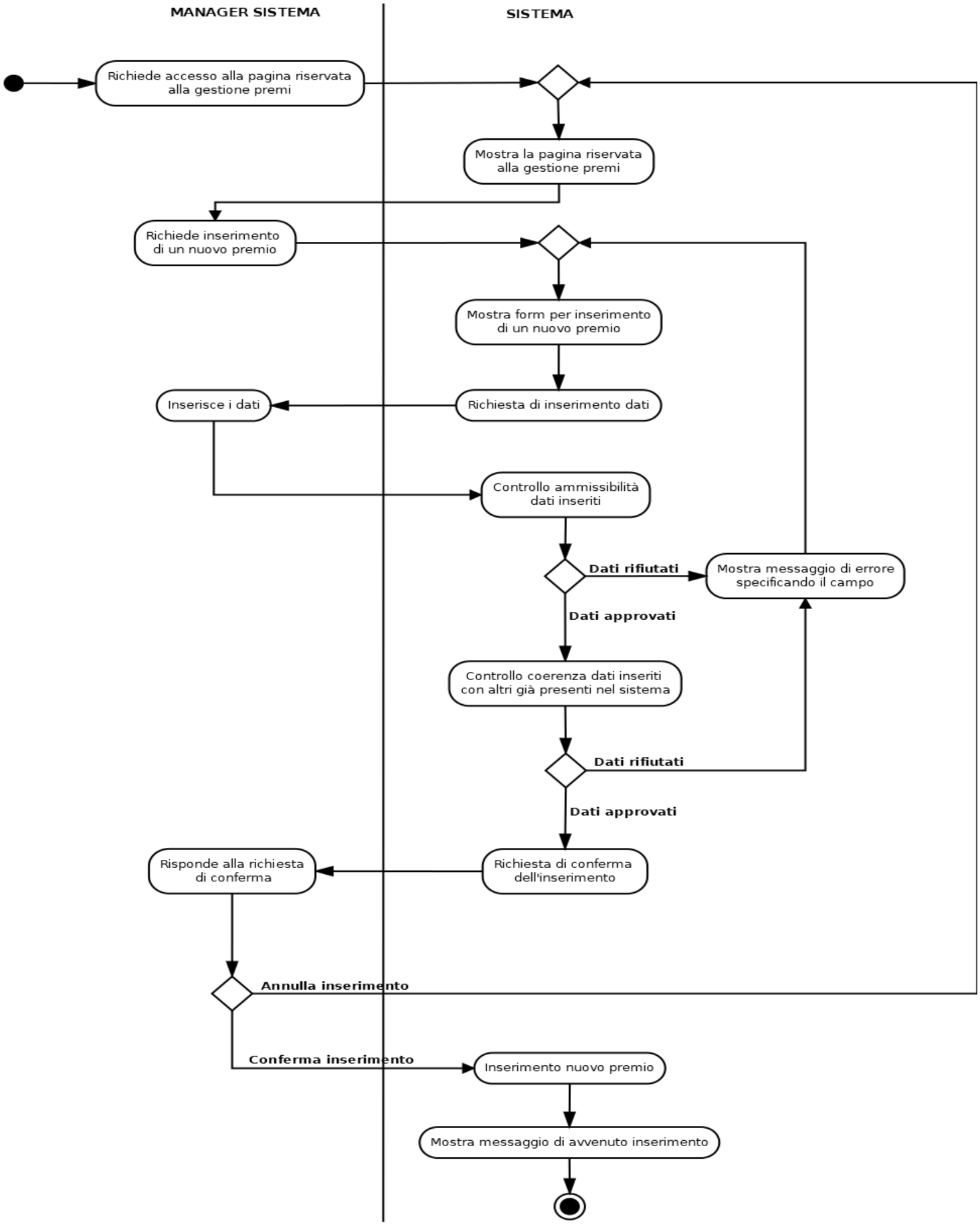

Manager Sistema
s:Sistema
ms:ManagerSistema
n:Negozio
InserisciNegozio(nomeN,...,usernameM,)
getAutenticato()
autenticato
ALT[autenticato==True]
LOOP[foreach negozio in Negozi]
negozio.getNome()
nome
OPT[nome==nomeN]
nome negozio già esistente
LOOP[foreach utente in Utenti]
u:Utente
utente.getUsername()
username
OPT[usernameM==username]
username già esistente
new
mn:ManagerNegozio
setNome(nomeM)
setCognome(cognomeM)
setEmail(emailM)
setUsername(usernameM)
setPassword(passwordM)
utente.add(mn)
new
n:Negozio
setNome(nomeN)
setIndirizzo(indirizzoN)
setNumTelefono(numTelN)
setEmail(emailN)
setPassword(passwordM)
setManagerNegozio(mn)
negozi.add(n)
Nuovo negozio inserito con successo
[ELSE]
Manager del sistema non autenticato

-UC: Effettuare logout



3.2.2 Activity Diagrams

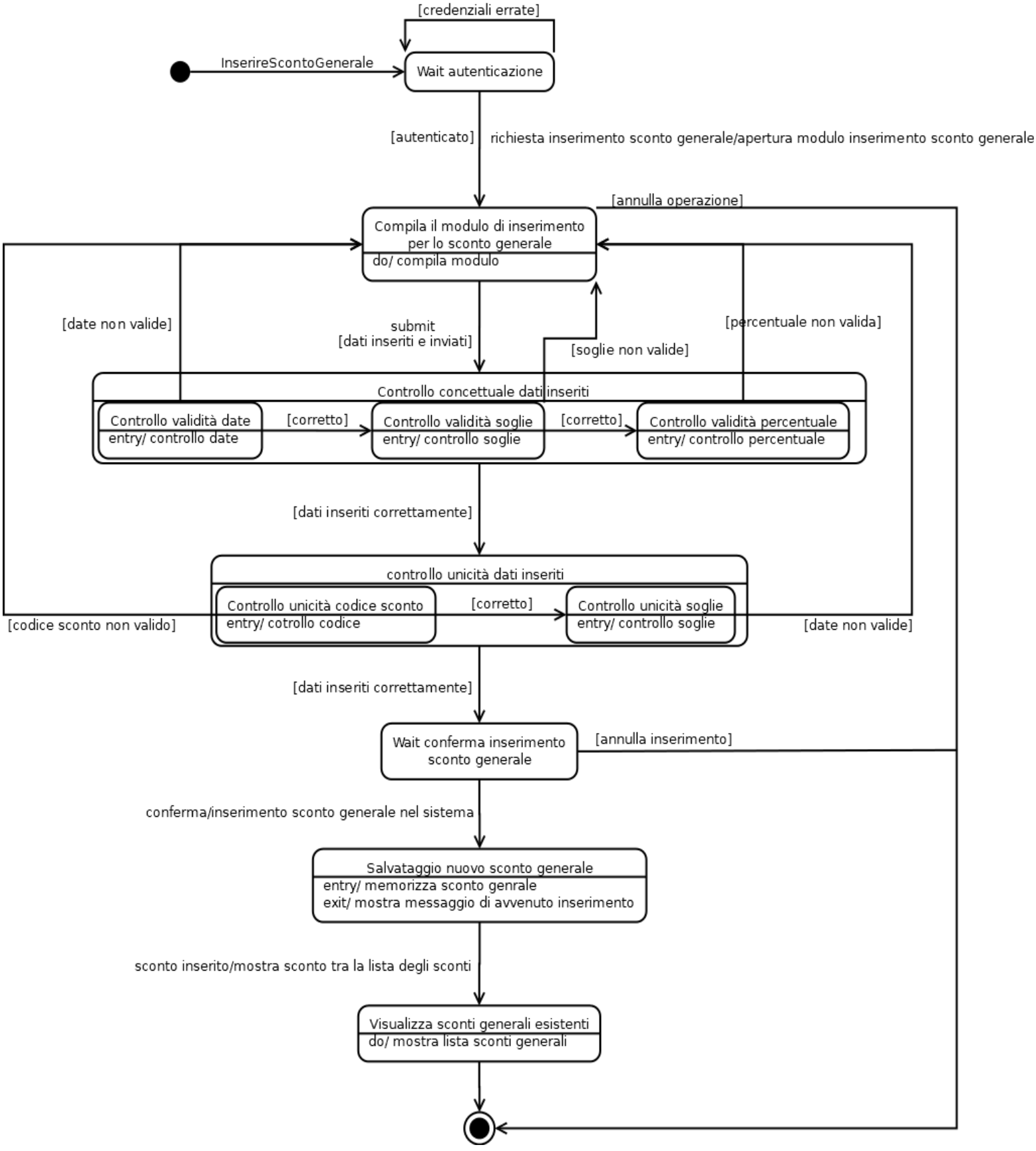
-UC: Aggiungere premio



-UC: Approvare recensioni

3.2.3 State Machine Diagrams

-UC: Inserimento sconto generale



4.0 Vincoli

1. “*Il Manager del sistema deve essere autenticato prima di poter svolgere l’operazione inserisciNegozio(…)*”

   Context Sistema :: inserisciNegozio(…):

   pre : self.gestore.autenticato = True

   IL VINCOLO DEVE ESSERE RISPETTATO ANCHE PER LE ALTRE OPERAZIONI (ricercaNegozio(…), modificaNegozio(…), eliminaNegozio(…), inserisciScontoGenerale(…), eliminaScontoGenerale(…), inserisciPremio(…), eliminaPremio(…), approvaRecensioni(…)).

2. “*Il Manager del sistema deve essere autenticato prima di poter svolgere l’operazione di logout() e deve essere deautenticato dopo*”

   Context Sistema :: logout(…):

   pre : self.managerSistema.autenticato = True

   post : self.managerSistema.autenticato = False

3. *“Il campo nome della classe Negozio deve essere unico in tutto il sistema”*

   Context Negozio:
   inv : Negozio : AllIstances() → forAll (s1, s2 and s1 != s2 | s1.nome != s2.nome)

4. *“Il campo codice sconto della classe ScontoGenerale deve essere unico in tutto il sistema”*

   Context ScontoGenerale:
   inv : ScontoGenerale: AllIstances() → forAll (s1, s2 and s1 != s2 | s1.codiceSconto != s2.codiceSconto)

5. *“Le soglie minime e massime della classe ScontoGenerale devono essere comprese tra 0 e 1000 poichè è permessa una spesa tra 0€ e 1000€”*

   Context ScontoGenerale:
   inv : ScontoGenerale: self.sogliaMin>=0 and self.sogliaMax<=1000

6. *“Ogni Negozio ha esattamente un Manager”*

Context Negozio:
inv : Negozio: self.mangerNegozio-> size()= 1

7. *“Il campo descrizione della classe Premio deve essere unico in tutto il sistema”*

Context Premio:
inv : Premio: AllIstances() → forAll (s1, s2 and s1 != s2 | s1.descrizione != s2.descrizione)

8. *“Il Sistema deve avere un Manager”*

Context Sistema:
inv : self.managerSistema !=Null()
inv : self.managerSistema -> size()=1