UNIVERSITÀ DEGLI STUDI DI PALERMO

FACOLTA' DI INGEGNERIA

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

DIPARTIMENTO DI INFORMATICA

A.A. 2003 – 2004

Tesina di ingegneria del software

Sistema multiagente per la gestione di un'azienda produttrice di biciclette

Docenti:
Massimo Cossentino
Umberto Lo Faso

Allievi ingegneri:
Giovanni Reina
Teodoro Ricciardello
Carmelo Scozzola

INDICE SINTETICO

INDICE

Software Project Management Plan (SPMP)

1 Software Project Management Plan (SPMP)

1.1 Prefazione

Questo documento esplicita il processo tecnico e la pianificazione della progettazione del sistema per la gestione di una azienda produttrice di biciclette. Esso è un documento di accompagnamento al Requirements Analysis Document (RAD); è rivolto agli sviluppatori, analisti e al cliente.

1.2 Introduzione

SuperBike Software è un software che si occupa della gestione di un'azienda produttrice di biciclette. L'obiettivo che ci si prefigge è quello di fornire agli impiegati di questa azienda dei mezzi semplici ed intuitivi per facilitare il controllo del flusso di informazioni.

1.2.1 Overview del progetto

La seguente tabella indica le attività principali che si devono affrontare al fine di poter progettare e realizzare il sistema proposto.

Schedulazione del progetto		
Data	**Fase di progetto**	**Attività**
10/10/2003→25/11/2003	Attività di studio: • Microsoft Project • Jade • Microsoft Access e SQL • Rational rose • Agent Factory • Passi	
25/11/2003		Riunione di coordinamento
26/11/2003→11/12/2003	Analisi: • Analisi del problema • Analisi dei requisiti • Descrizione scenari testuali • Requirements Analysis Document (RAD)	

12/12/2003		Revisione con il committente
15/12/2003→17/12/2003	• Modifiche RAD • Schedulazione del progetto	
17/12/2003		Presentazione del progetto al cliente
18/12/2003→28/01/2004	Progettazione software suddivisa nelle seguenti macro – fasi: • System requirements model • Agent society model • Agent implementation model • Deployment configuration • Progettazione database	
30/12/2003		Discussione coi membri del progetto dei risultati ottenuti dopo la fase Agent Identification contenuta nella fase System Requirement Model
13/01/2004		Discussione coi membri del progetto dei risultati ottenuti dopo la fase Domain Ontology Description contenuta nella fase Agent Society Model
21/01/2004		Discussione coi membri del progetto dei risultati parziali
29/01/2004		Riunione di fine progetto
30/01/2004→01/03/2004	Implementazione: • Ripartizione compiti di implementazione • Implementazione GUI • Implementazione agenti • Implementazione database • Assemblaggio delle componenti	

01/03/2004		Presentazione prima versione al cliente
02/03/2004→13/04/2004	Fase di testing suddivisa in: • Unit testing • Integration testing • System testing	
13/04/2004		Revisione finale
14/04/2004		Consegna del prodotto al cliente

1.2.2 Documenti prodotti

Si elenca in seguito la documentazione del progetto che sarà prodotta:

- **Documento del piano di sviluppo del progetto (SPMP):** definisce il processo tecnico e la pianificazione necessaria allo sviluppo del sistema (questo documento).
- **Documento di analisi dei requisiti (RAD):** descrive le funzionalità del sistema da realizzare. Questo documento è creato interagendo con gli esperti del dominio ed è approvato dal cliente.
- **Documento PASSI Project:** raccoglie le varie fasi della progettazione del sistema secondo la metodologia PASSI.
- **Progetto DataBase:** include il progetto concettuale e logico del database utilizzato;
- **Pacchetto di installazione del software:** sarà fornito un CD contentente tutto il necessario per l'installazione ed il corretto funzionamento del software nelle macchine dell'utente (Jade 3.0b1, JRE 1.4, Rdf codec, DataBase, Files esegiubili e codice sorgente).

1.2.3 Evoluzione del piano di sviluppo del progetto

Per poter pianificare correttamente ed effettuare in modo semplice cambiamenti nella pianificazione, per la gestione ed allocazione delle risorse necessarie alla realizzazione del prodotto e per la gestione delle previsioni si è utilizzato il tool Microsoft Project 2002 Professional.

Per la progettazione del sistema ad agenti si è utilizzata PASSI, una metodologia per sviluppare software multi-agente utilizzando l'UML.

Il prodotto finale e le sue versioni intermedie sono state testate su PC che utilizzano sistema operativo Windows XP Home Edition e Professional Edition, Linux Mandrake 9.2.

1.2.4 Materiale di riferimento

Per la realizzazione del sistema sono stati utilizzati i seguenti materiali di riferimento:

- Bruegge-Dutoit :*Object-Oriented Software Engineering: Conquering Complex and Changing System* (Corso di Ingegneria del Software)
- Tutto il materiale di esempio trovato sul sito di Ingegneria del Software del professore M. Cossentino. Tale materiale può essere recuperato collegandosi al sito internet http://www.csai.unipa.it/cossentino/se02_03/ e sfruttando i collegamenti ivi presenti
- Tutto il materiale riguardernte il "Project Management" fornito sul sito: http://www.pa.icar.cnr.it/~cossentino/project/
- Documentazione JADE 3.0b1 scaricabile dal sito http://jade.cselt.it/
- Deitel & Deitel*: Java Fondamenti di programmazione* (Manuale di java)
- Deitel & Deitel*: Java Tecniche avanzate di programmazione* (Manuale di java)
- Documentazione del J2SDK1.4.0
- Guida in linea di Rational Rose Enterprise Edition
- Guida in linea di Microsoft Project Professional 2002
- Guida in linea di Microsoft Access XP
- Guida in linea di MySql
- Guida in linea di Microsoft Visio Professional 2002

1.2.5 Acronimi e definizioni

AIP – Agent Identification Phase

COD – Communication Ontology Description

DBMS- Data Base Management System

DCP – Deployment Configuration Phase

DDP – Domain Description Phase

DOD – Domain Ontology Description

GUI - Graphical User Interface

JDK - Java Development Kit

JDBC - Java DataBase Connectivity

JVM - Java Virtual Machine

MASD – Multi Agent Structure Definition

MABD – Multi Agent Behaviour Description

PASSI – Project for Agent Societies Specification and Implementation

PTK – Passi Tool Kit

ODP – Ontology Description Phase

RDF – Resource Description Language

RDP – Role Description Phase

RAD - Requirements Analysis Document

RIP – Role Identification Phase

ROSE – Tool di progettazione e sviluppo software

SPMP - Software Project Management Plan

SQL – Structured Query Language

SASD – Single Agent Structure Definition

TSP – Task Specification Phase

UML - Unified Modeling Notation

XML – eXtensible Markup Language

1.3 Organizzazione del progetto

1.3.1 Schema del processo

Il progetto è iniziato il 10 ottobre 2003 ed è terminato il 14 aprile 2004. Durante tale periodo sono state svolte diverse attività. Presentiamo di seguito quelle principali che terminano (in genere) con la creazione di versioni intermedie del prodotto. Tali versioni intermedie (eccetto il prodotto finale stesso che si reputa essere completo e funzionante) servono per mostrare il sistema al cliente. Tali attività sono di seguito riportate.

- riassunto e presentazione risultati dello studio preliminare: 25 novembre 2003;
- presentazione del progetto globale e approvazione del cliente: 17 dicembre 2003;
- presentazione progetto sistema al cliente: 30 dicembre 2003;
- presentazione e dimostrazione prima versione del software al cliente: 1 marzo 2004;
- consegna del prodotto: 14 aprile 2004.

Riportiamo di seguito la descrizione delle varie attività e fasi di lavoro che si sono affrontate per poter realizzare il sistema software progettato.

1.3.1.1 Attività di studio

Questa fase è stata dedicata allo studio di fattibiltà del progetto e porta ad una prima descrizione formale. Si è provveduto poi al consolidamento delle conoscenze del linguaggio Java, dei tools Microsoft Project e Access, del Microsoft Visio, del tool di progettazione sistemi ad agenti Passi Tool Kit, del software AgentFactory e all'apprendimento dell'uso del tool Rational Rose , della piattaforma Jade e del linguaggio SQL.

Questa prima fase scaturisce nella stesura di questo documento.

1.3.1.2 Analisi dei requisiti

Durante questa attività è stato analizzato il problema ed è stato esaminato il sistema in termini delle funzionalità da fornire cercando di mantenere le proprietà di consistenza e di completezza.
Questa attività porta alla stesura degli scenari testuali che descrivono il sistema che si vuole realizzare e che il cliente desidera avere.
In seguito è stato prodotto il documento RAD (Requirements Analysis Document) nel quale sono stati indiviuati:

- requisiti funzionali: descrivono le interazioni tra il sistema e l'ambiente indipendentemente dall'implementazione. L'ambiente include l'utente ed ogni altra attività esterna al sistema con la quale interagisce.
- requisiti non funzionali: decsrivono gli aspetti del sistema visibili all'utente che non sono in relazione diretta con i requisiti funzionali del sistema (per esempio interfacce utente, prestazioni, etc);
- pseudorequisiti: sono requisiti imposti dal cliente che vincolano l'implementazione del sistema (per esempio il linguaggio di implementazione e piattaforma del sistema).

Inoltre è stata messa a punto una prima schedulazione delle attività da svolgere per avere un'idea del tempo e dei costi necessari.

1.3.1.3 Presentazione specifiche al cliente

Presentazione del progetto al cliente e approvazione delle funzionalità del sistema presentato tramite il documento RAD.

1.3.1.4 Progettazione software

Durante questa attività sono state svolte le varie fasi di progettazione previste da PASSI:

- System Requirements Model: descrive i requisiti del sistema, le funzionalità degli agenti, i ruoli giocati da ogni agente nel portare a termine i propri compiti e i comportamenti;
- Agent Society Model: include la descrizione dell'ontologia del sistema e delle comunicazioni tra gli agenti, dei ruoli e dei protocolli;
- Agent Implementation Model: include la descrizione della struttura e dei comportamenti di ogni singolo agente e della società degli agenti;

Inoltre è stato effettuato il progetto concettuale e logico del Database utilizzato.

1.3.1.5 Implementazione

Durante questa fase è stata effettuata la codifica delle interfacce e dei singoli agenti. È stata anche effettuata l'implementazione del Database.

1.3.1.6 Presentazione prima versione al cliente

Presentazione di una prima versione al cliente ed annotazione delle modifiche da apportare.

1.3.1.7 Testing

Durante questa attività è stato testato il sistema nel tentativo di rilevare il maggior numero di errori possibile e sono state effettuate le correzioni. L'attività di testing è suddivisa in :

- Unit testing: testing dei singoli sottosistemi, nello specifico, testing dei singoli agenti
- Integration testing: testing di integrazione dei sottosistemi, effettuato facendo interagire tra di loro gruppi di agenti
- System testing: testing globale del sistema realizzato

1.3.1.8 Fase finale del progetto

Durante questa attività è stata prodotta la documentazione mancante ed è stata rivista quella esistente.

1.3.1.9 Consegna prodotto al cliente

In questa fase è stato presentato il prodotto al cliente ed è stata fatta una dimostrazione del prodotto. Tale fase è terminata con la consegna del prodotto finale.

1.3.2 Risorse umane coinvolte nel progetto

Il progetto è stato affidato ad un singolo team composto dalle seguenti figure professionali:

- **Capo progetto:** Coordina il progetto e lo pianifica secondo le esigenze ed i tempi del committente. Coordina anche gli incontri con il cliente e gli incontri fra i membri del team.
- **Analista:** Si occupa di fornire un'adeguata analisi del problema insieme a degli esperti del dominio e cerca di tradurre in specifiche tecniche le richieste del cliente.
- **Progettista software:** Ha il compito di studiare il sistema ed effettuarne la decomposizione in sottosistemi.
- **Amministratore Jade:** Si occupa della gestione della piattaforma Jade.
- **Programmatore Java:** Ha il compito di sviluppare i componenti del sistema.
- **Programmatore GUI:** Ha il compito di sviluppare le interfacce grafiche del sistema.

- **Progettista Database:** Ha il compito di studiare il sistema ed effettuarne la progettazione del database.
- **Tester:** Ha il compito di testare il sistema e di comunicare ai programmatori eventuali errori trovati.

1.4 Strumenti utilizzati

Di seguito sono riportati gli strumenti utilizzati nella progettazione e sviluppo del prodotto:

- JADE 3.0b1;
- Rational Rose 2001 Enterprise Edition;
- Microsoft Access XP;
- Microsoft Project 2002 Professional;
- Microsoft Visio 2002 Professional;
- Microsoft Word XP;
- J2SDK 1.4.2;
- Agent Factory standalone version 1.2;
- PTK 1.2.1 PASSI Tool Kit;
- NetBeans IDE 3.5.1;
- Eclipse 3.0

1.5 Vista complessiva sulla pianificazione del progetto

Riportiamo tutte le informazioni di pianificazione del progetto. Tale attività di pianificazione è stata realizzata utilizzando gli strumenti messi a disposizione dal tool Microsoft Project 2003.

1.5.1 Diagrammi di pianificazione progetto

1.5.1.1 Risorse del progetto

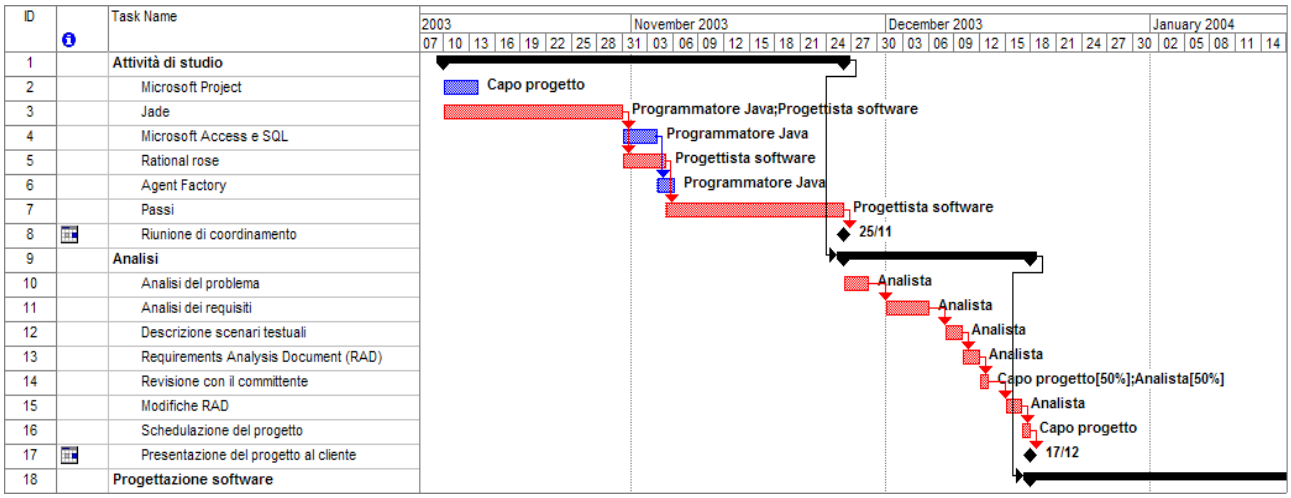
Mostriamo adesso le risorse che vengono utilizzate nel nostro progetto e gli attributi più importanti di esse.

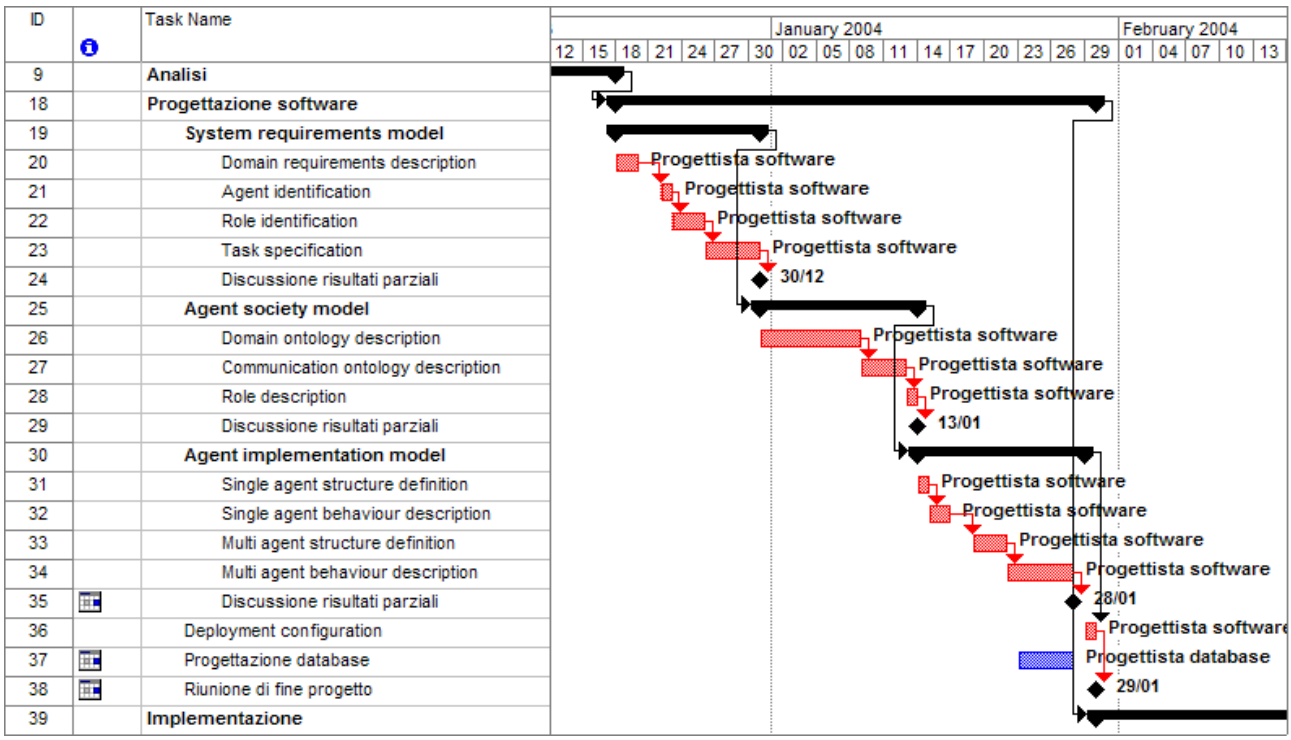
ID	Resource Name	Type	Initials	Max. Units	Std. Rate	Ovt. Rate
1	Capo progetto	Work	C	100%	€ 250,00/day	€ 40,00/hr
2	Analista	Work	A	100%	€ 140,00/day	€ 25,00/hr
3	Programmatore Java	Work	PJ	100%	€ 110,00/day	€ 20,00/hr
4	Programmatore GUI	Work	PG	100%	€ 110,00/day	€ 20,00/hr
5	Tester	Work	T	100%	€ 70,00/day	€ 15,00/hr
6	Progettista software	Work	PS	100%	€ 140,00/day	€ 25,00/hr
7	Progettista database	Work	PD	100%	€ 120,00/day	€ 25,00/hr
8	Amministratore Jade	Work	AJ	100%	€ 120,00/day	€ 25,00/hr

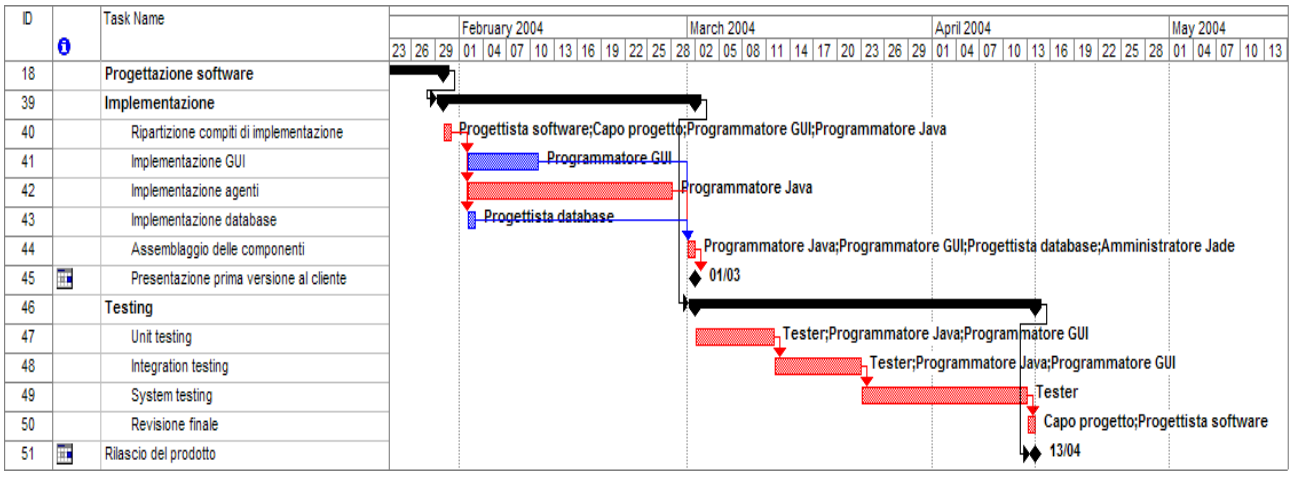
1.5.1.2 Attività del progetto

ID		Task Name	Work	Duration	Start	Finish	Resource Names
1		**Attività di studio**	**432 hrs**	**33 days**	**Fri 10/10/03**	**Tue 25/11/03**	
2		Microsoft Project	16 hrs	2 days	Fri 10/10/03	Mon 13/10/03	Capo progetto
3		Jade	240 hrs	15 days	Fri 10/10/03	Thu 30/10/03	Programmatore Java;Progettista software
4		Microsoft Access e SQL	16 hrs	2 days	Fri 31/10/03	Mon 03/11/03	Programmatore Java
5		Rational rose	24 hrs	3 days	Fri 31/10/03	Tue 04/11/03	Progettista software
6		Agent Factory	16 hrs	2 days	Tue 04/11/03	Wed 05/11/03	Programmatore Java
7		Passi	120 hrs	15 days	Wed 05/11/03	Tue 25/11/03	Progettista software
8		Riunione di coordinamento	0 hrs	0 days	Tue 25/11/03	Tue 25/11/03	Capo progetto;Analista;Progettista software
9		**Analisi**	**128 hrs**	**16 days?**	**Wed 26/11/03**	**Wed 17/12/03**	
10		Analisi del problema	24 hrs	3 days	Wed 26/11/03	Fri 28/11/03	Analista
11		Analisi dei requisiti	40 hrs	5 days	Mon 01/12/03	Fri 05/12/03	Analista
12		Descrizione scenari testuali	16 hrs	2 days	Mon 08/12/03	Tue 09/12/03	Analista
13		Requirements Analysis Document (RAD)	16 hrs	2 days	Wed 10/12/03	Thu 11/12/03	Analista
14		Revisione con il committente	8 hrs	1 day?	Fri 12/12/03	Fri 12/12/03	Capo progetto[50%];Analista[50%]
15		Modifiche RAD	16 hrs	2 days	Mon 15/12/03	Tue 16/12/03	Analista
16		Schedulazione del progetto	8 hrs	1 day?	Wed 17/12/03	Wed 17/12/03	Capo progetto
17		Presentazione del progetto al cliente	0 hrs	0 days	Wed 17/12/03	Wed 17/12/03	Capo progetto
18		**Progettazione software**	**224 hrs**	**31 days?**	**Thu 18/12/03**	**Thu 29/01/04**	
19		**System requirements model**	**72 hrs**	**9 days?**	**Thu 18/12/03**	**Tue 30/12/03**	
20		Domain requirements description	16 hrs	2 days	Thu 18/12/03	Fri 19/12/03	Progettista software
21		Agent identification	8 hrs	1 day?	Mon 22/12/03	Mon 22/12/03	Progettista software
22		Role identification	24 hrs	3 days	Tue 23/12/03	Thu 25/12/03	Progettista software
23		Task specification	24 hrs	3 days	Fri 26/12/03	Tue 30/12/03	Progettista software
24		Discussione risultati parziali	0 hrs	0 days	Tue 30/12/03	Tue 30/12/03	Progettista software;Capo progetto;Analista
25		**Agent society model**	**80 hrs**	**10 days?**	**Wed 31/12/03**	**Tue 13/01/04**	
26		Domain ontology description	56 hrs	7 days	Wed 31/12/03	Thu 08/01/04	Progettista software
27		Communication ontology description	16 hrs	2 days	Fri 09/01/04	Mon 12/01/04	Progettista software
28		Role description	8 hrs	1 day?	Tue 13/01/04	Tue 13/01/04	Progettista software
29		Discussione risultati parziali	0 hrs	0 days	Tue 13/01/04	Tue 13/01/04	Capo progetto;Progettista software
30		**Agent implementation model**	**48 hrs**	**6 days?**	**Wed 14/01/04**	**Wed 21/01/04**	
31		Single agent structure definition	8 hrs	1 day	Wed 14/01/04	Wed 14/01/04	Progettista software
32		Single agent behaviour description	8 hrs	1 day?	Thu 15/01/04	Thu 15/01/04	Progettista software
33		Multi agent structure definition	8 hrs	1 day?	Fri 16/01/04	Fri 16/01/04	Progettista software
34		Multi agent behaviour description	24 hrs	3 days	Mon 19/01/04	Wed 21/01/04	Progettista software
35		Discussione risultati parziali	0 hrs	0 days	Wed 21/01/04	Wed 21/01/04	Capo progetto;Progettista software
36		Deployment configuration	8 hrs	1 day?	Thu 22/01/04	Thu 22/01/04	Progettista software
37		Progettazione database	16 hrs	2 days	Wed 14/01/04	Thu 15/01/04	Progettista database
38		Riunione di fine progetto	0 hrs	0 days	Thu 29/01/04	Thu 29/01/04	getto;Progettista software;Progettista database
39		**Implementazione**	**288 hrs**	**22 days?**	**Fri 30/01/04**	**Mon 01/03/04**	
40		Ripartizione compiti di implementazione	32 hrs	1 day?	Fri 30/01/04	Fri 30/01/04	ogetto;Programmatore GUI;Programmatore Java
41		Implementazione GUI	56 hrs	7 days	Mon 02/02/04	Tue 10/02/04	Programmatore GUI
42		Implementazione agenti	160 hrs	20 days	Mon 02/02/04	Fri 27/02/04	Programmatore Java
43		Implementazione database	8 hrs	1 day?	Mon 02/02/04	Mon 02/02/04	Progettista database
44		Assemblaggio delle componenti	32 hrs	1 day?	Mon 01/03/04	Mon 01/03/04	e GUI;Progettista database;Amministratore Jade
45		Presentazione prima versione al cliente	0 hrs	0 days	Mon 01/03/04	Mon 01/03/04	Capo progetto
46		**Testing**	**496 hrs**	**31 days?**	**Tue 02/03/04**	**Tue 13/04/04**	
47		Unit testing	192 hrs	8 days	Tue 02/03/04	Thu 11/03/04	Tester;Programmatore Java;Programmatore GUI
48		Integration testing	168 hrs	7 days	Fri 12/03/04	Mon 22/03/04	Tester;Programmatore Java;Programmatore GUI
49		System testing	120 hrs	15 days	Tue 23/03/04	Mon 12/04/04	Tester
50		Revisione finale	16 hrs	1 day?	Tue 13/04/04	Tue 13/04/04	Capo progetto;Progettista software
51		Rilascio del prodotto	0 hrs	0 days	Tue 13/04/04	Tue 13/04/04	

1.5.1.3 Diagramma di Gantt







1.5.1.4 Diagramma di utilizzo delle risorse

ID	ℹ	Resource Name	Details																												
																2004															
								November																							
				6/10	13/10	20/10	27/1	03/11	10/11	17/11	24/11	01/12	08/12	15/12	22/12	29/12	05/01	12/01	19/01	26/01	02/02	09/02	16/02	23/02	01/03	08/03	15/03	22/03	29/03	05/04	12/04
1		Capo progetto	Work	8h	8h						0h		4h	8h		0h		0h	0h	8h					0h						8h
		Microsoft Project	Work	8h	8h																										
		Riunione di coordinamento	Work								0h																				
		Revisione con il committente	Work										4h																		
		Schedulazione del progetto	Work											8h																	
		Presentazione del progetto al cliente	Work											0h																	
		Discussione risultati parziali	Work													0h															
		Discussione risultati parziali	Work															0h													
		Discussione risultati parziali	Work																0h												
		Riunione di fine progetto	Work																	0h											
		Ripartizione compiti di implementazion	Work																	8h											
		Presentazione prima versione al cliente	Work																						0h						
		Revisione finale	Work																												8h
2		Analista	Work								24h	40h	36h	16h		0h															
		Riunione di coordinamento	Work								0h																				
		Analisi del problema	Work								24h																				
		Analisi dei requisiti	Work									40h																			
		Descrizione scenari testuali	Work										16h																		
		Requirements Analysis Document (RAL	Work										16h																		
		Revisione con il committente	Work										4h																		
		Modifiche RAD	Work											16h																	
		Discussione risultati parziali	Work													0h															
3		Programmatore Java	Work	8h	40h	40h	40h	24h												8h	40h	40h	40h	40h	40h	40h	40h	8h			
		Jade	Work	8h	40h	40h	32h																								
		Microsoft Access e SQL	Work				8h	8h																							
		Agent Factory	Work					16h																							
		Ripartizione compiti di implementazion	Work																	8h											
		Implementazione agenti	Work																		40h	40h	40h	40h							
		Assemblaggio delle componenti	Work																						8h						
		Unit testing	Work																						32h	32h					
		Integration testing	Work																							8h	40h	8h			

ID	ℹ	Resource Name	Type	Details													2004															
								November																								
					06/10	13/10	20/10	27/1	03/11	10/11	17/11	24/11	01/12	08/12	15/12	22/12	29/12	05/01	12/01	19/01	26/01	02/02	09/02	16/02	23/02	01/03	08/03	15/03	22/03	29/03	05/04	12/04
4		Programmatore GUI	Work	Work																	8h	40h	16h			40h	40h	40h	8h			
		Ripartizione com	*Work*	Work																	8h											
		Implementazione	*Work*	Work																		40h	16h									
		Assemblaggio de	*Work*	Work																						8h						
		Unit testing	*Work*	Work																						32h	32h					
		Integration testin	*Work*	Work																							8h	40h	8h			
5		Tester	Work	Work																						32h	40h	40h	40h	40h	40h	8h
		Unit testing	*Work*	Work																						32h	32h					
		Integration testin	*Work*	Work																							8h	40h	8h			
		System testing	*Work*	Work																									32h	40h	40h	8h
6		Progettista software	Work	Work	8h	40h	40h	40h	40h	40h	40h	16h			16h	40h	40h	40h	40h	32h	8h											8h
		Jade	*Work*	Work	8h	40h	40h	32h																								
		Rational rose	*Work*	Work				8h	16h																							
		Passi	*Work*	Work					24h	40h	40h	16h																				
		Riunione di coorc	*Work*	Work								0h																				
		Domain requirem	*Work*	Work											16h																	
		Agent identificati	*Work*	Work												8h																
		Role identificatio	*Work*	Work												24h																
		Task specificatio	*Work*	Work												8h	16h															
		Discussione risu	*Work*	Work													0h															
		Domain ontology	*Work*	Work													24h	32h														
		Communication c	*Work*	Work														8h	8h													
		Role description	*Work*	Work															8h													
		Discussione risu	*Work*	Work															0h													
		Single agent stru	*Work*	Work															8h													
		Single agent beh	*Work*	Work															8h													
		Multi agent struct	*Work*	Work															8h													
		Multi agent beha	*Work*	Work																24h												
		Discussione risu	*Work*	Work																0h												
		Deployment confi	*Work*	Work																8h												
		Riunione di fine p	*Work*	Work																	0h											
		Ripartizione com	*Work*	Work																	8h											
		Revisione finale	*Work*	Work																												8h
7		Progettista database	Work	Work															16h		0h	8h				8h						
		Progettazione da	*Work*	Work															16h													
		Riunione di fine p	*Work*	Work																	0h											
		Implementazione	*Work*	Work																		8h										
		Assemblaggio de	*Work*	Work																						8h						
8		Amministratore Jade	Work	Work																						8h						
		Assemblaggio de	*Work*	Work																						8h						

1.5.1.5 Costi del progetto

ID	Resource Name	Work	Cost
1	Capo progetto	44 hrs	€ 1.375,00
	Microsoft Project	*16 hrs*	*€ 500,00*
	Riunione di coordinamento	*0 hrs*	*€ 0,00*
	Revisione con il committente	*4 hrs*	*€ 125,00*
	Schedulazione del progetto	*8 hrs*	*€ 250,00*
	Presentazione del progetto al cliente	*0 hrs*	*€ 0,00*
	Discussione risultati parziali	*0 hrs*	*€ 0,00*
	Discussione risultati parziali	*0 hrs*	*€ 0,00*
	Discussione risultati parziali	*0 hrs*	*€ 0,00*
	Riunione di fine progetto	*0 hrs*	*€ 0,00*
	Ripartizione compiti di implementazione	*8 hrs*	*€ 250,00*
	Presentazione prima versione al cliente	*0 hrs*	*€ 0,00*
	Revisione finale	*8 hrs*	*€ 250,00*
2	Analista	116 hrs	€ 2.030,00
	Riunione di coordinamento	*0 hrs*	*€ 0,00*
	Analisi del problema	*24 hrs*	*€ 420,00*
	Analisi dei requisiti	*40 hrs*	*€ 700,00*
	Descrizione scenari testuali	*16 hrs*	*€ 280,00*
	Requirements Analysis Document (RAD)	*16 hrs*	*€ 280,00*
	Revisione con il committente	*4 hrs*	*€ 70,00*
	Modifiche RAD	*16 hrs*	*€ 280,00*
	Discussione risultati parziali	*0 hrs*	*€ 0,00*
3	Programmatore Java	448 hrs	€ 6.160,00
	Jade	*120 hrs*	*€ 1.650,00*
	Microsoft Access e SQL	*16 hrs*	*€ 220,00*
	Agent Factory	*16 hrs*	*€ 220,00*
	Ripartizione compiti di implementazione	*8 hrs*	*€ 110,00*
	Implementazione agenti	*160 hrs*	*€ 2.200,00*
	Assemblaggio delle componenti	*8 hrs*	*€ 110,00*
	Unit testing	*64 hrs*	*€ 880,00*
	Integration testing	*56 hrs*	*€ 770,00*
4	Programmatore GUI	192 hrs	€ 2.640,00
	Ripartizione compiti di implementazione	*8 hrs*	*€ 110,00*
	Implementazione GUI	*56 hrs*	*€ 770,00*
	Assemblaggio delle componenti	*8 hrs*	*€ 110,00*
	Unit testing	*64 hrs*	*€ 880,00*
	Integration testing	*56 hrs*	*€ 770,00*
5	Tester	240 hrs	€ 2.100,00
	Unit testing	*64 hrs*	*€ 560,00*
	Integration testing	*56 hrs*	*€ 490,00*
	System testing	*120 hrs*	*€ 1.050,00*
6	Progettista software	488 hrs	€ 8.540,00
	Jade	*120 hrs*	*€ 2.100,00*
	Rational rose	*24 hrs*	*€ 420,00*
	Passi	*120 hrs*	*€ 2.100,00*
	Riunione di coordinamento	*0 hrs*	*€ 0,00*
	Domain requirements description	*16 hrs*	*€ 280,00*
	Agent identification	*8 hrs*	*€ 140,00*
	Role identification	*24 hrs*	*€ 420,00*
	Task specification	*24 hrs*	*€ 420,00*
	Discussione risultati parziali	*0 hrs*	*€ 0,00*
	Domain ontology description	*56 hrs*	*€ 980,00*
	Communication ontology description	*16 hrs*	*€ 280,00*
	Role description	*8 hrs*	*€ 140,00*
	Discussione risultati parziali	*0 hrs*	*€ 0,00*
	Single agent structure definition	*8 hrs*	*€ 140,00*
	Single agent behaviour description	*8 hrs*	*€ 140,00*
	Multi agent structure definition	*8 hrs*	*€ 140,00*
	Multi agent behaviour description	*24 hrs*	*€ 420,00*
	Discussione risultati parziali	*0 hrs*	*€ 0,00*
	Deployment configuration	*8 hrs*	*€ 140,00*
	Riunione di fine progetto	*0 hrs*	*€ 0,00*
	Ripartizione compiti di implementazione	*8 hrs*	*€ 140,00*
	Revisione finale	*8 hrs*	*€ 140,00*
7	Progettista database	32 hrs	€ 480,00
	Progettazione database	*16 hrs*	*€ 240,00*
	Riunione di fine progetto	*0 hrs*	*€ 0,00*
	Implementazione database	*8 hrs*	*€ 120,00*
	Assemblaggio delle componenti	*8 hrs*	*€ 120,00*
8	Amministratore Jade	14 hrs	€ 120,00
	Assemblaggio delle componenti	*8 hrs*	*€ 120,00*

Il costo totale del software è di € 23.445,00.

Requirements Analysis Document (RAD)

2 Requirements Analysis Document (RAD)

2.1 Obiettivi generali

Lo scopo del sistema è integrare le diverse locazioni di una ditta produttrice di biciclette al fine di aumentarne l'efficienza. Il sistema di cui è richiesta la progettazione e implementazione si interfaccia con gli impiegati della azienda nei due settori principali: la produzione (nei due stabilimenti Nord e Sud) e la amministrazione. I due stabilimenti producano ognuno un tipo diverso di bicicletta: il modello A e il modello B.

Postazioni di lavoro previste presso l'amministrazione:

- addetto ufficio clienti
- addetto alla produzione
- responsabile settore produzione
- amministratore di sistema

Postazioni di lavoro previste presso ogni stabilimento:

- responsabile di stabilimento
- responsabile magazzino
- operaio addetto alla produzione.

Il sistema deve fornire un'interfaccia che permetta ai diversi dipendenti di espletare le proprie mansioni, in particolare:

- Raccolta degli ordini provenienti dalla clientela
- Smistamento degli ordini clienti
- Ricezione e schedulazione lotti di produzione
- Approvvigionamento materie prime
- Inventario e registrazione pezzi prodotti

Il sistema si prefigge di fornire alle diverse figure professionali della ditta gli strumenti necessari a svolgere le proprie mansioni, mantenendo la coerenza e la consistenza delle informazioni nell'intero sistema. Così facendo si perviene ad una gestione ordinata del flusso di informazioni con diverso grado di accessibilità alle medesime in base alle diverse figure professionali dei dipendenti.

Per concludere il sistema offre delle interfacce grafiche intuitive che rendono semplice l'utilizzo da parte dei dipendenti del sistema medesimo.

2.2 Sistema proposto

2.2.1 Overview del progetto

Il sistema si interfaccia con gli impiegati della azienda nei due settori principali: la produzione (nei due stabilimenti Nord e Sud) e la amministrazione. I due stabilimenti producono ognuno un tipo diverso di bicicletta: il modello A e il modello B.

Le attività tipiche in cui verrà introdotto il software nel settore amministrazione sono:

1. Raccolta degli ordini (ordini clienti) provenienti dalla clientela (negozi e catene di distribuzione).
2. Smistamento degli ordini clienti.

Le attività tipiche in cui verrà utilizzato il software negli stabilimenti produttivi sono:

3. Ricezione e schedulazione lotti di produzione.
4. Approvvigionamento materie prime.
5. Inventario e registrazione pezzi prodotti.

2.2.2 Requisiti funzionali

Il sistema può essere scomposto nelle seguenti macro funzionalità cosi come sono percepite dai dipendenti nelle diverse locazioni:

- Gestione dipendenti
- Gestione clienti
- Gestione ordini
- Smistamento degli ordini clienti
- Schedulazione lotti di produzione
- Gestione fornitori
- Gestione magazzino
- Inventario e registrazione pezzi prodotti.

Analizziamo nel dettaglio le funzionalità.

2.2.2.1 Gestione dipendenti

Il sistema permette di archiviare i dati relativi ai dipendenti dell’azienda. Le funzionalità previste dal programma sono le seguenti:

- Inserimento nuovo dipendente
- Modifica dati dipendente
- Visualizzazione dati dipendente

Figura professionale coinvolta: amministratore di sistema.

2.2.2.2 Gestione clienti

Il sistema permette di archiviare i dati relativi ai clienti dell'azienda. Le funzionalità previste dal programma sono le seguenti:

- Inserimento nuovo cliente
- Modifica dati cliente
- Visualizzazione dati cliente
- Ricerca dati cliente

Figura professionale coinvolta: addetto ufficio clienti.

2.2.2.3 Gestione ordini

Il sistema permette la raccolta degli ordini (ordini clienti) provenienti dalla clientela (negozi e catene di distribuzione). Questi ordini si caratterizzano per il prevedere un certo numero di pezzi di ognuno dei due modelli con delle specifiche date di consegna (anche differenti per i 2 modelli). Le funzionalità previste dal programma sono le seguenti:

- Inserimento nuovo ordine
- Modifica ordine
- Eliminazione ordine (solo nel caso in cui non sia già stato assegnato ad un lotto di produzione)
- Visualizzazione ordine
- Ricerca ordini cliente

Figura professionale coinvolta: addetto ufficio clienti.

2.2.2.4 Smistamento degli ordini clienti

Gli ordini vengono assegnati agli stabilimenti di produzione creando (automaticamente ma con partecipazione/supervisione umana) dei lotti di produzione caratterizzati da un numero costante di pezzi dello stesso modello (compreso tra 50 e 150) e data di consegna vicina. Funzione svolta da parte dell'addetto alla produzione. Se non è possibile creare dei lotti di produzione omogenei per data e/o di numero pezzi adeguato la decisione viene rimandata al responsabile del settore produzione che può decidere se inviare il lotto anomalo alla produzione oppure rifiutare l'ordine. Il programma assembla automaticamente soltanto lotti che siano facilmente identificabili (il resto del lavoro viene effettuato dall'utente interagendo con il programma).

Le funzionalità previste dal programma sono le seguenti:

- Creazione automatica dei lotti di produzione
- Creazione nuovo lotto

- Assegnazione e rimozione ordine a lotto
- Inserimento lottizzazione
- Recupero lottizzazione
- Recupero ordini
- Rifiuto ordine

Il sistema, inoltre, permette la gestione dei permessi dell'addetto produzione e del responsabile settore produzione, permettendo solo al secondo la creazione di lotti anomali, il rifiuto di un ordine e lo smistamento dei lotti di produzione agli stabilimenti.
Figure professionali coinvolte: addetto alla produzione, responsabile settore produzione.

2.2.2.5 Schedulazione lotti di produzione

Il sistema offre un supporto alla decisione dell'operatore fornendo i dati importanti per le decisioni e raccogliendo le sue scelte. Si suppone che siano sempre disponibili in magazzino le materie prime necessarie.
Le funzionalità previste dal programma sono le seguenti:

- Recupero lottizzazione e schedulazione
- Schedulazione dei lotti di produzione
- Visualizzazione dei lotti di produzione (numero pezzi, data di creazione, data di scadenza)
- Visualizzazione diagramma di Gantt dei lotti schedulati

Figura professionale: responsabile di stabilimento.

2.2.2.6 Gestione fornitori

Il sistema permette di archiviare i dati relativi ai fornitori di materie prime e componenti pronte dell'azienda. Le funzionalità previste dal programma sono le seguenti:

- Inserimento nuovo fornitore
- Modifica dati fornitore
- Visualizzazione dati fornitore
- Ricerca dati fornitore

Figura professionale coinvolta: responsabile magazzino.

2.2.2.7 Gestione magazzino

A partire dal numero di biciclette da produrre è possibile pianificare gli acquisti di materie prime (es. pezzi di tubo per i telai) e componenti pronti (esempio cerchi per le ruote, viti e bulloni). Per ognuno dei due modelli è presente una distinta di produzione che specifica i componenti e le materie prime necessarie per la costruzione di un singolo esemplare. Il programma permettere la

creazione/modifica di tale distinta di produzione. Per ogni lotto di produzione.assegnato allo stabilimento devono essere approvvigionati i materiali necessari a sostituire le scorte di magazzino usate per la produzione. Il processo si completa con la stampa da inviare ai fornitori dei materiali necessari. Ogni ordine specifica la data di consegna richiesta (secondo gli accordi presi con i fornitori).

Le funzionalità previste dal programma sono le seguenti:

- Creazione e modifica distinta di produzione
- Gestione componenti (creazione, modifica, eliminazione)
- Gestione ordini materie prime e componenti pronte (creazione e modifica)
- Anteprime di stampa e stampa ordine componenti

Figura professionale: responsabile magazzino.

2.2.2.8 Inventario e registrazione pezzi prodotti

Il programma registra il numero di telaio di ognuna delle biciclette prodotte, la data di produzione, il numero di lotto di produzione e il cliente cui si riferisce l'ordine. Il programma genera (in modo sequenziale, con codici identificativi diversi per i due stabilimenti) i numeri di telaio e i relativi dati e ne permette la stampa (con anteprima a video) su una etichetta adesiva che verrà attaccata sulla bicicletta.

Le funzionalità offerte dal programma sono le seguenti:

- Generazione automatica etichette
- Visualizzazione etichette
- Stampa etichette
- Registrazione automatica pezzi prodotti.

Figura professionale: operaio addetto alla produzione.

2.2.3 Requisiti nonfunzionali

2.2.3.1 Interfaccia utente

Il sistema presenta delle interfacce grafiche per i vari utenti a finestre pertanto si presuppone che questi ultimi abbiano dimestichezza con questo tipo di interfacce. Ogni interfaccia mostra ad ogni utente le potenzialità per le quali esso è abilitato. Al fine di evitare inconsistenze nelle informazioni archiviate il sistema propone una serie di controlli nelle operazioni di input e relativa comunicazione all'utente mediante appositi messaggi.

I controlli di cui sopra sono relativi per esempio all'inserimento delle date, delle quantità (ordini e lotti) e alla cancellazione (non è possibile cancellare un ordine inserito in un lotto di produzione).

2.2.3.2 Considerazioni hardware

Al fine di permettere il funzionamento ottimale del sistema si riportano i seguenti requisiti minimi hardware per ogni locazione (l'hardware proposto è quello utilizzato nella fase di testing):

- Amministrazione: server dotato di gruppo di continuità con hard-disk da 100Gb; terminale munito di mouse,tastiera e gruppo di continuità connesso alla rete locale. Processore Pentium 3 700 Mhz dotato di hard-disk 3Gb.

Postazioni di lavoro previste presso l'amministrazione:

- addetto ufficio clienti: terminale munito di mouse, tastiera e gruppo di continuità connesso alla rete locale. Processore Pentium 3 700 Mhz dotato di hard-disk 3Gb.
- addetto alla produzione: terminale munito di mouse, tastiera e gruppo di continuità connesso alla rete locale. Processore Pentium 3 700 Mhz dotato di hard-disk 3Gb.
- responsabile settore produzione: terminale munito di mouse, tastiera e gruppo di continuità connesso alla rete locale. Processore Pentium 3 700 Mhz dotato di hard-disk 3Gb.
- amministratore di sistema: terminale munito di mouse, tastiera e gruppo di continuità connesso alla rete locale. Processore Pentium 3 700 Mhz dotato di hard-disk 3Gb.

Postazioni di lavoro previste presso ogni stabilimento:

- responsabile di stabilimento: terminale munito di mouse, tastiera e gruppo di continuità connesso alla rete locale. Processore Pentium 3 700 Mhz dotato di hard-disk 3Gb.
- responsabile magazzino: terminale munito di mouse, tastiera e gruppo di continuità connesso alla rete locale. Processore Pentium 3 700 Mhz dotato di hard-disk 3Gb.
- operaio addetto alla produzione: terminale munito di mouse, tastiera e gruppo di continuità connesso alla rete locale. Processore Pentium 3 700 Mhz dotato di hard-disk 3Gb.

Rete locale Ethernet da 100 Mbps. Si suppone che le locazioni siano in un unico sito ed è quindi installabile una rete locale.

Ogni computer del sistema deve essere fornito di:

- Jade 3.0b1 con codec RDF
- JRE 1.4

Nel server, oltre al suddetto software, deve esser presente :

- Microsoft Access

2.2.3.3 Gestione degli errori e tolleranza ai guasti

Il sistema prevede un controllo sugli input già a livello di interfaccia. Il sistema, mediante la disabilitazione della possibilità di effettuare operazioni mentre ne è in corso un'altra, cerca di

instradare l’utente verso un corretto utlizzo. Quando si cerca di inserire dati non consentiti l’operazione non viene eseguita e viene mostrato a video un messaggio di errore.

2.2.3.4 Sicurezza

Il sistema prevede un sistema di autenticazione mediante username e password per ogni postazione di lavoro. Si suppone che la sicurezza del sistema sia gestita dal sistema operativo dei singoli terminali e della rete (protezione accessi esterni).

2.2.4 Pseudo – requisiti

Vincolo fondamentale per la progettazione e lo sviluppo del tool è la modellazione di un sistema multiagente (MAS, multi agent system) mediante la metodologia PASSI, usufruendo dell’apposito tool PTK (Passi ToolKit) disponibile come plugin per il Software di progettazione “Rational Rose”.
Il linguaggio utilizzato per lo sviluppo dell’intero sistema è Java dal momento che è stato scelto di implementare il sistema con la tecnologia ad agenti utilizzando la piattaforma JADE.
Il DBMS utilizzato è Microsoft Access poichè il cliente è proprietario di tale licenza.

2.2.5 Modelli del sistema

2.2.5.1 Scenari

Nel seguente paragrafo verranno riportati gli scenari più comuni per il sistema in questione.

2.2.5.1.1 Autenticazione dipendenti

Autenticazione addetto ufficio clienti

Attori coinvolti: Richard: addetto ufficio clienti

Flusso di eventi:

1. Richard appena arriva in ufficio accende il suo terminale
2. Il sistema fa partire il programma di gestione ordini e clienti e mostra a Richard il form per l'autenticazione
3. Richard inserisce il suo nome utente e password e seleziona OK per inviare i propri dati
4. Il sistema verifica i dati forniti da Richard e mostra l'interfaccia del programma di gestione ordini e clienti

Autenticazione addetto ufficio clienti fallita

Attori coinvolti: Richard: addetto ufficio clienti

Flusso di eventi:

1. Richard appena arriva in ufficio accende il suo terminale
2. Il sistema fa partire il programma di gestione ordini e clienti e mostra a Richard il form per l'autenticazione
3. Richard inserisce nome utente e/o password errati e clicca su OK per inviare i dati
4. Il sistema controlla i dati forniti da Richard e mostra una finestra con un avviso di nome utente o password errata
5. Richard legge il messaggio di avviso e chiude la finestra
6. Il sistema mostra nuovamente il form per inserire nome utente e password

Autenticazione addetto produzione

Attori coinvolti: Nick: Addetto alla produzione

Flusso di eventi:

1. appena arriva in ufficio accende il suo terminale
2. Il sistema fa partire il programma per la gestione dei lotti e mostra a Nick il form per l'autenticazione
3. Nick inserisce il suo nome utente e password e seleziona OK per inviare i propri dati
4. Il sistema verifica i dati forniti da Nick, mostra l'interfaccia del programma per la gestione dei lotti e recupera gli ordini non ancora assegnati a dei lotti
5. Il programma crea un primo raggruppamento in lotti (lottizzazione automatica) degli ordini recuperati e lo mostra a Nick

Autenticazione responsabile settore produzione

Attori coinvolti: David: Responsabile settore produzione

Flusso di eventi:

1. David appena arriva in ufficio accende il suo terminale
2. Il sistema fa partire il programma per la gestione dei lotti e mostra a David il form per l'autenticazione
3. David inserisce il suo nome utente e password e seleziona OK per inviare i propri dati
4. Il sistema verifica i dati forniti da David e mostra l'interfaccia del programma per la gestione dei lotti

Autenticazione responsabile stabilimento

Attori coinvolti: Roger: Responsabile di stabilimento

Flusso di eventi:

1. Roger appena arriva in ufficio accende il suo terminale
2. Il sistema fa partire il programma per la gestione della schedulazione e mostra a Roger il form per l'autenticazione
3. Roger inserisce il suo nome utente e password e seleziona OK per inviare i propri dati
4. Il sistema verifica i dati forniti da Roger

5. Il sistema recupera i lotti non ancora schedulati, la schedulazione già effettuata a partire da due settimane prima del giorno correnete e mostra l'interfaccia del programma per la gestione della schedulazione

Autenticazione responsabile magazzino

Attori coinvolti: Syd: Responsabile magazzino

Flusso di eventi:

1. Syd appena arriva in ufficio accende il suo terminale
2. Il sistema fa partire il programma per la gestione del magazzino e mostra a Syd il form per l'autenticazione
3. Syd inserisce il suo nome utente e password e seleziona OK per inviare i propri dati
4. Il sistema verifica i dati forniti da Syd e mostra l'interfaccia del programma per la gestione del magazzino

Autenticazione amministratore di sistema

Attori coinvolti: Joe: Amministratore di sistema

Flusso di eventi:

1. Joe appena arriva in ufficio accende il suo terminale
2. Il sistema fa partire il programma per la gestione dei dipendenti e mostra a Joe il form per l'autenticazione
3. Joe inserisce il suo nome utente e password e seleziona OK per inviare i propri dati
4. Il sistema verifica i dati forniti da Joe e mostra l'interfaccia del programma per la gestione dei dipendenti

2.2.5.1.2 Gestione Ordini e clienti

Modifica dati cliente

Attori coinvolti: Richard: Addetto ufficio clienti

Flusso di eventi:

1. Richard riceve una telefonata di un cliente che vuole modificare i dati registrati

2. Richard apre il programma per la ricerca dei clienti chiede al cliente il suo codice
3. Richard inserisce il codice cliente e visualizza i dati relativi al cliente
4. Richard chiede al cliente i dati da modificare e, una volta finito, conferma le modifiche per aggiornare i dati del cliente

Inserimento ordine di un cliente

Attori coinvolti: Richard: Addetto ufficio clienti

Flusso di eventi:

1. Richard riceve la telefonata di un cliente che vuole effettuare un ordine
2. Richard chiede al cliente se ha già effettuato ordini e quale sia il suo codice cliente
3. Il cliente risponde che ha già effettuato ordini in passato e comunica a Richard il proprio codice cliente
4. Richard apre la finestra per l'inserimento dei dati di un nuovo ordine ed inserisce il codice comunicato dal cliente
5. Richard verifica col cliente che il codice sia corretto e provvede a riempire il form con i dati relativi al numero di biciclette ed alla data di consegna per i 2 tipi di bicicletta possibili A e B
6. Richard conferma l'ordine e lo invia al database

Inserimento ordine di un nuovo cliente

Attori coinvolti: Richard: addetto ufficio clienti

Flusso di eventi:

1. Richard riceve la telefonata di un cliente che vuole effettuare un ordine
2. Richard chiede al cliente se ha già effettuato ordini e quale sia il suo codice cliente
3. Il cliente risponde che non ha mai fatto ordini in passato con questa ditta e non ha, quindi, un codice cliente
4. Richard apre la finestra per inserimento di un nuovo cliente e inserisce il nome del cliente, indirizzo, telefono, partita IVA ed eventualmente e-mail
5. Richard conferma i dati inseriti e memorizza il nuovo cliente

6. Richard apre la finestra per l'inserimento di un nuovo ordine e seleziona il nome del cliente appena aggiunto
7. Richard provvede a riempire il form con i dati relativi al numero di biciclette ed alla data di consegna per i 2 tipi di bicicletta possibili A e B
8. Richard conferma l'ordine e lo invia al database

Modifica di un ordine (cercando gli ordini del cliente)

Attori coinvolti: Richard: Addetto ufficio clienti

Flusso di eventi:

1. Richard riceve una telefonata di un cliente che vuole modificare un ordine effettuato in precedenza
2. Richard apre la finestra relativa alla ricerca degli ordini e chiede al cliente il codice cliente per visualizzarne gli ordini
3. Richard chiede al cliente quando è stato fatto l'ordine ed, una volta individuato l'ordine da modificare, seleziona l'ordine per visualizzarlo
4. Richard modifica l'ordine secondo i desideri del cliente e memorizza le modifiche per aggiornare l'ordine

Modifica di un ordine (visualizzando la lista degli ordini)

Attori coinvolti: Richard: Addetto ufficio clienti

Flusso di eventi:

1. Richard riceve una telefonata di un cliente che vuole modificare un ordine effettuato in precedenza
2. Richard apre la finestra contenente la lista di tutti gli ordini effettuati
3. Richard chiede al cliente quando è stato fatto l'ordine ed, una volta individuato l'ordine da modificare, seleziona l'ordine per visualizzarlo
4. Richard modifica l'ordine secondo i desideri del cliente e memorizza le modifiche per aggiornare l'ordine

Tentativo fallito di modifica di un ordine

Attori coinvolti: Richard: Addetto ufficio clienti

Flusso di eventi:

1. Richard riceve una telefonata di un cliente che vuole modificare un ordine effettuato in precedenza

2. Richard apre la finestra relativa alla ricerca degli ordini e chiede al cliente il codice cliente per visualizzarne gli ordini
3. Richard chiede al cliente quando è stato fatto l'ordine ed, una volta individuato l'ordine da modificare, seleziona l'ordine per visualizzarlo
4. Il sistema avvisa Richard che l'ordine non si può modificare perché è già stato inserito in un lotto
5. Richard comunica al cliente che l'ordine non si può modificare perché è passato troppo tempo dalla sua effettuazione e l'ordine è già stato lottizzato per la produzione

Eliminazione ordine

Attori coinvolti: Richard: Addetto ufficio clienti

Flusso di eventi:

1. Richard riceve una telefonata di un cliente che vuole annullare un ordine effettuato in precedenza
6. Richard apre la finestra relativa alla ricerca degli ordini e chiede al cliente il codice cliente per visualizzarne gli ordini
7. Richard chiede al cliente quando è stato fatto l'ordine ed, una volta individuato l'ordine da eliminare, seleziona l'ordine per visualizzarlo
8. Richard seleziona l'opzione Elimina
9. Il programma mostra una richiesta di conferma di eliminazione dell'ordine
10. Richard conferma l'eliminazione e l'ordine viene eliminato

Rifiuto di un ordine con creazione di un nuovo ordine

Attori coinvolti: Richard: Addetto ufficio clienti

Flusso di eventi:

1. Richard riceve un messaggio dal responsabile settore produzione che informa che si deve rifiutare un ordine e mostra i dati relativi all'ordine da rifiutare
2. Richard chiama il cliente che ha effettuato l'ordine per riferirgli che non è possibile evadere il suo ordine e se vuole annullarlo per farne un altro per un altra scadenza
3. Il cliente accetta di annullare il vecchio ordine per effettuarne un nuovo

4. Richard seleziona il vecchio ordine da annullare e seleziona l'opzione “Elimina” per eliminarlo
5. Richard,quindi, apre la finestra per l'inserimento dei dati di un nuovo ordine ed inserisce i dati relativi al nuovo ordine
6. Richard salva i dati del nuovo ordine selezionando l'opzione “Salva” ed il nuovo lotto è aggiunto alla lista degli ordini

Rifiuto di un ordine con modifica dell'ordine esistente

Attori coinvolti: Richard: Addetto ufficio clienti

Flusso di eventi:

1. Richard riceve un messaggio dal responsabile settore produzione che informa che si deve rifiutare un ordine e mostra i dati relativi all'ordine da rifiutare
2. Richard chiama il cliente che ha effettuato l'ordine per riferirgli che non è possibile evadere il suo ordine e se vuole annullarlo per farne un altro per un altra scadenza
3. Il cliente non accetta di annullare il vecchio ordine perché ha assolutamente bisogno di quelle biciclette per evadere delle richieste importanti
4. Richard cerca di venire incontro al cliente dicendo che è possibile lasciare l'ordine così com'è ma bisogna rinviare la data di consegna di qualche giorno perché non è possibile produrre l'ordine entro la data indicata in precedenza
5. Il cliente accetta di rinviare la consegna di qualche giorno purché le biciclette gli siano consegnate nel più breve tempo possibile
6. Richard seleziona l'ordine da modificare e seleziona l'opzione “Modifica” per cambiare la data di consegna
7. Richard salva i dati dell'ordine modificato selezionando l'opzione “Modifica” e l'ordine viene aggiornato

2.2.5.1.3 Gestione Lotti

Utilizzo lottizzazione automatica senza ordini non assegnati ma con necessità di modifiche

Attori Coinvolti: Nick: Addetto alla produzione

David: Responsabile settore produzione

Flusso di eventi:

1. All'avvio il programma per la gestione dei lotti recupera gli ordini non ancora assegnati a dei lotti e propone a Nick una prima lottizzazione assemblata dal programma stesso in maniera automatica
2. Nick verifica che non ci sono altri ordini da assegnare e si accorge che la lottizzazione automatica generata dal programma può creare problemi di accavallamento in fase di produzione poiché la date di scadenza di alcuni lotti piuttosto grandi sono troppo vicine
3. Nick seleziona uno alla volta i lotti che possono dare problemi e li elimina usando l'apposita opzione
4. Gli ordini non più assegnati vengono riassegnati da Nick a dei nuovi lotti meglio distribuiti per data e numero di pezzi in maniera che non ci possano essere problemi in fase di schedulazione e produzione
5. Nick invia i lotti a David per la conferma definitiva e il salvataggio dei lotti creati nel database
6. il programma per la gestione dei lotti mostra a David la lottizzazione così come è stata spedita da Nick
7. David, dopo aver controllato la lottizzazione, la salva inviandola al database

Utilizzo lottizzazione automatica senza ordini non assegnati e senza necessità di modifiche

Attori Coinvolti: Nick: Addetto alla produzione

David: Responsabile settore produzione

Flusso di eventi:

1. All'avvio il programma per la gestione dei lotti recupera gli ordini non ancora assegnati a dei lotti e propone a Nick una prima lottizzazione assemblata dal programma stesso in maniera automatica
2. Nick verifica che non ci sono altri ordini da assegnare controlla la lottizzazione automatica generata dal programma per vedere se le date di scadenza dei lotti non possano creare problemi di accavallamento in fase di produzione
3. Nick invia i lotti a David per la conferma definitiva e il salvataggio dei lotti creati nel database

4. il programma per la gestione dei lotti mostra a David la lottizzazione così come è stata spedita da Nick
5. David, dopo aver controllato la lottizzazione, la salva inviandola al database

Utilizzo lottizzazione automatica con ordini non assegnati e con creazione di lotti anomali

Attori Coinvolti: Nick: Addetto alla produzione

David: Responsabile settore produzione

Flusso di eventi:

1. All'avvio il programma per la gestione dei lotti recupera gli ordini non ancora assegnati a dei lotti e propone a Nick una prima lottizzazione assemblata dal programma stesso in maniera automatica
2. Nick nota che ci sono degli ordini rimanenti che il programma non è riuscito ad assegnare ad alcun lotto
3. Nick prova a modificare la lottizzazione esistente ma non riesce ad assegnare gli ordini senza superare la dimensione massima consentita per un lotto (150 pezzi)
4. Poiché a Nick non è consentito creare lotti più grandi di 150 pezzi (lotti anomali) invia la lottizzazione così com'è insieme agli ordini non assegnati a David che risolverà il problema
5. David confronta le date di scadenza degli ordini rimanenti con le date di scadenza dei lotti creati e crea dei lotti anomali per poter assegnare gli ordini ai lotti esistenti
6. David controlla la lottizzazione per assicurarsi che non ci siano problemi e, quindi la invia al database per il salvataggio

Utilizzo lottizzazione automatica con ordine non assegnato e con rifiuto di ordine

Attori Coinvolti: Nick: Addetto alla produzione

David: Responsabile settore produzione

Flusso di eventi:

1. All'avvio il programma per la gestione dei lotti recupera gli ordini non ancora assegnati a dei lotti e propone a Nick una prima lottizzazione assemblata dal programma stesso in maniera automatica

2. Nick nota che c'è un ordine che il programma non è riuscito ad assegnare ad alcun lotto
3. Nick prova a modificare la lottizzazione esistente ma non riesce ad assegnare l'ordine senza superare la dimensione massima consentita per un lotto (150 pezzi)
4. Poiché a Nick non è consentito creare lotti più grandi di 150 pezzi (lotti anomali) invia la lottizzazione così com'è insieme agli ordini non assegnati a David che risolverà il problema
5. David confronta le date di scadenza dell'ordine non assegnato con le date di scadenza dei lotti creati e decide che non vale la pena modificare la lottizzazione esistente o creare un nuovo lotto a causa di un solo ordine
6. David seleziona l'ordine non assegnato e seleziona l'opzione “Rifiuta ordine” per eliminarlo dalla lista degli ordini
7. Il programma di gestione dei lotti mostra una finestra di conferma di rifiuto dell'ordine
8. David conferma il rifiuto e l'ordine viene rifiutato
9. David controlla la lottizzazione per assicurarsi che non ci siano problemi e, quindi la invia al database per il salvataggio

2.2.5.1.4 Schedulazione dei lotti

Creazione schedulazione lotti

Attori coinvolti: Roger: Responsabile di stabilimento A

Flusso di eventi:

1. il programma visualizza i lotti recuperati indicando per ciascun lotto dimensione, scadenza e numero di pezzi già schedulati
2. Roger assegna per ogni giorno i lotti ed il numero di pezzi per ogni lotto che devono essere prodotti
3. Schedulati tutti i lotti, Roger salva la schedulazione in modo che sia inviata al database

Schedulazione lotto in giorni non lavorativi

Attori coinvolti: Roger: Responsabile di stabilimento A

Flusso di eventi:

1. il programma visualizza i lotti recuperati indicando per ciascun lotto dimensione, scadenza e numero di pezzi già schedulati
2. Roger seleziona un giorno non lavorativo per la schedulazione, inserisce la quantità e conferma
3. Il sistema mostra un form avvertendo l'utente che si sta tentando di schedulare in un giorno non lavorativo e chiede conferma
4. Roger conferma l'inserimento

Schedulazione oltre la data di scadenza

Attori coinvolti: Roger: Responsabile di stabilimento A

Flusso di eventi:

1. il programma visualizza i lotti recuperati indicando per ciascun lotto dimensione, scadenza e numero di pezzi già schedulati
2. Roger seleziona un giorno successivo alla data di scadenza del lotto per la schedulazione, inserisce la quantità e conferma
3. Il sistema mostra un messaggio di errore all'utente specificando che si sta tentando di schedulare un lotto oltre la data di scadenza

Schedulazione oltre la capacità massima giornaliera dello stabilimento

Attori coinvolti: Roger: Responsabile di stabilimento A

Flusso di eventi:

1. il programma visualizza i lotti recuperati indicando per ciascun lotto dimensione, scadenza e numero di pezzi già schedulati
2. Roger seleziona un giorno per la schedulazione, inserisce la quantità e conferma
3. Il sistema mostra un messaggio di errore all'utente specificando che si sta tentando di produrre per un giorno un numero di biciclette superiore alla capacità massima dello stabilimento

2.2.5.1.5 Gestione distinta di produzione

Modifica distinta di produzione

Attori coinvolti: Syd: Responsabile magazzino

Flusso di eventi:

1. Syd seleziona l’opzione “Modifica Distinta”
2. Il programma di gestione dei consumi mostra una finestra contenente la distinta di produzione del modello di bicicletta
3. Syd modifica i componenti del modello secondo le proprie necessità
4. Syd salva le le modifiche apportate alla distinta di produzione
5. Il programma mostra una richiesta di conferma prima di inviare i nuovi dati al database
6. Syd conferma la richiesta di salvataggio e i dati vengono inviati al database

2.2.5.1.6 Pianificazione ordini materie prime

Creazione ordine a fornitori

Attori coinvolti: Syd: Responsabile magazzino

Flusso di eventi:

1. Syd seleziona l'opzione “Crea ordine” per creare un nuovo ordine materie prime e componenti pronte
2. Il programma mostra un form contenente gli ordini per i vari fornitori
3. Syd seleziona l'ordine da creare dalla lista degli ordini visualizzata, lo apre e riempie i campi relativi al numero di pezzi e la data di consegna
4. Syd salva l'ordine appena creato per inviarlo al database per la sua evasione
5. il programma chiede conferma alla richiesta di salvataggio dell'ordine
6. Syd risponde affermativamente alla richiesta di conferma e l'ordine viene salvato

Creazione e stampa di ordine a fornitori

Attori coinvolti: Syd: Responsabile magazzino

Flusso di eventi:

1. Syd seleziona l'opzione “Crea ordine” per creare un nuovo ordine materie prime e componenti pronte
2. Il programma mostra un form contenente gli ordini per i vari fornitori

3. Syd seleziona l'ordine da creare dalla lista degli ordini visualizzata, lo apre e riempie i campi relativi al numero di pezzi e la data di consegna
4. Syd seleziona l'opzione “Anteprima di stampa” per visualizzare a video l'aspetto dell'ordine dopo la stampa
5. Dopo aver visualizzato l'ordine per il fornitore Syd seleziona l'opzione “Stampa” per stampare l'ordine su carta
6. Il programma stampa l'ordine

Inserimento di un nuovo componente

Attori coinvolti: Syd: Responsabile magazzino

Flusso di eventi:

1. Syd seleziona l'opzione “Nuovo componente” per aggiungere una nuova componente alla lista dei componenti che è possibile aggiungere ad un modello della bicicletta
2. Il programma mostra un form per l'inserimento dei dati relativi al nuovo componente e il numero di pezzi del componente che è possibile montare per un determinato modello
3. Syd salva il nuovo componente per essere visualizzato nella lista di tutti i componenti
4. Il programma chiede conferma del salvataggio del nuovo componente
5. Syd conferma il salvataggio ed il nuovo componente è aggiunto nella lista dei componenti

Modifica di un componente

Attori coinvolti: Syd: Responsabile magazzino

Flusso di eventi:

1. Syd seleziona l'opzione “Visualizza lista componenti” per modificare i dati di un componente già esistente nella lista dei componenti
2. Syd seleziona dalla lista dei componenti il componente di cui si devono cambiare i dati e lo apre per modificarlo
3. Syd modifica i dati da modificare, quindi seleziona l'opzione “Modifica” per salvare le modifiche apportate al componente
4. Il programma mostra una richiesta di conferma del salvataggio

5. Syd conferma il salvataggio e i dati modificati vengono inviati al database per essere utilizzati quando necessario

Modifica dei dati di un fornitore

Attori coinvolti: Syd: Responsabile magazzino

Flusso di eventi:

1. Syd seleziona l'opzione “Apri dati fornitori” per modificare i dati di un fornitore
2. Il programma mostra una lista di tutti i fornitori conosciuti
3. Syd seleziona il fornitore di cui deve modificare i dati e apre la finestra dettagliata di tutti i suoi dati per modificarli
4. Syd effettua le modifiche necessarie ai dati e seleziona l'opzione “Modifica” per salvare i nuovi dati
5. Il sistema chiede conferma della richiesta di modifica dei dati del fornitore
6. Syd risponde in maniera affermativa alla richiesta di conferma e i dati vengono salvati nel database per essere utilizzati successivamente

Inserimento nuovo fornitore

Attori coinvolti: Syd: Responsabile magazzino

Flusso di eventi:

1. Syd seleziona l'opzione “Nuovo fornitore” per inserire i dati di un nuovo fornitore
2. Il programma di gestione dei fornitori mostra una nuova finestra per inserire i dati del nuovo fornitore
3. Syd salva i dati del nuovo fornitore selezionando l'opzione “Salva”
4. Il programma di gestione fornitori mostra una finestra contenente una richiesta di conferma del salvataggio dei nuovi dati
5. Syd conferma il salvataggio ed i dati sono inviati al database per essere utilizzati in seguito

2.2.5.1.7 Gestione dipendenti

Modifica dati dipendente

Attori coinvolti: Joe: Amministratore di sistema

Flusso di eventi:

1. Joe seleziona il dipendente di cui vuole cambiare i dati dalla lista dei dipendenti
2. Il programma di gestione dipendenti visualizza in una nuova finestra i dati relativi al dipendente
3. Joe effettua le necessarie modifiche ai dati del dipendente e, quindi salva i nuovi dati selezionando l'opzione “Modifica”
4. Il programma di gestione dipendenti mostra una richiesta di conferma del salvataggio dei dati del dipendente
5. Joe conferma il salvataggio ed i dati sono inviati al database per essere utilizzati in seguito

2.2.5.1.8 Produzione biciclette

Stampa etichetta bicicletta

Attori coinvolti: Freddie: Operaio addetto alla produzione

Flusso di eventi:

1. Freddie visualizza a video l'etichetta relativa alla prossima bicicletta da produrre così come essa apparirà una volta stampata
2. Freddie stampa l'etichetta selezionando l'opzione “Stampa” che compare accanto all'area in cui è visualizzata l'etichetta
3. Il programma invia l'etichetta alla stampante per la stampa e visualizza l'etichetta relativa alla successiva bicicletta da produrre
4. Freddie incolla l'etichetta appena stampata al telaio della bicicletta appena prodotta

Prima bicicletta della giornata

Attori coinvolti: Freddie: Operaio addetto alla produzione

Flusso di eventi:

1. Freddie la mattina arriva alla sua postazione ed accende il sistema usando l'apposito tasto
2. Il programma di gestione della produzione fa partire la sua interfaccia grafica e visualizza l'etichetta della prima bicicletta in produzione quel giorno

3. Freddie stampa l'etichetta visualizzata selezionando l'opzione “Stampa” che compare accanto all'area in cui è visualizzata l'etichetta
4. Il programma invia l'etichetta alla stampante per la stampa e visualizza l'etichetta relativa alla successiva bicicletta da produrre
5. Freddie incolla l'etichetta appena stampata al telaio della bicicletta appena prodotta

Ultima bicicletta della giornata

Attori coinvolti: Freddie: Operaio addetto alla produzione

Flusso di eventi:

1. Freddie visualizza a video l'etichetta relativa alla prossima bicicletta da produrre così come essa apparirà una volta stampata
2. Freddie stampa l'etichetta selezionando l'opzione “Stampa” che compare accanto all'area in cui è visualizzata l'etichetta
3. Il programma invia l'etichetta alla stampante per la stampa e stampa un avviso che sono state schedulate altre biciclette per questa giornata
4. Freddie incolla l'etichetta appena stampata al telaio della bicicletta appena prodotta
5. Freddie visualizza il messaggio visualizzato sullo schermo spegne il sistema e se ne va

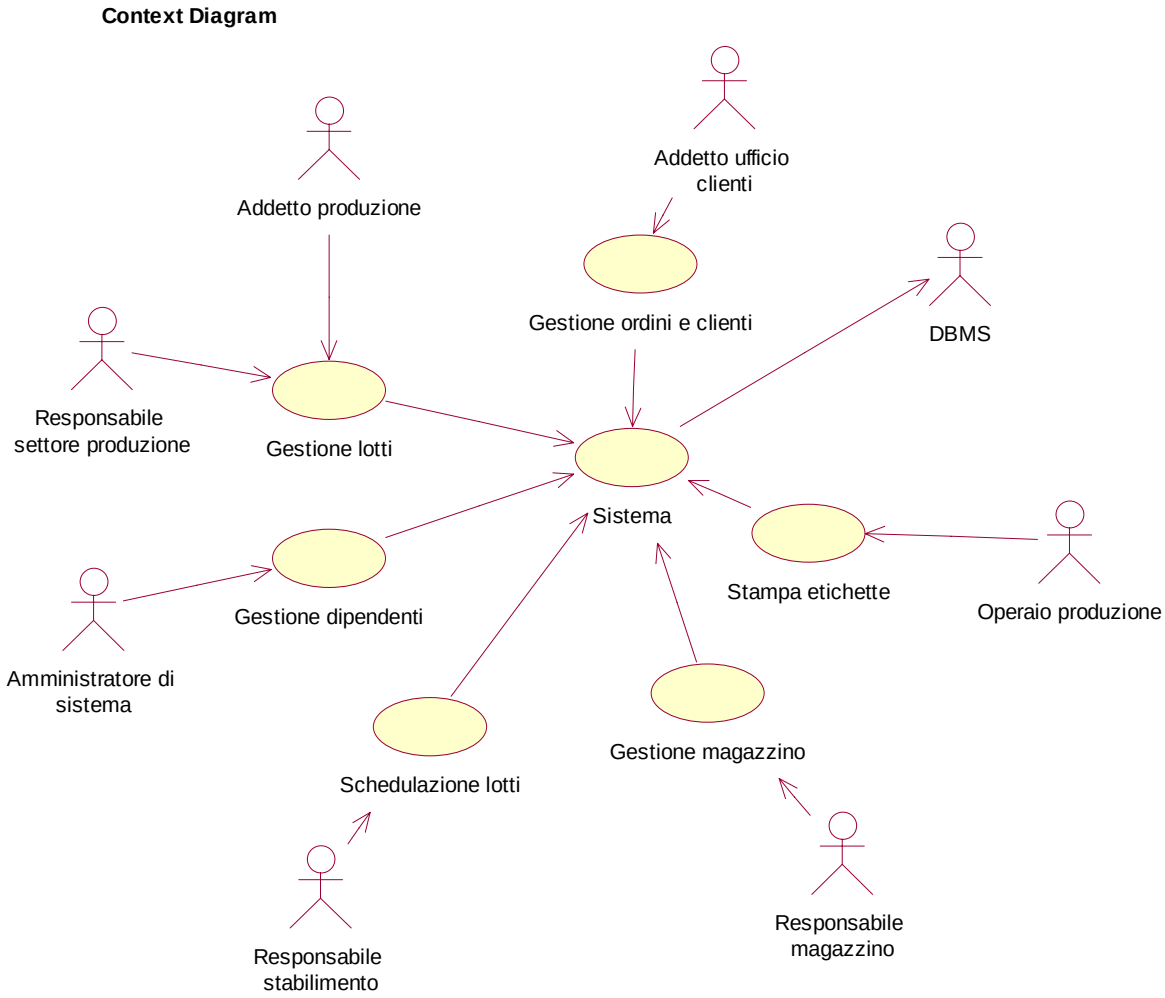
PASSI PROJECT

3 PASSI project

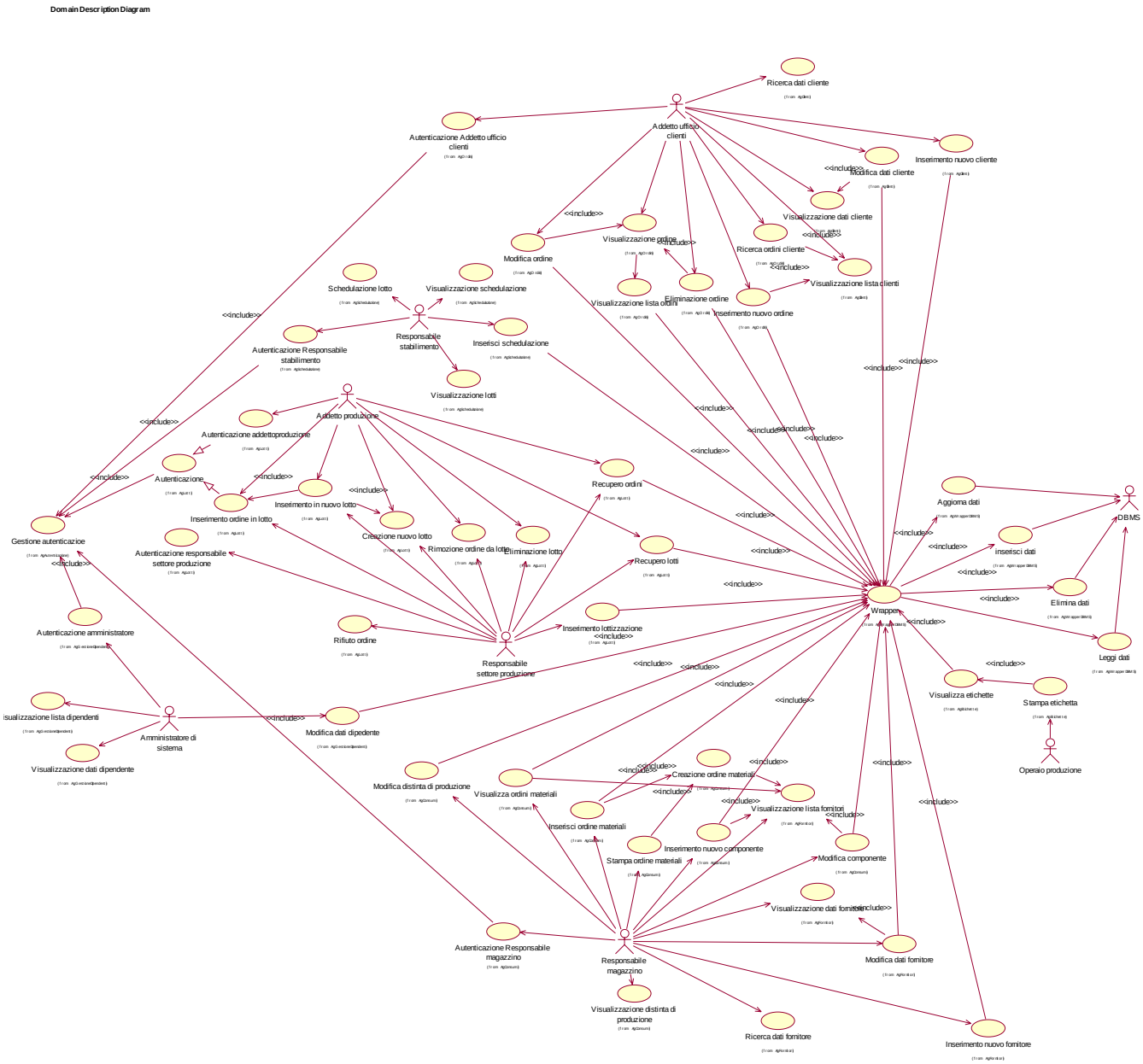
3.1 Domain Description phase

Descriviamo i requisiti del sistema direttamente in termini di diagramma dei casi d'uso. Per far questo dobbiamo progettare una serie gerarchica di diagrammi dei casi d'uso che forniscano una descrizione funzionale del sistema. Il diagramma al livello di astrazione più alto è il 'context diagram'. Gli scenari dettagliati dei diagrammi dei casi d'uso verranno illustrati utilizzando dei diagrammi di sequenza.

Questo è diagramma dei casi d'uso al più alto livello di astrazione utilizzato per la descrizione funzionale del sistema. L'analisi funzionale del sistema comincia da qui.



Questo è il diagramma dei casi d'uso più dettagliato. L'Agent Identification comincia da qui. E' possibile pensare ad un agente come un caso d'uso o un package di casi d'uso.



Use Case	*Description*
Ricerca dati cliente	Attori partecipanti: Addetto Ufficio Clienti Condizione d'ingresso: L'utente seleziona l'opzione "Cerca dati cliente" Flusso d'eventi: 1. Il sistema mostra il form per la ricerca dei dati del cliente; 2. L'utente inserisce i parametri della ricerca nel form; 3. L'utente avvia la ricerca selezionando l'opzione "Cerca"; 4. Il sistema mostra i risultati corrispondenti ai criteri di ricerca impostati 5. L'utente seleziona il cliente desiderato; Condizione d'uscita: L'utente conferma la selezione
Inserimento nuovo cliente	Attori partecipanti: Addetto ufficio clienti Condizione di ingresso: L'addetto ufficio clienti seleziona l'opzione "Aggiungi nuovo cliente" Flusso di eventi: 1. L'addetto ufficio clienti inserisce i dati del cliente nel form di registrazione 2. I dati vengono memorizzati DBMS tramite il caso d'uso "Wrapper" Condizione di uscita: L'addetto ufficio clienti conferma l'inserimento

Modifica dati cliente

Attori partecipanti:

Addetto ufficio clienti

Condizione di ingresso:

L'utente seleziona il cliente di cui desidera modificare i dati

Flusso di eventi:

1. Il sistema preleva i dati relativi al cliente tramite il caso d'uso "Visualizzazione dati cliente"
2. Il sistema mostra all'addetto un form con i dati del cliente
3. L'utente modifica i campi di interesse
4. L'utente conferma la modifica

Condizione di uscita:

I dati vengono modificati nel DBMS tramite il caso d'uso "wrapper"

Visualizzazione dati cliente

Attori partecipanti:

Addetto ufficio clienti

Condizione di ingresso:

L'addetto ufficio clienti seleziona l'opzione "Apri dati cliente"

Flusso di eventi:

1. L'utente seleziona il cliente di cui desidera visualizzare i dati
1. 2.Il sistema legge i dati relativi al cliente dal DBMS tramite il caso d'uso "Wrapper"

Condizione di uscita:

Il sistema mostra all'utente i dati del cliente

Visualizzazione lista clienti

Attori partecipanti:

Addetto ufficio clienti

Condizione di ingresso:

L'utente seleziona l'opzione "visualizza lista clienti"

Flusso di eventi:

1. Il sistema preleva i dati dal DBMS tramite il caso d'uso "wrapper"

Condizione di uscita:

Il sistema mostra la lista dei clienti

Ricerca ordini cliente

Attori partecipanti:

Addetto ufficio clienti

Condizione di ingresso:

L'addetto ufficio clienti seleziona l'opzione "Cerca ordini cliente"

Flusso di eventi:

1. L'utente seleziona il cliente dalla lista visualizzata tramite il caso d'uso "Visualizza lista clienti"
2. L'utente seleziona l'opzione "cerca"

Condizione di uscita:

Il sistema mostra all'utente la lista degli ordini effettuati dal cliente.

Inserimento nuovo ordine

Attori partecipanti:

Addetto ufficio clienti

Condizione di ingresso:

L'addetto ufficio clienti seleziona "Nuovo ordine"

Flusso di eventi:

1. Il sistema mostra il form di inserimento dei dati
2. L'utente seleziona il cliente dalla lista visualizzata tramite il caso d'uso "Visualizza lista clienti"
3. L'utente riempie il form inserendo quantità, modello e data di consegna richiesta dell'ordine
4. L'addetto ufficio clienti conferma l'inserimento dell'ordine

Condizione di uscita:

L'ordine viene inserito nel DBMS utilizzando il caso d'uso "Wrapper"

Visualizzazione lista ordini

Attori partecipanti:

Addetto Ufficio Clienti

Condizione di ingresso:

L'Addetto Ufficio Clienti seleziona l'opzione "Visualizza lista ordini"

Flusso di eventi:

1. Il sistema mostra all'utente un form per selezionare l'intervallo di tempo
2. Il sistema preleva gli ordini effettuati nell'intervallo di tempo selezionato tramite il caso d'uso "wrapper"

Condizione di uscita:

Il sistema visualizza la lista degli ordini

Eliminazione ordine

Attori partecipanti:

Addetto ufficio clienti

Condizione di ingresso:

L'utente seleziona l'ordine da eliminare

Flusso di eventi:

1. L'utente apre l'ordine selezionato tramite il caso d'uso "Visualizza ordine"
2. L'utente seleziona l'opzione "elimina ordine"
3. Il sistema chiede la conferma dell'eliminazione dell'ordine

Condizione di uscita:

Il sistema elimina dal DBMS l'ordine selezionato tramite il caso d'uso "wrapper"

Visualizzazione ordine	Attori partecipanti: Addetto ufficio clienti Condizione di ingresso: L'addetto ufficio clienti seleziona l'ordine da visualizzare Flusso di eventi: 1. L'utente seleziona l'opzione "Apri ordine" Condizione di uscita: Il sistema mostra all'utente l'ordine
Modifica ordine	Attori partecipanti: Addetto ufficio clienti Condizione di ingresso: L'utente seleziona l'ordine da eliminare Flusso di eventi: 1. L'utente apre l'ordine selezionato tramite il caso d'uso "Visualizza ordine" 2. L'utente effettua le modifiche volute

3. L'utente seleziona l'opzione "modifica ordine"
4. Il sistema chiede la conferma dela modifica dell'ordine

Condizione di uscita:

Il sistema modifica l'ordine nel DBMS tramite il caso d'uso "wrapper"

Autenticazione Addetto ufficio clienti

Attori partecipanti:

Addetto ufficio clienti

Condizione di ingresso:

L'Addetto ufficio clienti avvia il sistema

Flusso di eventi:

1. Il sistema mostra un form per l'inserimento dei dati per l'autenticazione
2. L'Addetto ufficio clienti inserisce username e password e conferma
3. Il sistema verifica i dati utilizzando il caso d'uso "GestioneAutenticazione"

Condizione di uscita:

Il sistema avvia il programma di gestione clienti e ordini

Schedulazione lotto

Attori partecipanti:

Responsabile stabilimento

Condizione di ingresso:

L'utente seleziona un lotto da schedulare

Flusso di eventi:

1. L'utente seleziona un giorno

Condizione di uscita:

L'utente inserisce la quantità a schedulare per il giorno selezionato

Visualizzazione schedulazione

Attori partecipanti:

Responsabile stabilimento:

Condizione di ingresso:

L'utente seleziona la visualizzazione della schedulazione

Flusso di eventi:

1. L'utente seleziona l'intervallo di tempo della schedulazione da visualizzare

Condizione di uscita:

Il sistema visualizza la schedulazione per l'intervallo selezionato.

Inserisci schedulazione

Attori partecipanti:

Responsabile stabilimento

Condizione d'ingresso:

L'utente seleziona l'opzione "inserisci schedulazione"

Flusso di eventi:

1. Il sistema mostra un from per la conferma dell'inserimento dei dati
2. L'utente conferma l'inserimento

Condizione di uscita:

Il sistema inserisce i dati nel DBMS tramite il caso d'uso "wrapper"

Visualizzazione lotti	Attori partecipanti: Responsabile stabilimento Condizione di ingresso: L'utente seleziona la visualizzazione dei lotti Condizione di uscita: Il sistema mostra i lotti schedulati e da schedulare
Autenticazione Responsabile stabilimento	Attori partecipanti: Responsabile stabilimento Condizione di ingresso: Il responsabile stabilimento avvia il sistema Flusso di eventi: 1. Il sistema mostra un form per l'inserimento dei dati per l'autenticazione 2. Il responsabile stabilimento inserisce username e password e conferma 3. Il sistema verifica i dati utilizzando il caso d'uso "GestioneAutenticazione" Condizione di uscita: Il sistema avvia il programma di schedulazione dei lotti
Recupero ordini	Attori coinvolti: Addetto produzione o responsabile settore produzione Condizione di ingresso: L'utente seleziona l'opzione "recupera ordini"

Flusso di eventi:

1. Il sistema preleva gli ordini tramite il caso d'uso "wrapper"

Condizione d'uscita:

Il sistema visualizza gli ordini

Recupero lotti

Attori coinvolti:

Addetto produzione o responsabile settore produzione

Condizione di ingresso:

L'utente seleziona l'opzione "recupera lotti"

Flusso di eventi:

1. Il sistema preleva i lotti tramite il caso d'uso "wrapper"

Condizione d'uscita:

Il sistema visualizza i lotti

Inserimento lottizzazione

Attori partecipanti:

Responsabile settore produzione

Condizione di ingresso:

L'utente seleziona "salva lottizzazione"

Flusso di eventi:

1. Il sistema chiede conferma

Condizione di uscita:

L'utente seleziona conferma e il sistema invia i dati al DBMS tramite il caso d'uso "wrapper"

Eliminazione lotto

Attori partecipanti:

Addetto Produzione o Responsabile Produzione

Condizione di ingresso:

L'utente seleziona un lotto e seleziona "Elimina lotto"

1. Il sistema chiede una conferma all'utente
2. L'utente seleziona conferma
3. Il sistema elimina il lotto

Condizione di uscita:

Il sistema aggiorna la lista degli ordini e dei lotti

Rimozione ordine da lotto

Attori partecipanti:

Addetto Produzione o Responsabile Produzione

Condizione di ingresso:

L'utente seleziona uno o più ordini e seleziona l'opzione "Rimuovi da lotto"

Flusso di eventi:

1. Il sitema rimuove gli ordini selezionati dal lotto

Condizione di uscita:

Il sistema aggiorna la lista degli ordini

Creazione nuovo lotto

Attori partecipanti:

Addetto produzione o Responsabile settore produzione

Condizione di ingresso:

L'utente seleziona l'opzione "Crea nuovo lotto"

Flusso di eventi:

1. Il sistema crea un nuovo lotto

Condizione di uscita:

Il sistema aggiorna la lista dei lotti

Rifiuto ordine

Attori partecipanti:

Reponsabile settore produzione

Condizione di ingresso:

Il Responsabile settore produzione seleziona un ordine e sceglie l'opzione "rifiuta ordine"

Flusso di eventi:

1. Il sistema mostra un form con la richiesta di conferma

Condizione di uscita:

L'utente conferma il rifiuto dell'ordine

Inserimento in nuovo lotto

Attori partecipanti:

Addetto produzione o Responsabile settore produzione

Condizione di ingresso:

L'utente seleziona un ordine e seleziona l'opzione "assegna nuovo lotto"

Flusso di eventi:

1. Il sistema crea un nuovo lotto tramite il caso d'uso "Creazione nuovo lotto"

Condizione di uscita:

Il sistema assegna l'ordine selezionato al nuovo lotto tramite il caso d'uso "Inserimento ordine in lotto"

Autenticazione addetto produzione	Attori partecipanti: Addetto produzione Condizione di ingresso: L'Addetto produzione avvia il sistema Flusso di eventi: 1. Il sistema mostra un form per l'inserimento dei dati per l'autenticazione 2. L'Addetto produzione inserisce username e password e conferma 3. Il sistema verifica i dati utilizzando il caso d'uso "gestione autenticazione" Condizione di uscita: Il sistema avvia il programma di gestione lotti attivando le limitazioni dei permessi per l'addetto produzione
Inserimento ordine in lotto	Attori partecipanti: Addetto Produzione o Responsabile Produzione Condizione di ingresso: L'utente seleziona un ordine e un lotto Flusso di eventi: 1. L'utente seleziona l'opzione assegna Condizione di uscita: Il sistema aggiorna la lista degli ordini e dei lotti
Autenticazione responsabile settore produzione	Attori partecipanti: Responsabile settore produzione

Condizione di ingresso:

Il responsabile settore produzione avvia il sistema

Flusso di eventi:

1. Il sistema mostra un form per l'inserimento dei dati per l'autenticazione
2. Il responsabile settore produzione inserisce username e password e conferma
3. Il sistema verifica i dati utilizzando il caso d'uso "gestione autenticazione"

Condizione di uscita:

Il sistema avvia il programma di gestione lotti attivando tutti i permessi per il responsabile settore produzione

Autenticazione	Attori partecipanti: Dipendente settore produzione Condizione di ingresso: Il dipendente settore produzione avvia il sistema Flusso di eventi: 1. Il sistema mostra un form per l'inserimento dei dati per l'autenticazione 2. Il dipendente settore produzione inserisce username e password e conferma 3. Il sistema verifica i dati utilizzando il caso d'uso "gestione autenticazione" Condizione di uscita: Il sistema avvia il programma di gestione lotti attivando i permessi per il dipendente

Gestione autenticazioe

Condizione di ingresso:

Il sistema riceve una richiesta di verifica dati utente

Flusso di eventi:

1. Il sistema verifica username e password

Condizione di uscita:

Il sistema notifica l'accettazione dell'utente

Autenticazione amministratore

Attori partecipanti:

Amministratore dei sistema

Condizione di ingresso:

L'Amministratore di sistema avvia il sistema di gestione dipendenti

Flusso di eventi:

1. Il sistema mostra un form per l'inserimento dei dati per l'autenticazione
2. L'Amministratore dei sistema inserisce username e password e conferma
3. Il sistema verifica i dati utilizzando il caso d'uso "GestioneAutenticazione"

Condizione di uscita:

Il sistema avvia il programma di gestione dipendenti

Modifica dati dipedente

Attori partecipanti:

Amministratore di sistema

Condizione di ingresso:

L'utente seleziona il dipendente di cui desidera modificare i dati

Flusso di eventi:

1. Il sistema preleva i dati relativi al dipendente tramite il caso d'uso "Visualizzazione dati dipendente"
2. Il sistema mostra all'addetto un form con i dati del dipendente
3. L'utente modifica i campi di interesse
4. L'utente conferma la modifica

Condizione di uscita:

I dati vengono modificati nel DBMS tramite il caso d'uso "wrapper"

Visualizzazione dati dipendente	Attori partecipanti: Amministratore di sistema Condizione di ingresso: L'utente seleziona il dipendente di cui desidera visualizzare i dati Flusso di eventi: 1. L'utente seleziona l'opzione "Apri dati dipendente" Condizione di uscita: Il sistema mostra all'utente i dati del dipendente
Visualizzazione lista dipendenti	Attori partecipanti: Amministratore di sistema Condizione di ingresso: L'utente seleziona l'opzione "visualizza lista dipendenti" Condizione di uscita: Il sistema mostra la lista dei dipendenti

Stampa etichetta

Attori coinvolti:

Operaio Produzione

Condizione di ingresso:

L'Operaio Produzione seleziona l'opzione "Stampa"

Flusso di eventi:

1. Il sistema mostra all'utente i dati della bicicletta tramite il caso d'uso "VisualizzaEtichette"
2. Il sistema stampa l'etichetta contenente il numero di telaio della biciclette, la data di produzione (odierna), il numero del lotto di produzione e il cliente a cui si riferisce l'ordine

Condizione di uscita:

L'addetto ritira l'etichette dalla stampante

Visualizza etichette

Condizione di ingresso:

Il sistema preleva i dati relativi alla bicicletta tramite il caso d'uso "Wrapper"

Condizione di uscita:

Il sistema visualizza i dati sul monitor

Wrapper

Condizione di ingresso:

Il sistema riceve una richiesta di accesso al DBMS

Flusso di eventi:

1. Il sistema interagisce col DBMS tramite i casi d'uso: "Inserisci dati", "Aggiorna dati", "Elimina dati", "Leggi dati"

Condizione di uscita:

Il sistema notifica l'avvenuta operazione

Aggiorna dati	Condizione d'ingresso: Il sistema chiede l'aggiornamento di un dato già contenuto nel DBMS Condizione d'uscita: Il dato viene aggiornato nel DBMS
inserisci dati	Condizione d'ingresso: Il sistema richiede l'inserimento di un dato nel DBMS; Condizione d'uscita: Il dato viene inserito nel DBMS.
Elimina dati	Condizione d'ingresso: Il sistema richiede l'eliminazione di un dato dal DBMS; Condizione d'uscita: Il dato viene eliminato dal DBMS
Leggi dati	Condizione d'ingresso: Il sistema richiede la lettura di un dato nel DBMS Condizione d'uscita: Il dato viene letto dal DBMS
Inserimento nuovo fornitore	Attori partecipanti: Resopnsabile magazzuino Condizione di ingresso: L'utente seleziona l'opzione "Aggiungi nuovo fornitore"

Flusso di eventi:

1. L'utente inserisce i dati del fornitore nel form di registrazione
2. I dati vengono memorizzati DBMS tramite il caso d'uso "Wrapper"

Condizione di uscita:

L'utente conferma l'inserimento

Modifica dati fornitore	Attori partecipanti: Responsabile magazzino Condizione di ingresso: L'utente seleziona il fornitore di cui desidera modificare i dati Flusso di eventi: 1. Il sistema preleva i dati relativi al fornitore tramite il caso d'uso "Visualizzazione dati fornitore" 2. Il sistema mostra all'addetto un form con i dati del cliente 3. L'utente modifica i campi di interesse 4. L'utente conferma la modifica Condizione di uscita: I dati vengono modificati nel DBMS tramite il caso d'uso "wrapper"
Ricerca dati fornitore	Attori partecipanti: Responsabile magazino Condizione d'ingresso: L'utente seleziona l'opzione "Cerca dati fornitore"

Flusso d'eventi:

1. Il sistema mostra il form per la ricerca dei dati del fornitore
2. L'utente inserisce i parametri della ricerca nel form;
3. L'utente avvia la ricerca selezionando l'opzione "Cerca";
4. Il sistema mostra i risultati corrispondenti ai criteri di ricerca impostati
5. L'utente seleziona il fornitore desiderato;

Condizione d'uscita:

L'utente conferma la selezione

Visualizzazione dati fornitore	Attori partecipanti: Responsabile magazzino Condizione di ingresso: L'utente seleziona l'opzione "Apri dati fornitore" Flusso di eventi: 1. L'utente seleziona il fornitore di cui desidera visualizzare i dati 2. Il sistema legge i dati relativi al cliente dal DBMS tramite il caso d'uso "Wrapper" Condizione di uscita: Il sistema mostra all'utente i dati del fornitore
Visualizzazione lista fornitori	Attori partecipanti: Responsabile magazzino Condizione di ingresso: L'utente seleziona l'opzione "visualizza lista fornitori"

	Flusso di eventi: 1. Il sistema preleva i dati dal DBMS tramite il caso d'uso "wrapper" Condizione di uscita: Il sistema mostra la lista dei fornitori
Visualizzazione distinta di produzione	Attori partecipanti: Responsabile magazzino Condizione di ingresso: L'utente seleziona l'opzione "Apri distinta " Condizione d'uscita: Il sistema mostra un form contenente la distinta di produzione
Autenticazione Responsabile magazzino	Attori partecipanti: Responsabile magazino Condizione di ingresso: Il responsabile magazzino avvia il sistema Flusso di eventi: 1. Il sistema mostra un form per l'inserimento dei dati per l'autenticazione 2. Il responsabile magazzino inserisce username e password e conferma 3. Il sistema verifica i dati utilizzando il caso d'uso "GestioneAutenticazione" Condizione di uscita: Il sistema avvia il programma di gestione consumi e fornitori

Modifica distinta di produzione

Attori partecipanti:

Responsabile magazzino

Condizione di ingresso:

L'utente seleziona l'opzione "modifica distinta di produzione

Flusso di eventi:

1. Il sistema mostra un form contenente la lista di tutti i componenti disponibili e la lista dei componentio inseriti nella distinta con le relative quantità
2. L'utente modifica componenti e relative quantità

Condizione di uscita:

Il sistema inserisce i dati nel DBMS tramite il caso d'uso "wrapper"

Inserisci ordine materiali

Attori coinvolti:

Responsabile magazzino

Condizione di ingresso:

L'utente seleziona un ordine materiali tramite il caso d'uso "Creazione ordine materiali"

Flusso di eventi:

1. L'utente seleziona l'opzione "salva ordine"

Condizione di uscita:

Il sistema inserisce nel DBMS l'ordine selezionato tramite il caso d'uso "wrapper"

Inserimento nuovo componente

Attori partecipanti:

Responsabile magazzino

Condizione di ingresso:

L'utente seleziona " Nuovo componente"

Flusso di eventi:

1. Il sistema mostra il form di inserimento dei dati
2. L'utente seleziona il fornitore dalla lista visualizzata tramite il caso d'uso "Visualizza lista fornitori"
3. L'utente riempie il form inserendo nome e descrizione del componente
4. L'utente conferma l'inserimento del componente

Condizione di uscita:

Il componente viene inserito nel DBMS utilizzando il caso d'uso "Wrapper"

Creazione ordine materiali	Attori partecipanti: Responsabile magazzino Condizione di ingresso: L'utente seleziona l'opzione "Crea ordini" Flusso di eventi: 1. Il sistema mostra all'utente un form contenente gli ordini per tutti i fornitori 2. Il sistema preleva la schedulazione mediante il caso d'uso "Visualizza schedulazione" 3. L'utente seleziona un fornitore dalla lista visualizzata tramite il caso d'uso "Visualizza lista fornitori" Condizione di uscita: Il sistema visualizza l'ordine relativo al fornitore selezionato

Stampa ordine materiali

Attori partecipanti:

Responsabile magazino

Condizione di ingresso:

L'utente seleziona un ordine materiali tramite il caso d'uso "Creazione ordine materiali"

Flusso di eventi:

1. L'utente seleziona l'opzione "anteprima di stampa"
2. Il sistemamostra l'anteprima di stampa dell'ordine selezionato
3. L'utente seleziona l'opzione "stampa"

Condizione di uscita:

Il sistema procede on la stampa dell'ordine selezionato

Modifica componente

Attori partecipanti:

Responsabile magazzino

Condizione di ingresso:

L'utente seleziona l'opzione "Visualizza lista componenti" e seleziona un componente dalla lista.

Flusso di eventi:

1. Il sistema mostra un form con i dati del componente
2. Il sistema mostra la lista dei fornitori disponibili tramite il caso d'uso "Visualizza dati fornitore"
3. L'utente seleziona il nuovo fornitore
4. L'attore modifica i dati del componente

Condizione di uscita:

Il sistema modifica i dati del componente nel DBMS tramite il caso d'uso "Wrapper"

Visualizza ordini materiali

Attori partecipanti:

Responsabile magazzino

Condizione di ingresso:

L'utente seleziona visualizza lista ordini

Flusso di eventi:

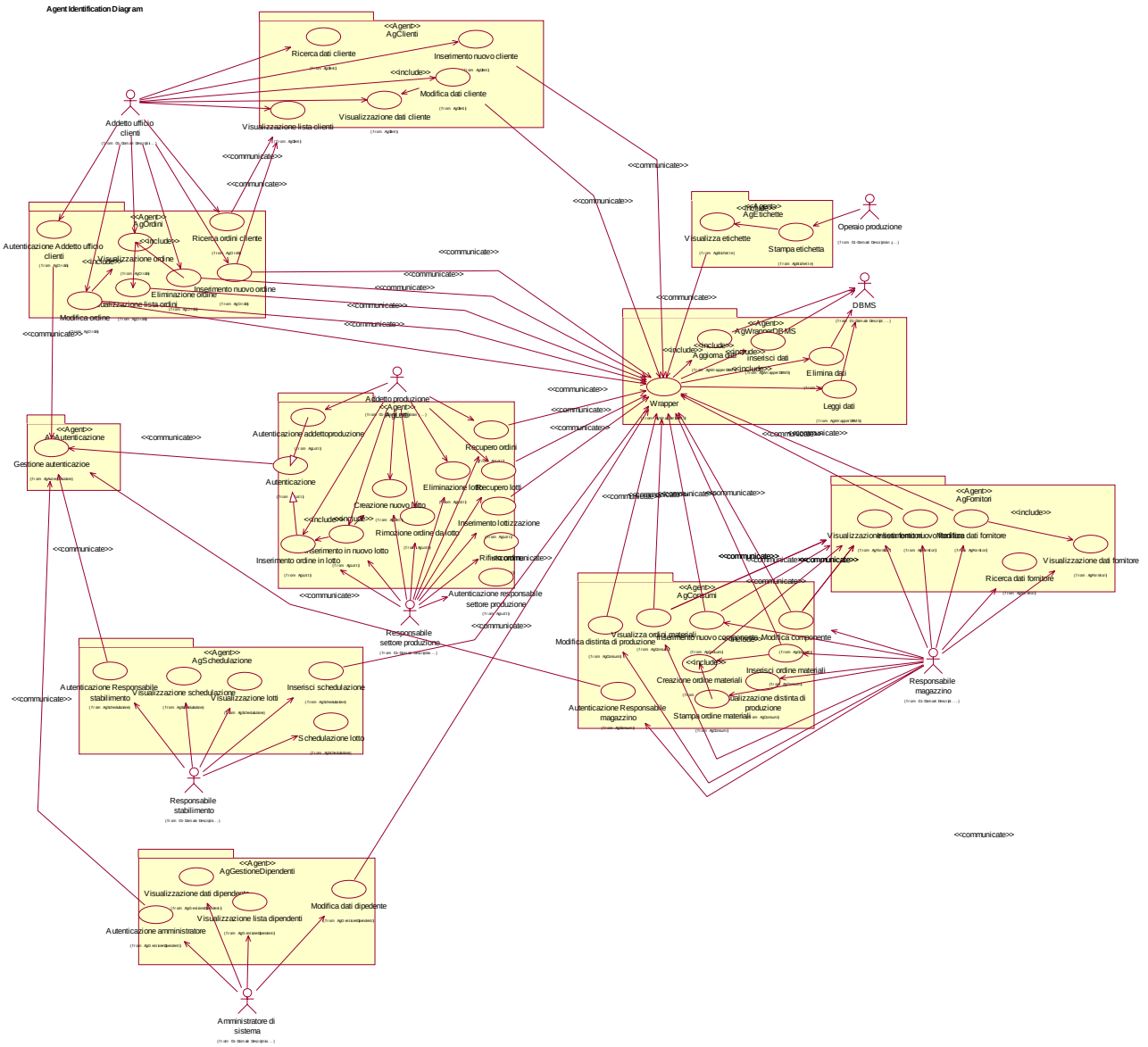
1. Il sistema mostra la lista dei fornitori tramite il caso d’uso “Visualizza lista fornitori”
2. . L'utente seleziona il fornitore

Condizione di uscita:

Il sistema visualizza la lista degli ordini effettuati dal fornitore selezionato.

3.2 Agent Identification phase

Questo diagramma è creato automaticamente dal 'Domain Description'. Ogni package defnisce le funzionzalità di uno specifico agente. Il nome del package è il nome dell'agente. Le entità esterne che che interagiscono con il sistema sono rappresentate come attori.

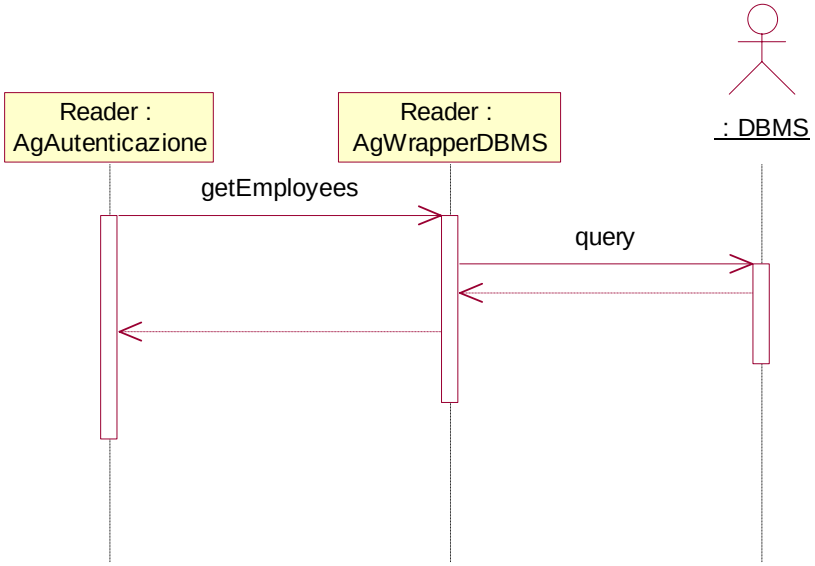


3.3 Role Identification phase

L'identificazione dei ruoli viene effettuta attraverso una serie di diagrammi di sequenza che specificano gli scenari più importanti dei casi d'uso dell'Agent Identification.

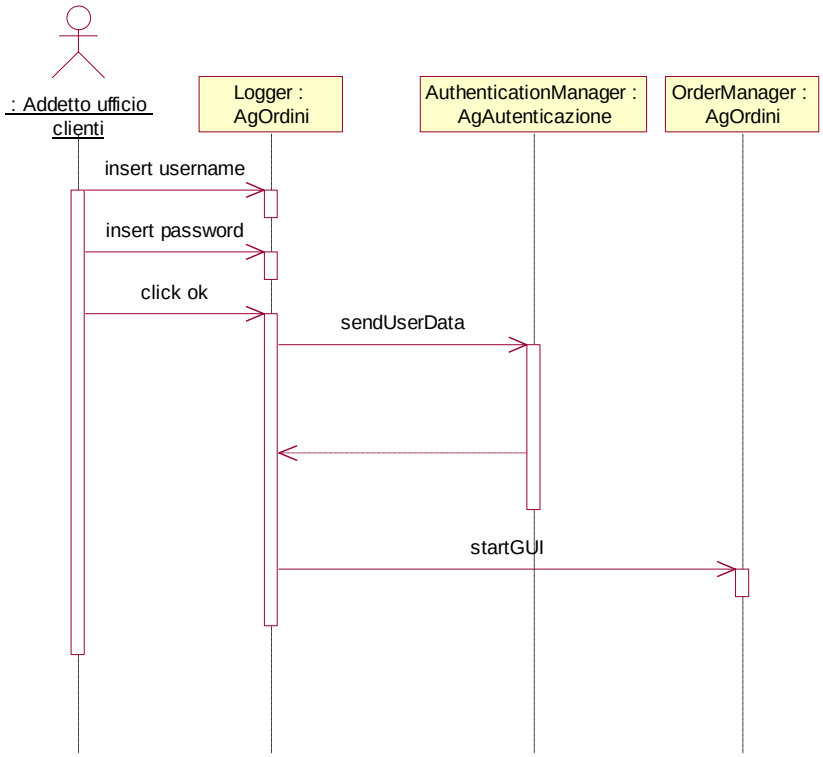
3.3.1 AgAutenticazione-Wrapper

All'avvio del sistema l'agente autenticazione recupera dal database i dati relativi agli impiegati che verranno utilizzati per la gestione delle autenticazioni. L'agente autenticazione, nel ruolo di reader, manda un messaggio al wrapper chiedendo la lista degli impiegati. Il wrapper, nel ruolo di reader, comunica col DBMS che gli restituisce la lista degli impiegati. Il wrapper restituisce tale lista all'agente autenticazione.



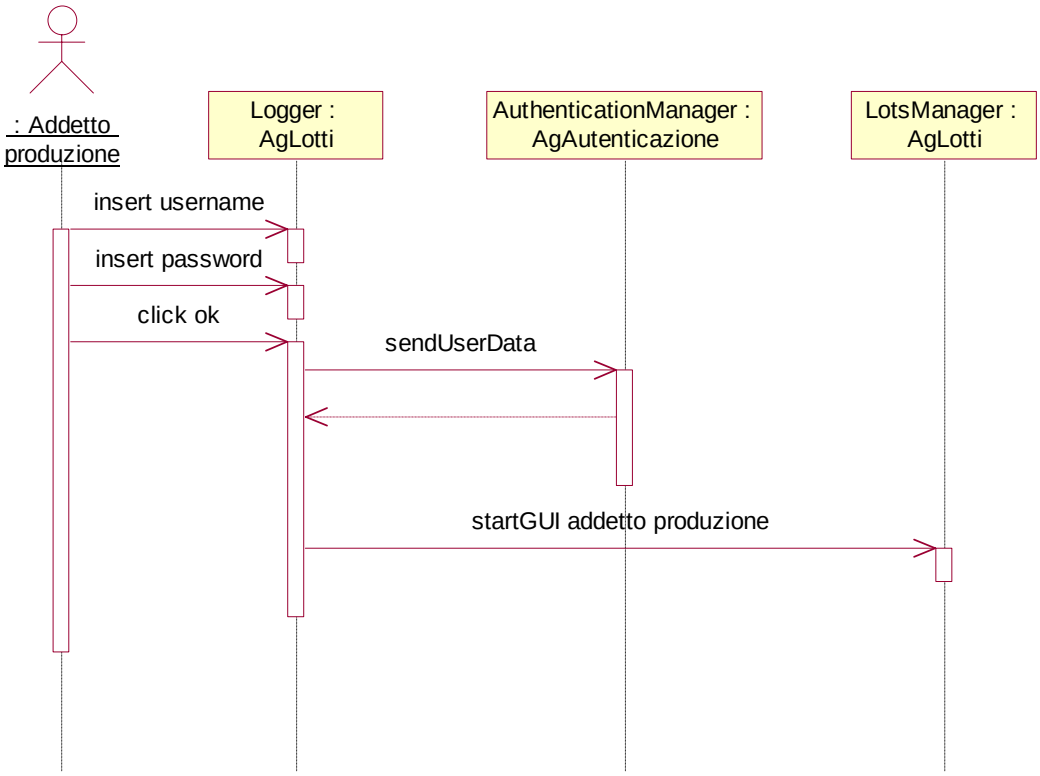
3.3.2 Autenticazione Addetto ufficio clienti

L'addetto ufficio clienti avvia l'agente ordini. L'agente, nel ruolo di logger, mostra un form per l'inserimento dei dati di autenticazione per accedere al programma di gestione degli ordini e dei clienti. L'addetto ufficio clienti inserisce username e password. L'agente ordini invia i dati dell'utente all'agente autenticazione che, nel ruolo di authentication manager, verifica la presenza di username e password all'interno della lista che ha in memoria, e restituisce il ruolo corrispondente ai dati che ha ricevuto. L'agente ordini controlla che il ruolo sia quello di addetto ufficio clienti e, in caso affermativo, avvia la gestione degli ordini e clienti cambiando il suo ruolo in quello di order manager.



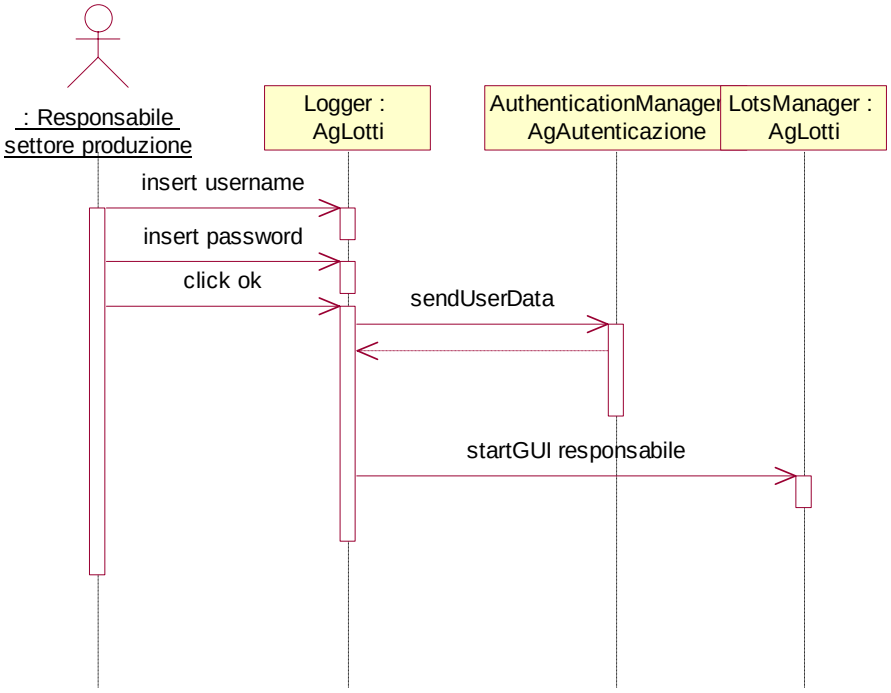
3.3.3 Autenticazione Addetto produzione

L'addetto produzione avvia l'agente lotti. L'agente, nel ruolo di logger, mostra un form per l'inserimento dei dati di autenticazione per accedere al programma di gestione dei lotti. L'addetto produzione inserisce username e password. L'agente lotti invia i dati dell'utente all'agente autenticazione che, nel ruolo di authentication manager, verifica la presenza di username e password all'interno della lista che ha in memoria, e restituisce il ruolo corrispondente ai dati che ha ricevuto. L'agente lotti controlla che il ruolo sia quello di addetto produzione e, in caso affermativo, avvia la gestione dei lotti, attivando la restrizione dei permessi per l'addetto produzione, cambiando il suo ruolo in quello di lots manager.



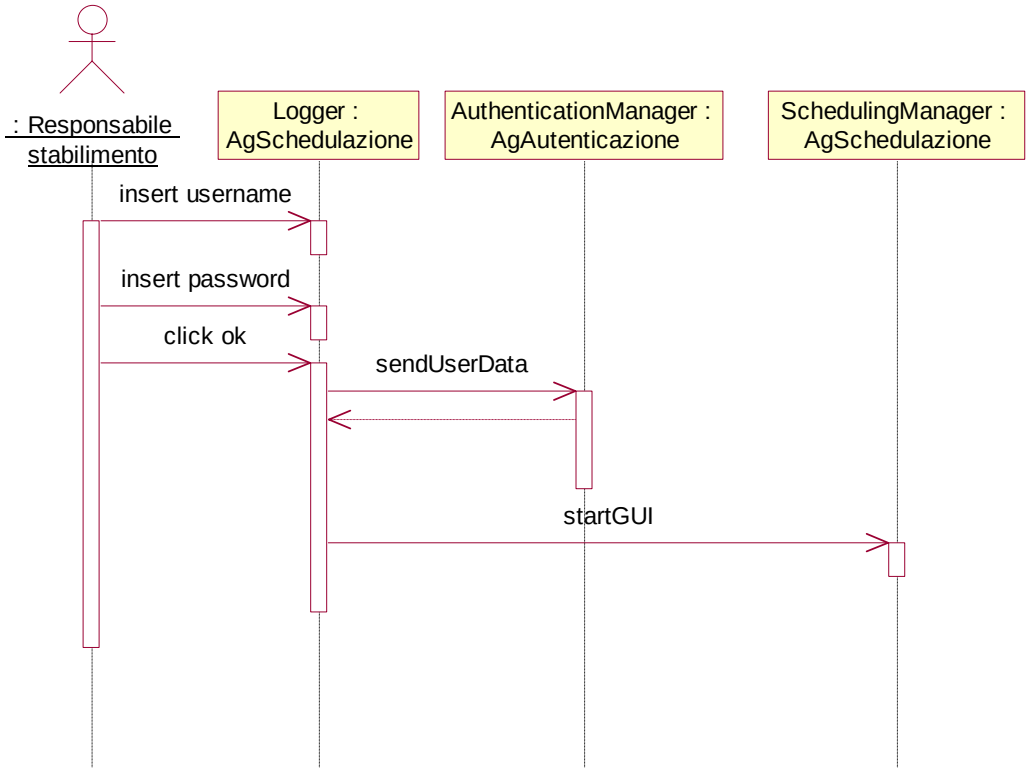
3.3.4 Autenticazione responsabile settore produzione

Il responsabile settore produzione avvia l'agente lotti. L'agente, nel ruolo di logger, mostra un form per l'inserimento dei dati di autenticazione per accedere al programma di gestione dei lotti. Il responsabile settore produzione inserisce username e password. L'agente lotti invia i dati dell'utente all'agente autenticazione che, nel ruolo di authentication manager, verifica la presenza di username e password all'interno della lista che ha in memoria, e restituisce il ruolo corrispondente ai dati che ha ricevuto. L'agente lotti controlla che il ruolo sia quello di responsabile settere produzione e, in caso affermativo, avvia la gestione dei lotti, attivando tutti i permessi per il responsabile settore produzione, cambiando il suo ruolo in quello di lots manager.



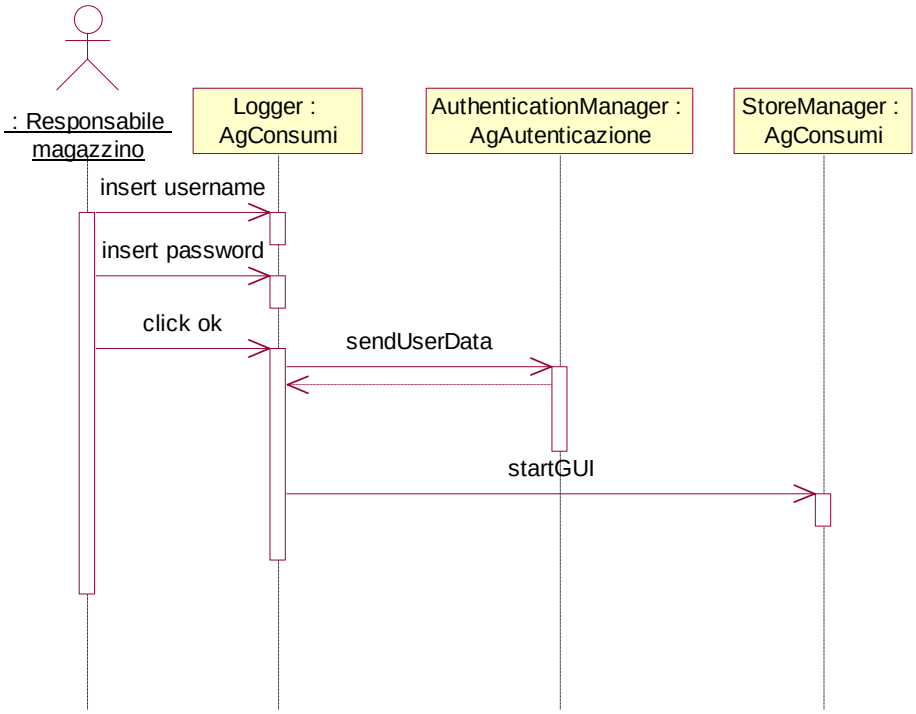
3.3.5 Autenticazione Responsabile stabilimento

Il responsabile stabilimento avvia l'agente schedulazione. L'agente, nel ruolo di logger, mostra un form per l'inserimento dei dati di autenticazione per accedere al programma di gestione della schedulazione. Il responsabile stabilimento inserisce username e password. L'agente schedulazione invia i dati dell'utente all'agente autenticazione che, nel ruolo di authentication manager, verifica la presenza di username e password all'interno della lista che ha in memoria, e restituisce il ruolo corrispondente ai dati che ha ricevuto. L'agente schedulazione controlla che il ruolo sia quello di responsabile stabilimento e, in caso affermativo, avvia la gestione della schedulazione cambiando il suo ruolo in quello di scheduling manager.



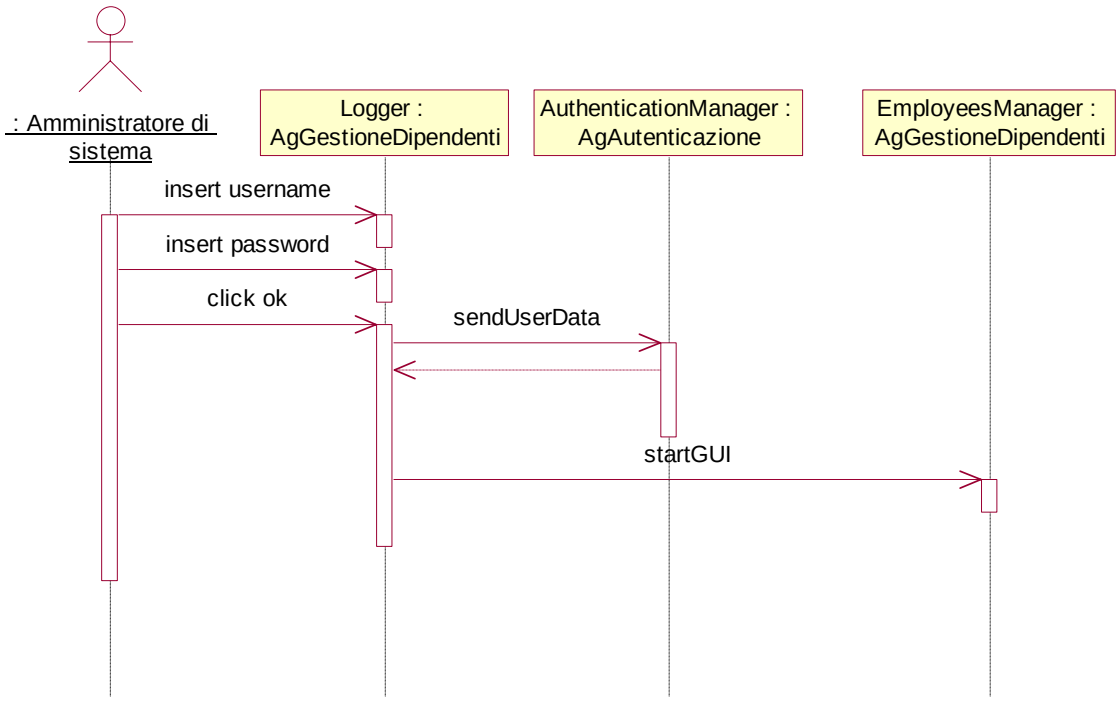
3.3.6 Autenticazione Responsabile magazzino

Il responsabile magazzino avvia l'agente consumi. L'agente, nel ruolo di logger, mostra un form per l'inserimento dei dati di autenticazione per accedere al programma di gestione dei consumi. Il responsabile magazzino inserisce username e password. L'agente consumi invia i dati dell'utente all'agente autenticazione che, nel ruolo di authentication manager, verifica la presenza di username e password all'interno della lista che ha in memoria, e restituisce il ruolo corrispondente ai dati che ha ricevuto. L'agente consumi controlla che il ruolo sia quello di responsabile magazzino e, in caso affermativo, avvia la gestione dei consumi cambiando il suo ruolo in quello di store manager.



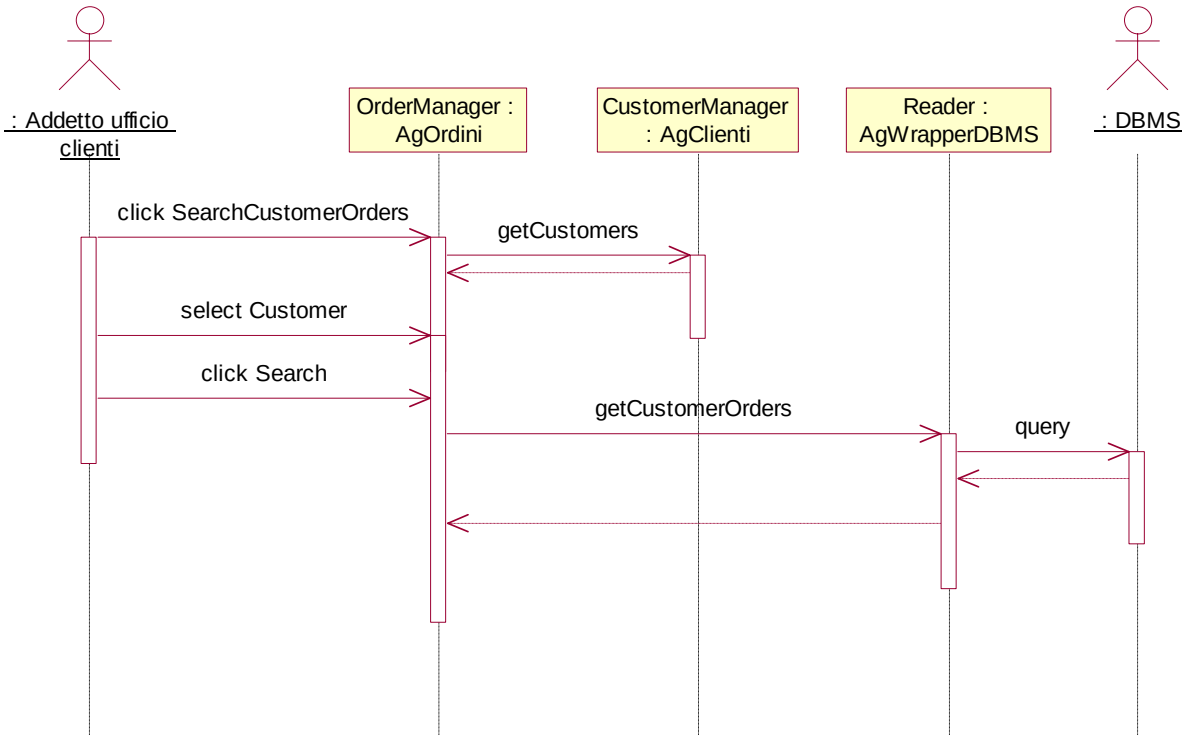
3.3.7 Autenticazione Amministratore di sistema

L'amministratore di sistema avvia l'agente gestione dipendenti. L'agente, nel ruolo di logger, mostra un form per l'inserimento dei dati di autenticazione per accedere al programma di gestione dei dipendenti. L'amministratore di sistema inserisce username e password. L'agente gestione dipendenti invia i dati dell'utente all'agente autenticazione che, nel ruolo di authentication manager, verifica la presenza di username e password all'interno della lista che ha in memoria, e restituisce il ruolo corrispondente ai dati che ha ricevuto. L'agente gestione dipendenti controlla che il ruolo sia quello di amministratore di sistema e, in caso affermativo, avvia la gestione dei dipendenti cambiando il suo ruolo in quello di employees manager.



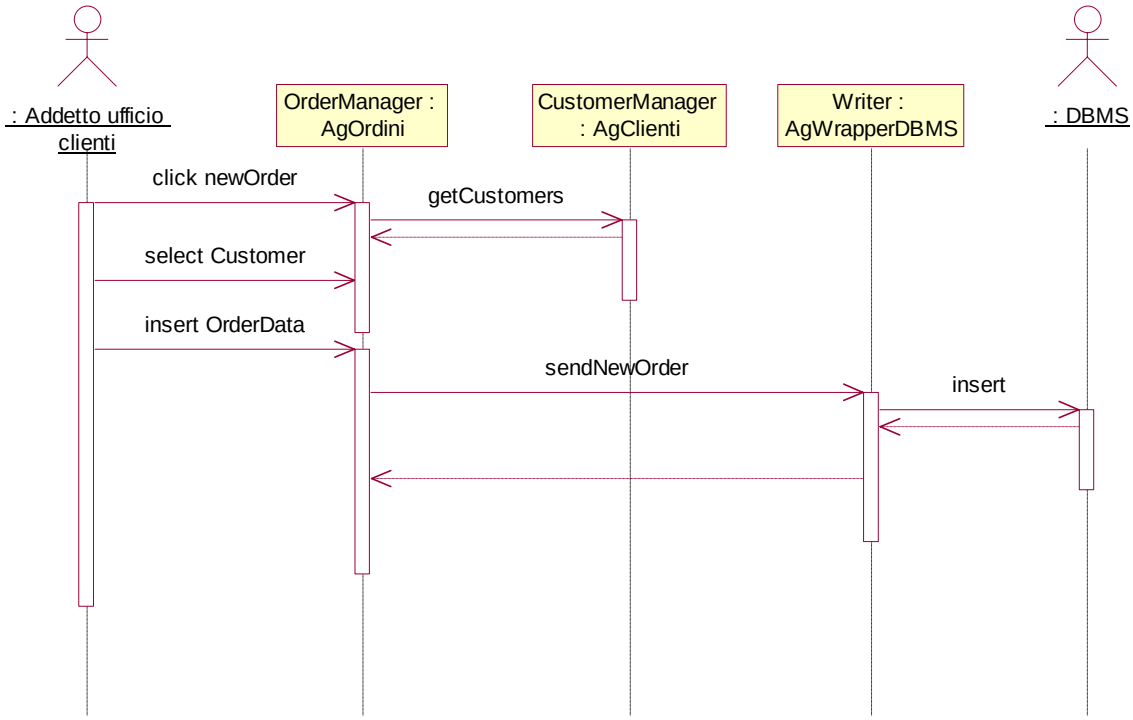
3.3.8 Ricerca ordini cliente

L'addetto ufficio clienti vuole visualizzare tutti gli ordini effettuati da un cliente. Seleziona l'opzione 'cerca ordini cliente' cliccando su un pulsante dell'interfaccia dell'agente ordini; quest'ultimo, nel ruolo di order manager, manda un messaggio all'agente clienti richiedendo la lista dei clienti. L'agente clienti, nel ruolo di customer manager, restituisce all'agente ordini la lista dei clienti. L'agente ordini mostra un form per la ricerca dgli ordini. L'addetto uffifio clienti seleziona il cliente dasiderato e seleziona l'opzione 'cerca'. L'agente ordini manda un messaggio al wrapper il quale, nel ruolo di reader, comunica con il DMBS e restituisce all'agente ordini gli ordini effettuati dal cliente. Infine l'agente ordini mostra all'addetto ufficio clienti la lista desiderata.



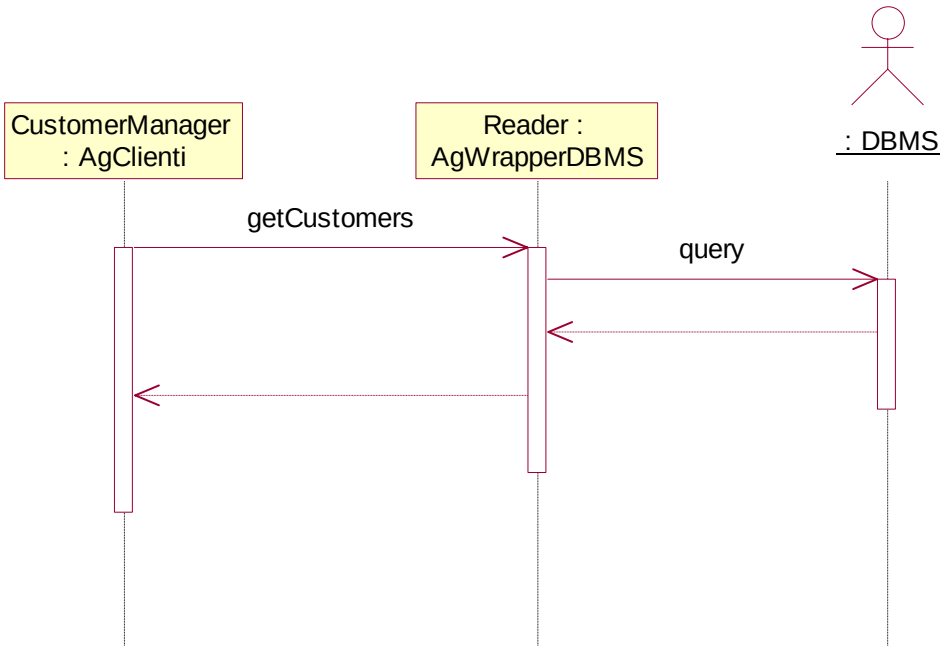
3.3.9 Inserimento nuovo ordine

L'addetto ufficio clienti desidera inserire un nuovo ordine. Seleziona l'opzione 'nuovo ordine' dall'interfaccia dell'agente ordini; quest'ultimo, nel ruolo di order manager, manda un messaggio all'agente clienti richiedendo la lista dei clienti. L'agente clienti, nel ruolo di customer manager, restituisce all'agente ordini la lista dei clienti. L'agente ordini mostra un form per l'inserimento di un nuovo ordine. L'addetto ufficio clienti seleziona il cliente a cui l'ordine si riferisce, scegli quantità e data di consegna per i due modelli di biciclette ordinate (non necessariamente entrambi i modelli) e conferma l'inserimento. L'agente ordini invia un messaggio contenente il nuovo ordine al wrapper il quale, nel ruolo di writer, comunica al DBMS l'inserimento dei dati.



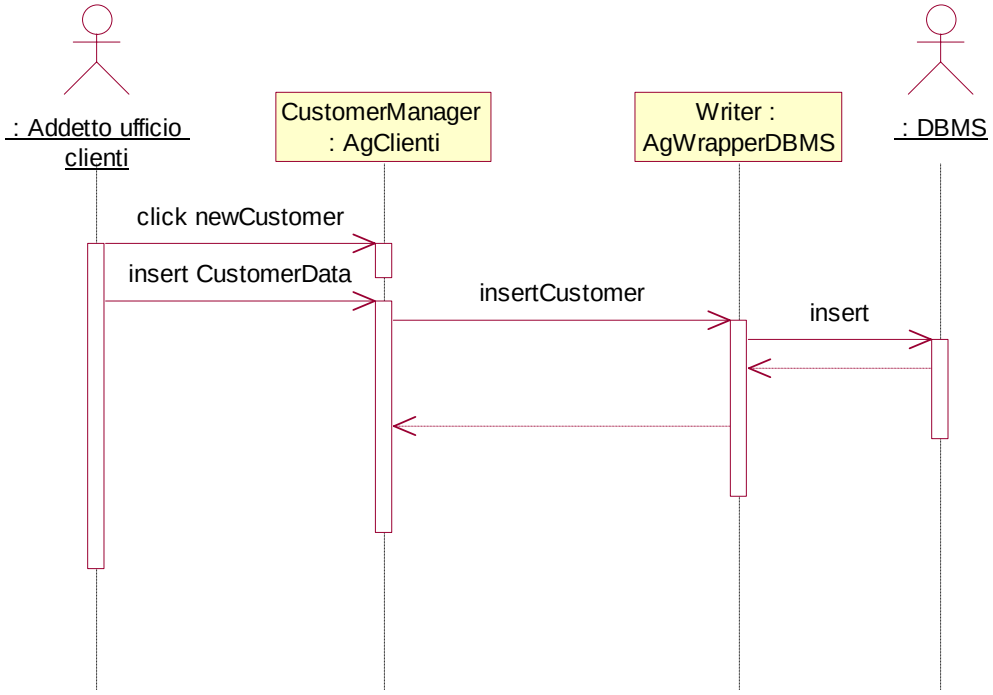
3.3.10 Avvio agente clienti

All’avvio del sistema l’agente clienti recupera dal database i dati relativi ai clienti. L’agente clienti, nel ruolo di cutomer manager, manda un messaggio al wrapper chiedendo la lista dei clienti. Il wrapper, nel ruolo di reader, comunica col DBMS che gli restituisce la lista dei clienti richiesta. Il wrapper restituisce tale lista all’agente clienti.



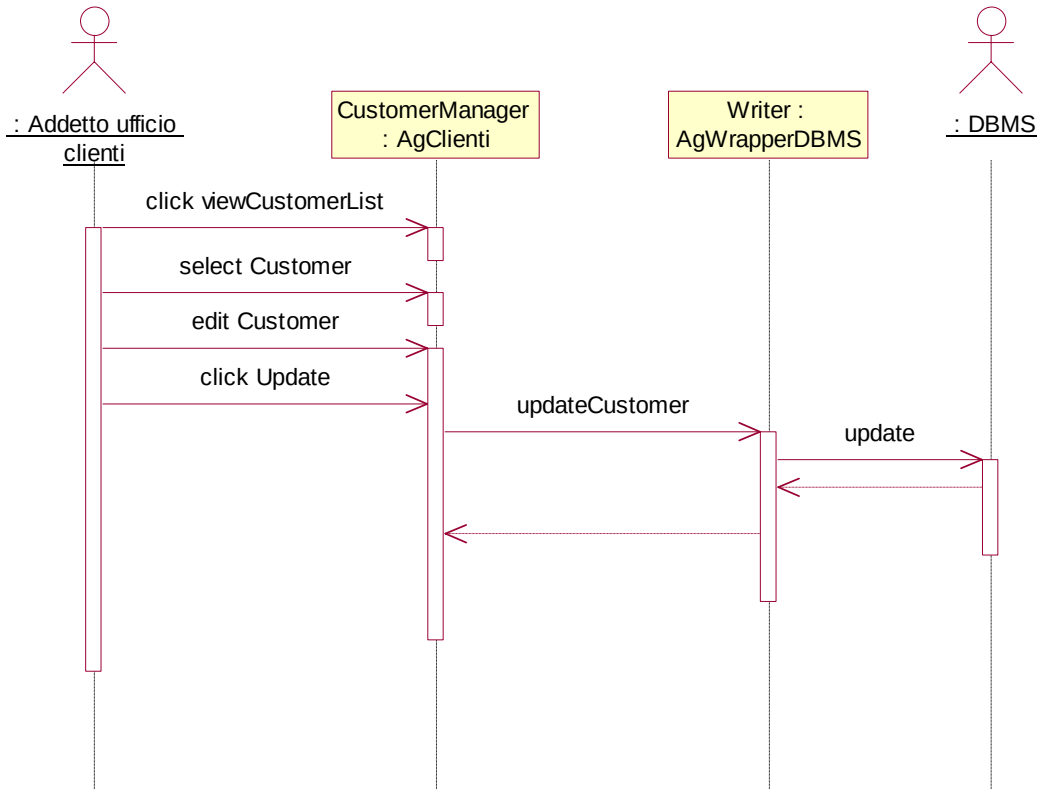
3.3.11 Inserimento nuovo cliente

L'addetto ufficio clienti desidera inserire un nuovo cliente. Seleziona l'opzione 'nuovo cliente' dall'interfaccia dell'agente clienti. L'agente clienti, nel ruolo di customer manager, mostra un form per l'inserimento di un nuovo cliente. L'addetto ufficio clienti inserisce i dati relativi al cliente (nome, cognome, compagnia, indirizzo, e-mail, telefono, partita Iva) e conferma l'inserimento. L'agente clienti invia un messaggio contenente il nuovo cliente al wrapper il quale, nel ruolo di writer, comunica al DBMS l'inserimento dei dati.



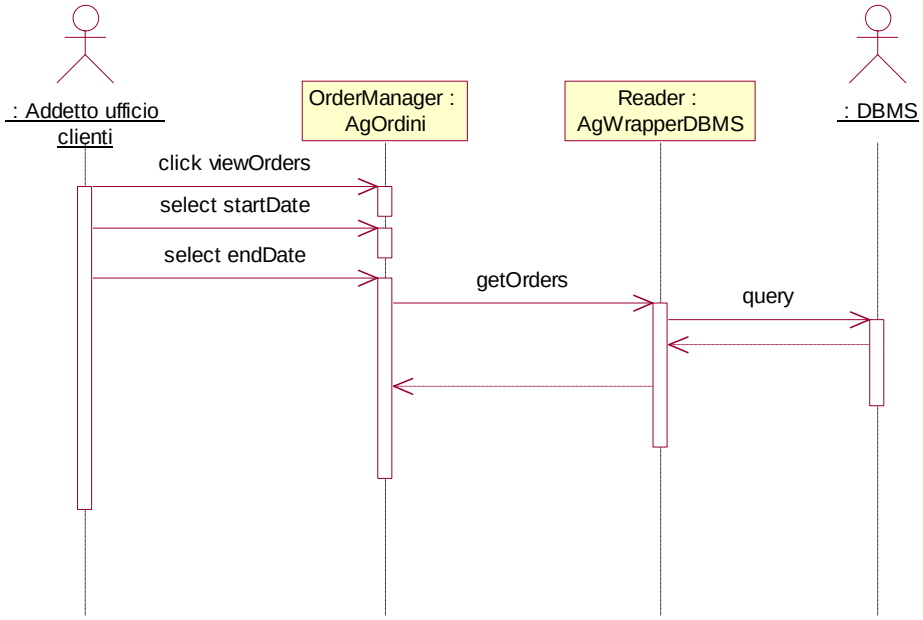
3.3.12 Modifica dati cliente

L'addetto ufficio clienti desidera modificare i dati di un cliente. Seleziona l'opzione 'visualizza lista clienti' dall'interfaccia dell'agente clienti. L'agente clienti, nel ruolo di customer manager, mostra un form contenente la lista dei clienti. L'addetto ufficio clienti seleziona il cliente di cui desidera modificare i dati e seleziona l'opzione 'apri'. L'agente clienti mostra un form contenente i dati relativi al cliente selezionato. L'addetto ufficio clienti effettua le modifiche desiderate e conferma l'aggiornamento dei dati. L'agente clienti invia un messaggio contenente i dati aggiornati sul cliente al wrapper il quale, nel ruolo di writer, comunica al DBMS l'aggiornamento dei dati.



3.3.13 Visualizza lista ordini

L’addetto ufficio clienti desidera visualizzare gli ordini che sono stati effettuati in un certo intervallo di tempo. Seleziona l’opzione ‘visualizza ordini’ dall’interfaccia dell’agente ordini. L’agente ordini mostra un form per la scelta dell’intervallo di tempo desiderato. L’addetto ufficio clienti sceglie la data iniziale, la data finale e conferma. L’agente ordini manda un messaggio al wrapper il quale, nel ruolo di reader, effettua una query al DBMS prelevando gli ordini desiderati. Il wrapper restituisce tale lista all’agente ordini. L’agente ordini visualizza la lista degli ordini all’addetto ufficio clienti.

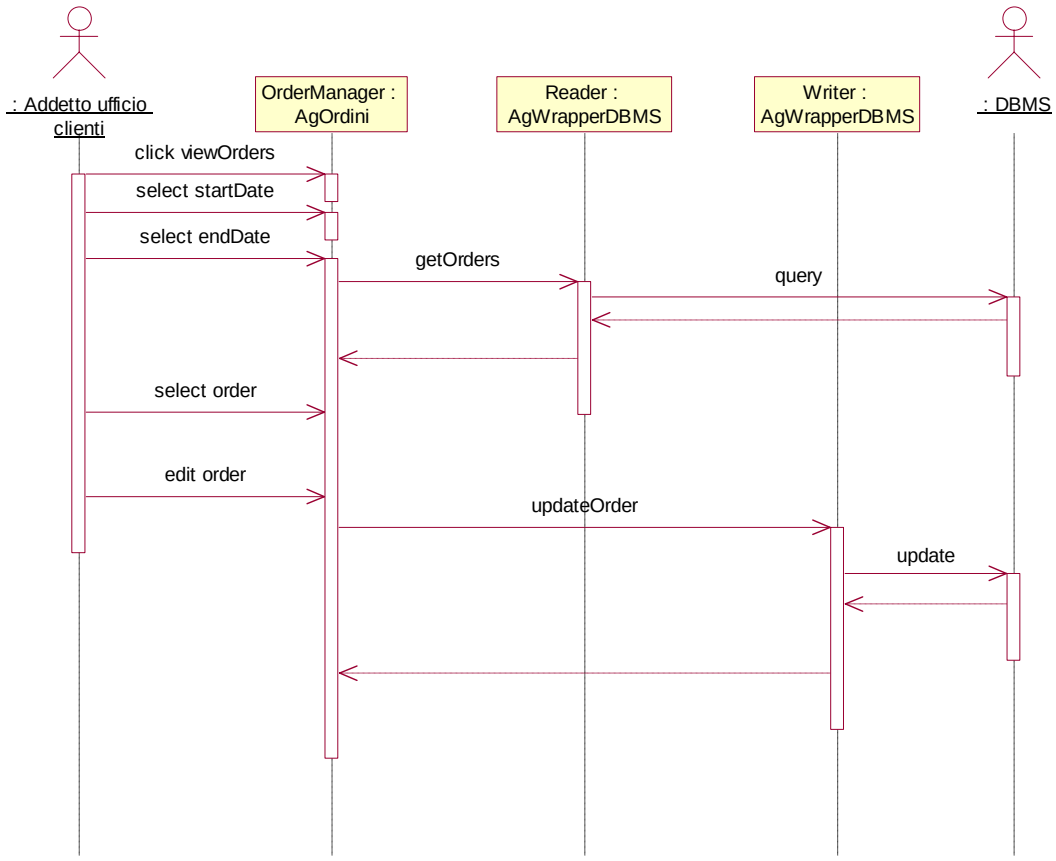


3.3.14 Modifica ordine

L'addetto ufficio clienti desidera modificare i dati di un ordine. Seleziona l'opzione 'visualizza ordini' dall'interfaccia dell'agente ordini. L'agente ordini mostra un form per la scelta dell'intervallo di tempo desiderato. L'addetto ufficio clienti sceglie la data iniziale, la data finale e conferma. L'agente ordini manda un messaggio al wrapper il quale, nel ruolo di reader, effettua una query al DBMS prelevando gli ordini desiderati. Il wrapper restituisce tale lista all'agente ordini. L'agente ordini visualizza la lista degli ordini all'addetto ufficio clienti.

L'addetto ufficio clienti seleziona l'ordine di cui desidera modificare i dati e seleziona l'opzione 'apri'. L'agente ordini mostra un form contenente i dati relativi all'ordine selezionato. L'addetto ufficio clienti effettua le modifiche desiderate e conferma l'aggiornamento dei dati.

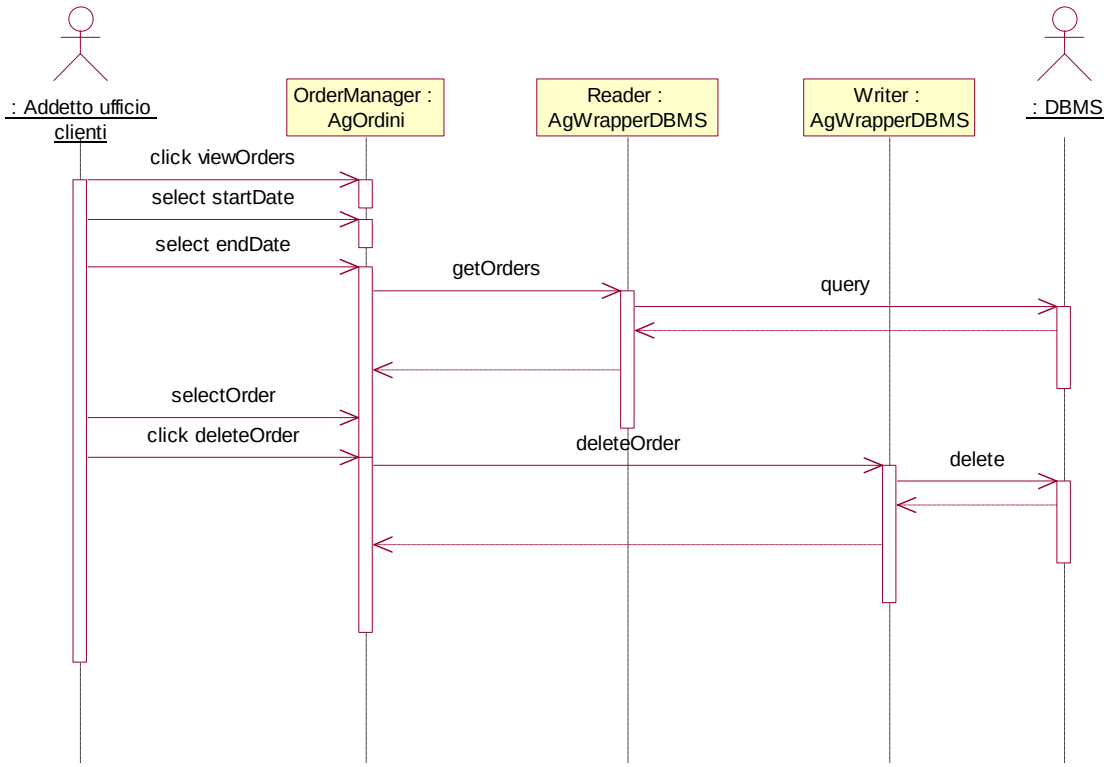
L'agente ordini controlla che l'ordine non sia già stato schedulato e, in tal caso, invia un messaggio contenente i dati aggiornati sul cliente al wrapper il quale, nel ruolo di writer, comunica al DBMS l'aggiornamento dei dati.



3.3.15 Cancella ordine

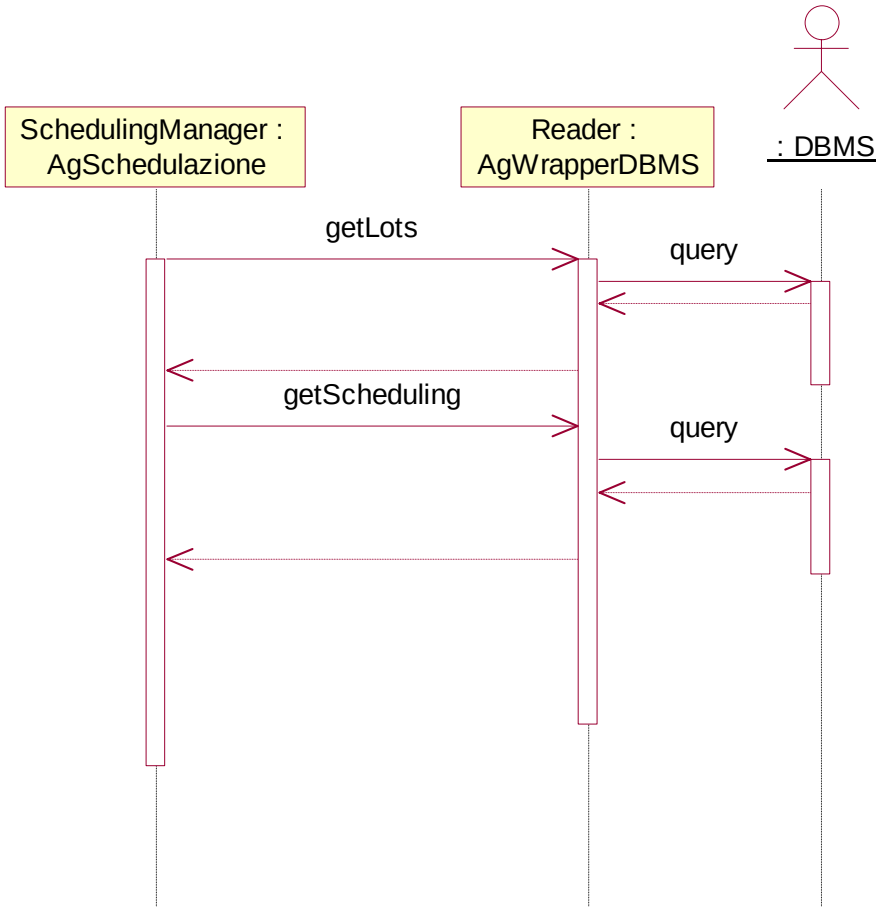
L'addetto ufficio clienti desidera eliminare un ordine. Seleziona l'opzione 'visualizza ordini' dall'interfaccia dell'agente ordini. L'agente ordini mostra un form per la scelta dell'intervallo di tempo desiderato. L'addetto ufficio clienti sceglie la data iniziale, la data finale e conferma. L'agente ordini manda un messaggio al wrapper il quale, nel ruolo di reader, effettua una query al DBMS prelevando gli ordini desiderati. Il wrapper restituisce tale lista all'agente ordini. L'agente ordini visualizza la lista degli ordini all'addetto ufficio clienti.

L'addetto ufficio clienti seleziona l'ordine che desidera eliminare e seleziona l'opzione 'elimina'. L'agente ordini mostra un form chiedendo la conferma dell'eliminazione. L'addetto ufficio clienti conferma l'eliminazione dell'ordine selezionato. L'agente ordini controlla che l'ordine non sia già stato schedulato e, in tal caso, invia un messaggio contenente l'ordine da eliminare al wrapper il quale, nel ruolo di writer, comunica al DBMS l'eliminazione dei dati.



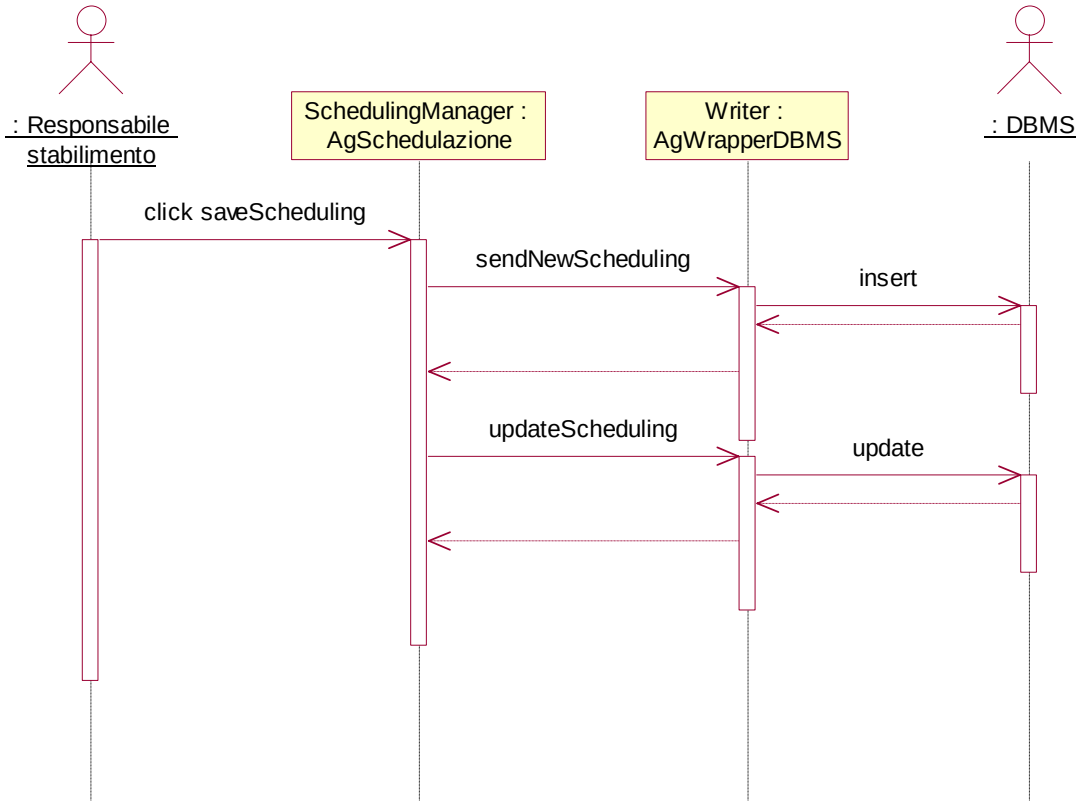
3.3.16 Avvio agente schedulazione

All'avvio del sistema l'agente schedulazione recupera dal database i lotti non ancora schedulati e la schedulazione già affettuata a partire da due settimane prima della data corrente. L'agente schedulazione, nel ruolo di scheduling manager, manda un messaggio al wrapper chiedendo i dati. Il wrapper, nel ruolo di reader, comunica col DBMS che gli restituisce la lista dei lotti e la schedulazione richiesti. Il wrapper restituisce queste liste liste all'agente schedulazione.



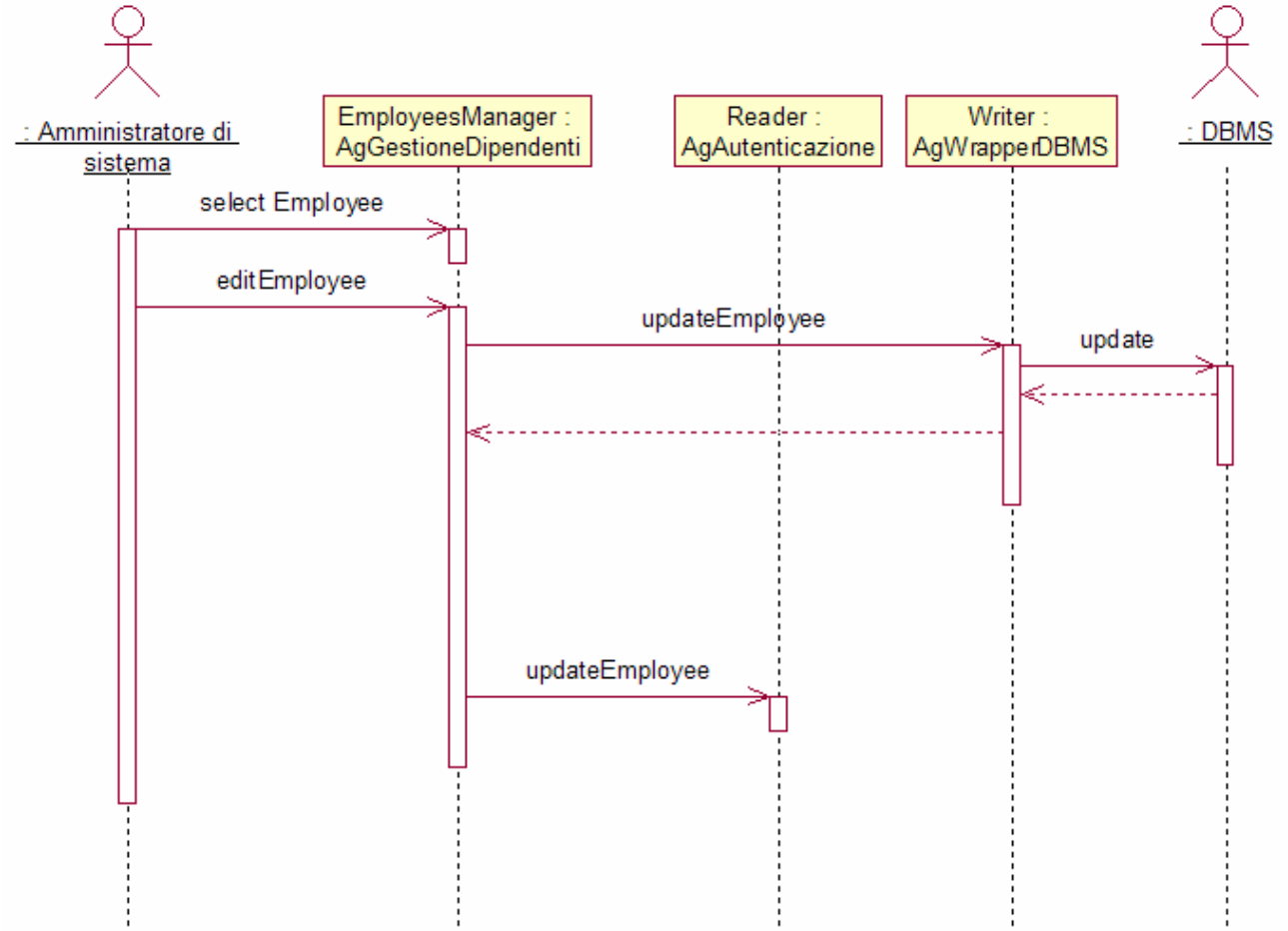
3.3.17 Inserisci schedulazione

Il responsabile stabilimento desisera inserire nell'archicio dell'azienda la schedulazione effettuata. Il responsabile stabiliemnto seleziona l'opzione 'salva schedulazione' dall'inetrfaccia dell'agente schedulazione. L'agente schedulazione, nel ruolo di scheduling manager, invia un messaggio al wrapper contenente la nuova schedulazione creata. Il wrapper comunica i dati al DBMS il quale li inserisce nel database. L'agente schedulazione, inoltre, invia al wrapper anche l'aggiornamento di schedulazioni eventualmente modificate dal responsabile stabilimento. Il wrapper comunica i dati al DBMS che provvede ad agiornare i dati nel database.



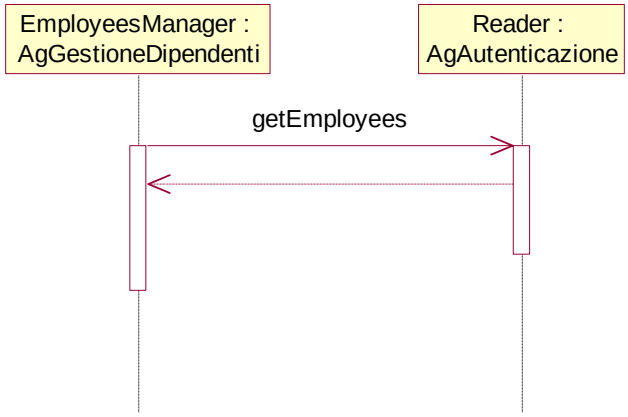
3.3.18 Modifica dati dipendenti

L'amministratore di sistema desidera modificare i dati di un dipendente. L'amministratore di sistema seleziona il dipendente di cui desidera modificare i dati e seleziona l'opzione 'apri'. L'agente gestione dipendenti, nel ruolo di emplyees manager, mostra un form contenente i dati relativi al dipendente selezionato. L'amministratore di sistema effettua le modifiche desiderate e conferma l'aggiornamento dei dati. L'agente gestione dipendenti invia un messaggio contenente i dati aggiornati sul dipendente al wrapper il quale, nel ruolo di writer, comunica al DBMS l'aggiornamento dei dati. L'agente gestione dipendenti invia i nuovi dati anche all'agente autenticazione.



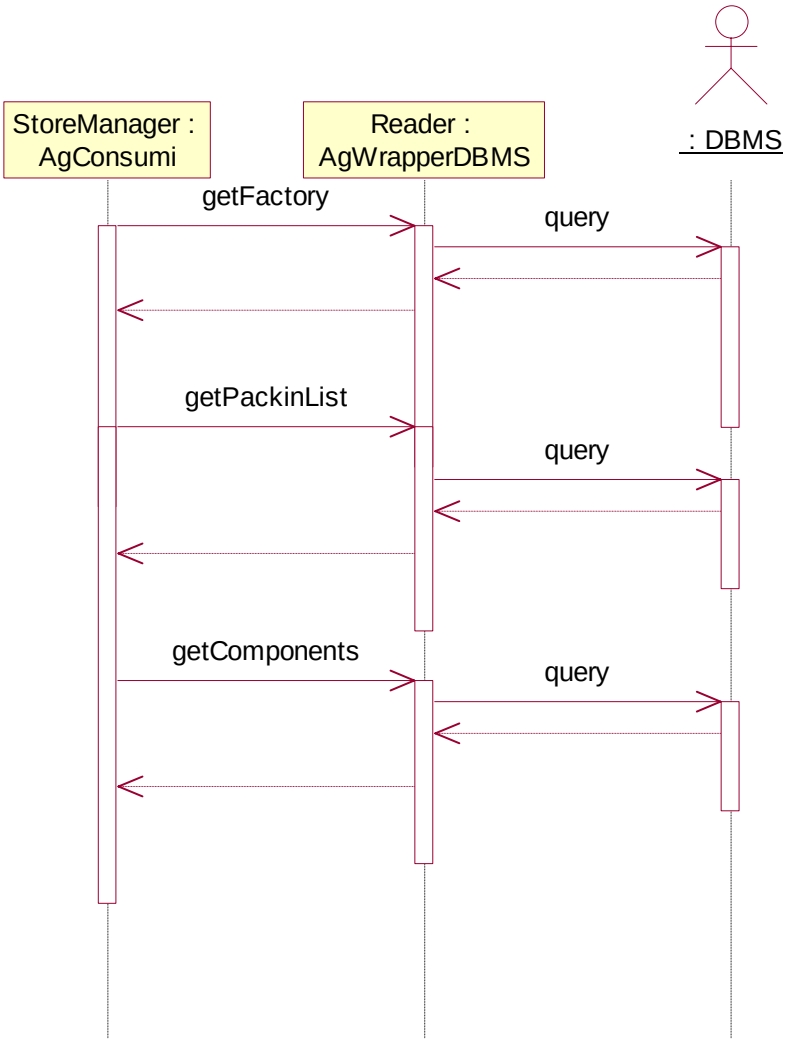
3.3.19 Avvio Agente gestione dipendenti

All’avvio del sistema l’agente gestione dipendenti, nel ruolo di emplyees manager, invia un messaggio all’agente autenticazione chiedendogli la lista dei dipendenti. L’agente autenticazione, nel ruolo di reader, invia tale lista all’agente gestione dipendenti.



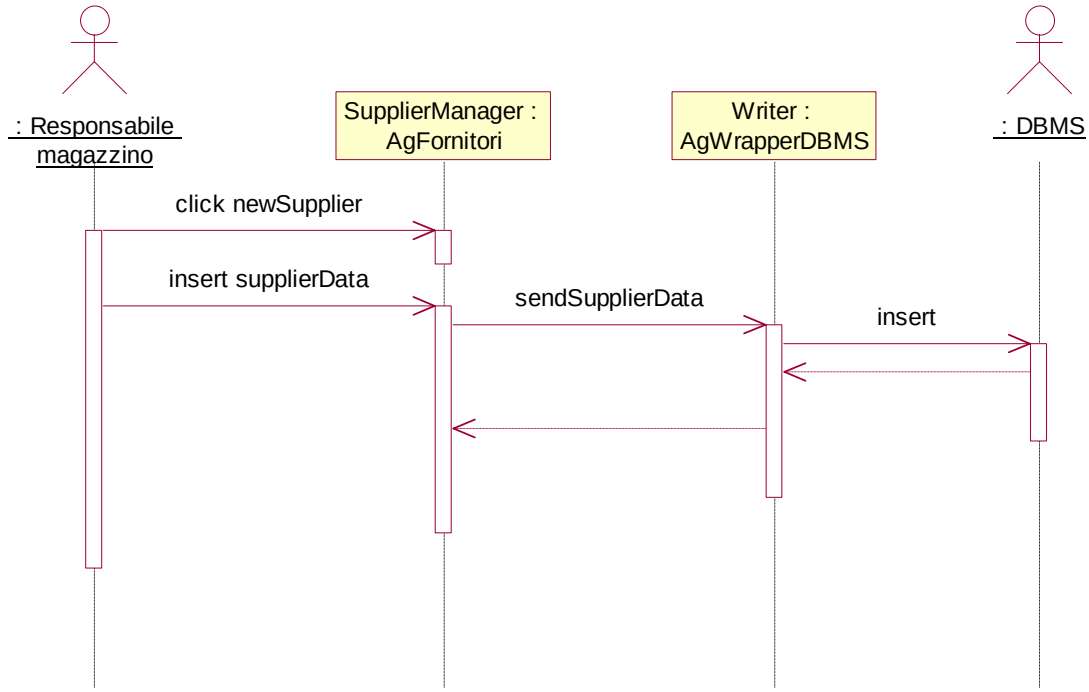
3.3.20 Avvio agente consumi

All'avvio del sistema l'agente consumi, nel ruolo di store manager, richiede al wrapper dei dati tuili all'approvvigionamento. Richiede i dati sullo stabilimento, la distinta di produzione del modello di bicicletta prodotto nello stabilimento e la lista deo componenti. Il wrapper, nel ruolo di reader, esegue una serie di query al DBMS il quale restituisce i dati richiesti al wrapper. A sua volta il wrapper restituisce i dati all'agente consumi.



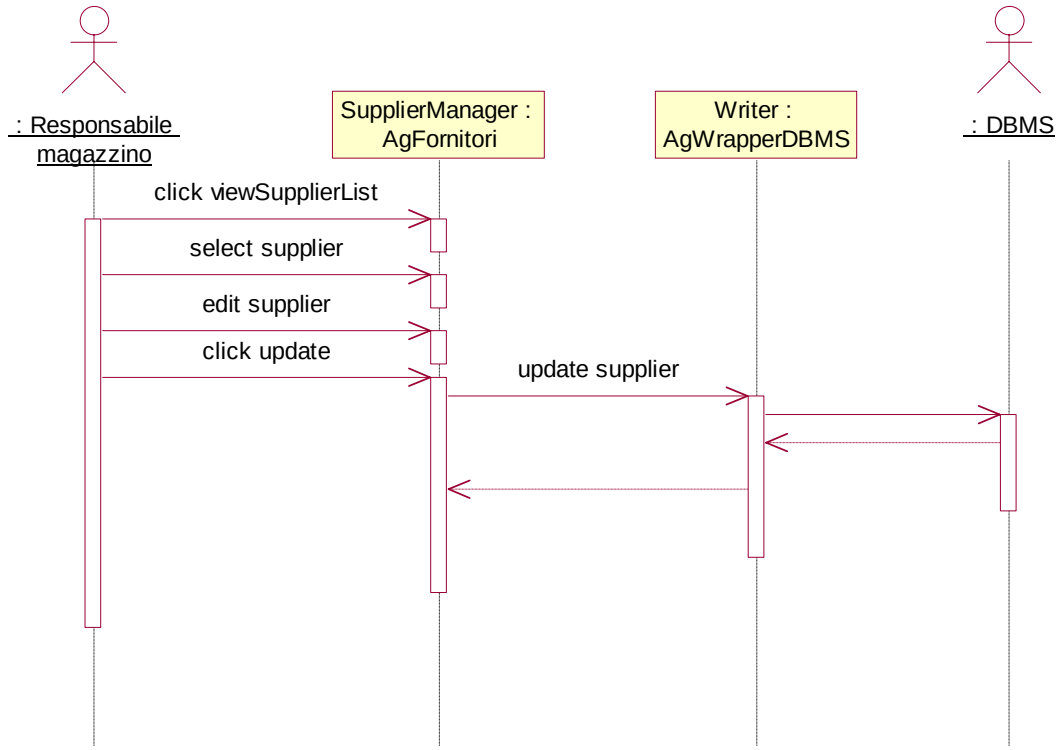
3.3.21 Inserimento nuovo fornitore

Il responsabile magazzino desidera inserire un nuovo fornitore. Seleziona l'opzione 'nuovo fornitore' dall'interfaccia dell'agente fornitori. L'agente fornitori, nel ruolo di supplier manager, mostra un form per l'inserimento di un nuovo fornitore. Il responsabile magazzino inserisce i dati relativi al fornitore (nome, cognome, compagnia, indirizzo, e-mail, telefono, partita Iva) e conferma l'inserimento. L'agente fornitori invia un messaggio contenente il nuovo fornitore al wrapper il quale, nel ruolo di writer, comunica al DBMS l'inserimento dei dati.



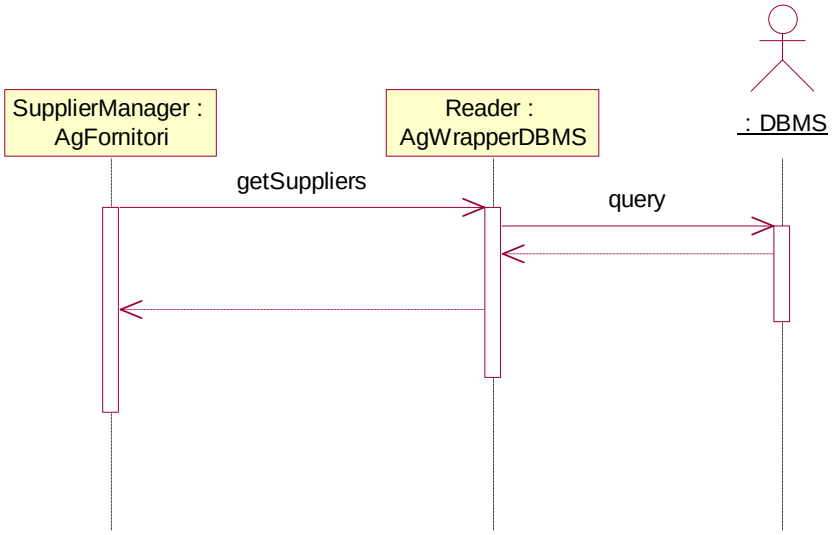
3.3.22 Modifica dati fornitore

Il responsabile magazzino desidera modificare i dati di un fornitore. Seleziona l'opzione 'visualizza lista fornitori dall'interfaccia dell'agente fornitori. L'agente fornitori, nel ruolo di supplier manager, mostra un form contenente la lista dei fornitori. Il responsabile magazzino seleziona il fornitore di cui desidera modificare i dati e seleziona l'opzione 'apri'. L'agente fornitori mostra un form contenente i dati relativi al fornitore selezionato. Il responsabile magazzino effettua le modifiche desiderate e conferma l'aggiornamento dei dati. L'agente fornitori invia un messaggio contenente i dati aggiornati sul fornitore al wrapper il quale, nel ruolo di writer, comunica al DBMS l'aggiornamento dei dati.



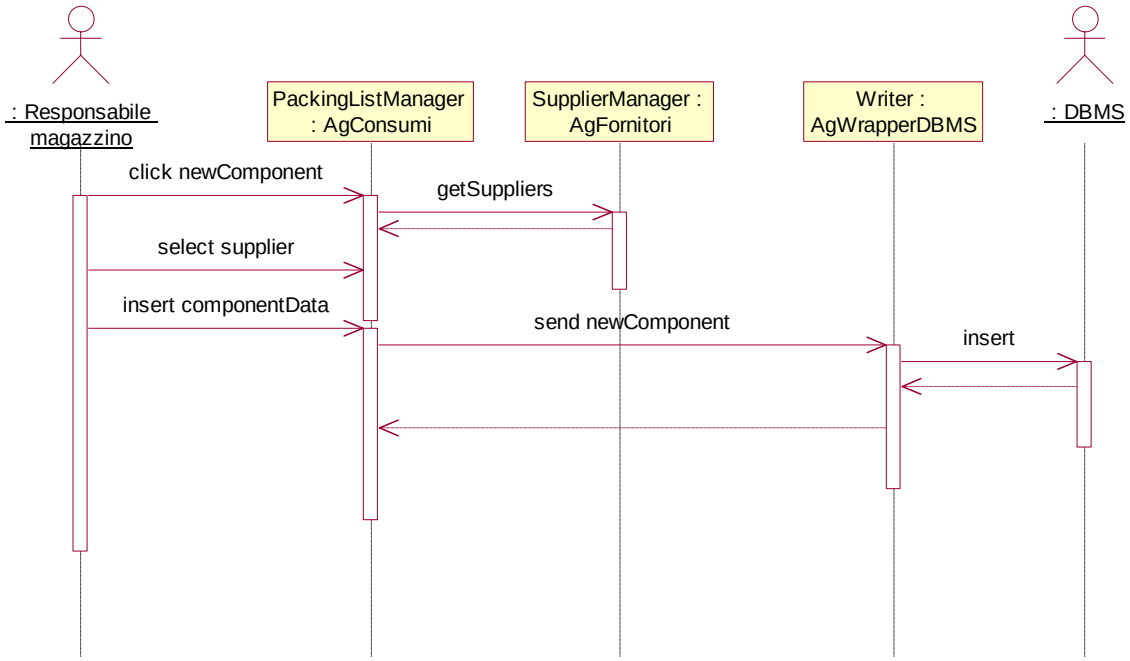
3.3.23 Avvio agente fornitori

All'avvio del sistema l'agente fornitori recupera dal database i dati relativi ai fornitori. L'agente fornitori, nel ruolo di supplier manager, manda un messaggio al wrapper chiedendo la lista dei fornitori. Il wrapper, nel ruolo di reader, comunica col DBMS che gli restituisce la lista dei fornitori richiesta. Il wrapper restituisce tale lista all'agente fornitori.



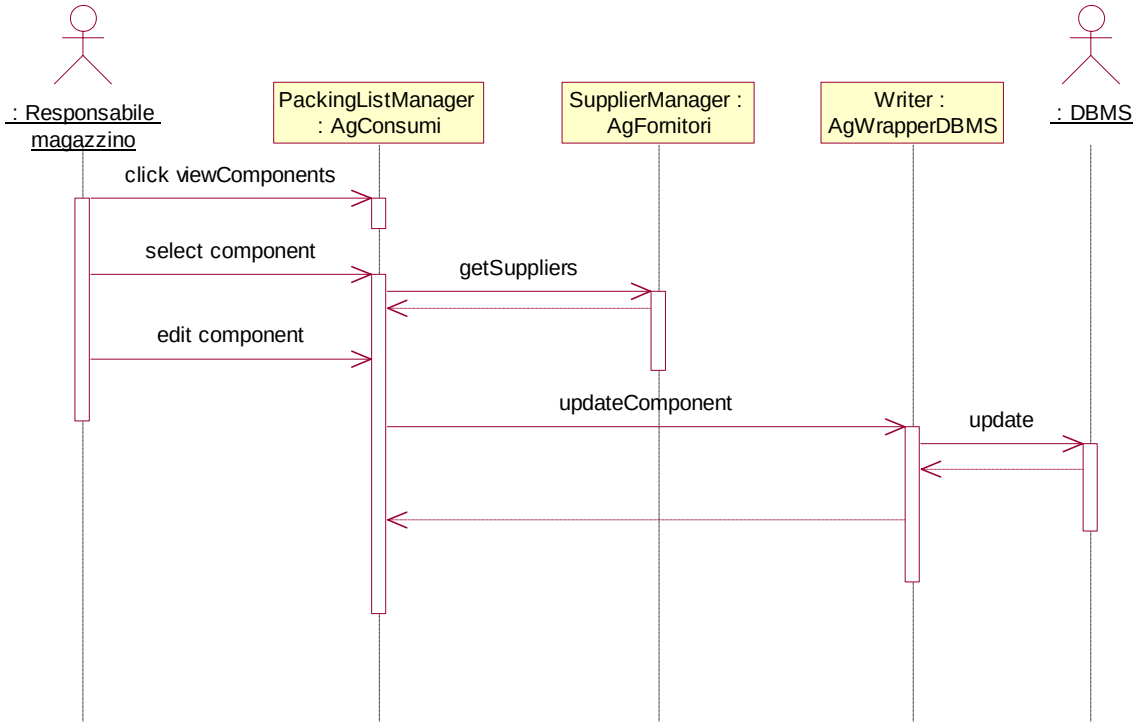
3.3.24 Inserimento nuovo componente

Il rersponsabile magazzino desidera inserire un nuovo componente. Seleziona l'opzione 'nuovo componente' dall'interfaccia dell'agente consumi. Quest'ultimo, nel ruolo di packing list manager, invia un messaggio all'agente fornitori richiedendo la lista dei fornitori dell'azinda. L'agente fornitori, nel ruolo di supplier manager, erstituisce all'agente consumi tale lista. L'agente fornitori mostra all'utente un form per l'inserimento del nuovo componente.



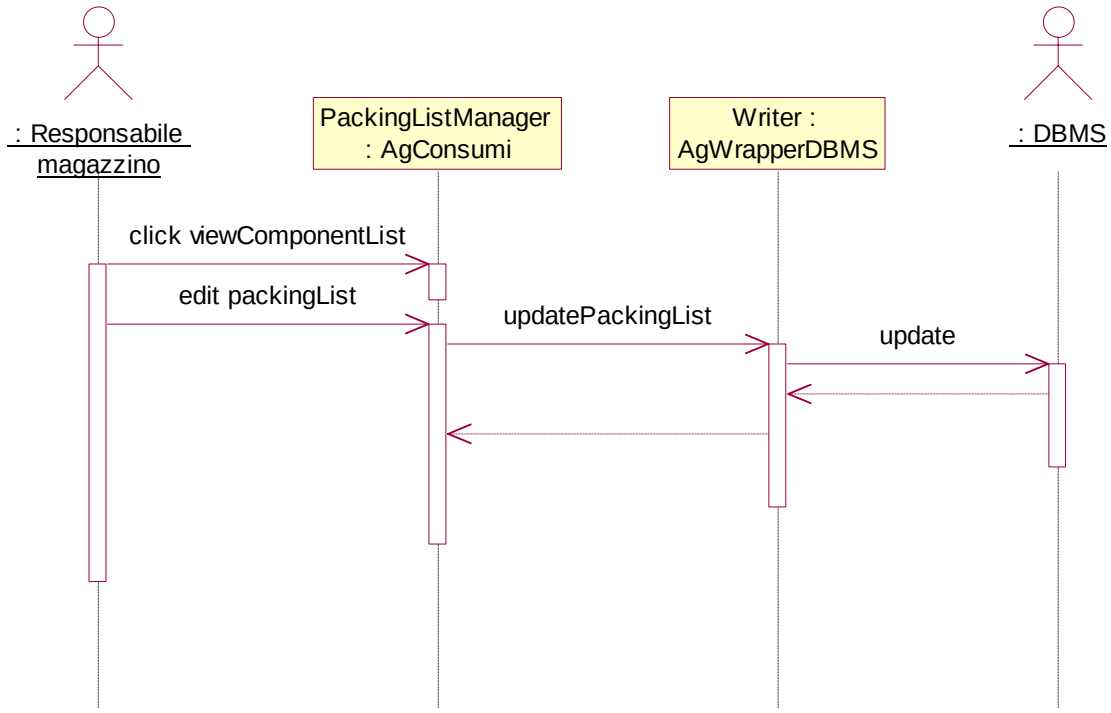
3.3.25 Modifica componente

Il reponsabile magazzino desidera modificare i dati di un componente. Seleziona l'opzione 'visualizza lista componenti' dall'interfaccia dell'agente consumi. Quest'ultimo, nel ruolo di packing list manager, invia un messaggio all'egente fornitori con la richiesta della lista dei fornitori. L'agent fornitori, nel ruolo di supplier manager, invia all'agente consumi la lista dei fornitori richiesta. L'agente consumi mostra all'utente la lista dei componenti. Il responsabile magazzino seleziona il componente da modificare e seleziona l'opzione 'apri'. L'agente consumi mostra all'utente un form contenente i dati del componente. Il responsabile magazzino modifica i dati del componente e seleziona l'opzione 'modifica'. L'agente consumi chiede conferma, l'utente conferma la modifica e l'agente consumi invia i dati da aggiornare al wrapper. Il wrapper, nel ruolo di writer, comunica al DBMS i dati da aggiornare.



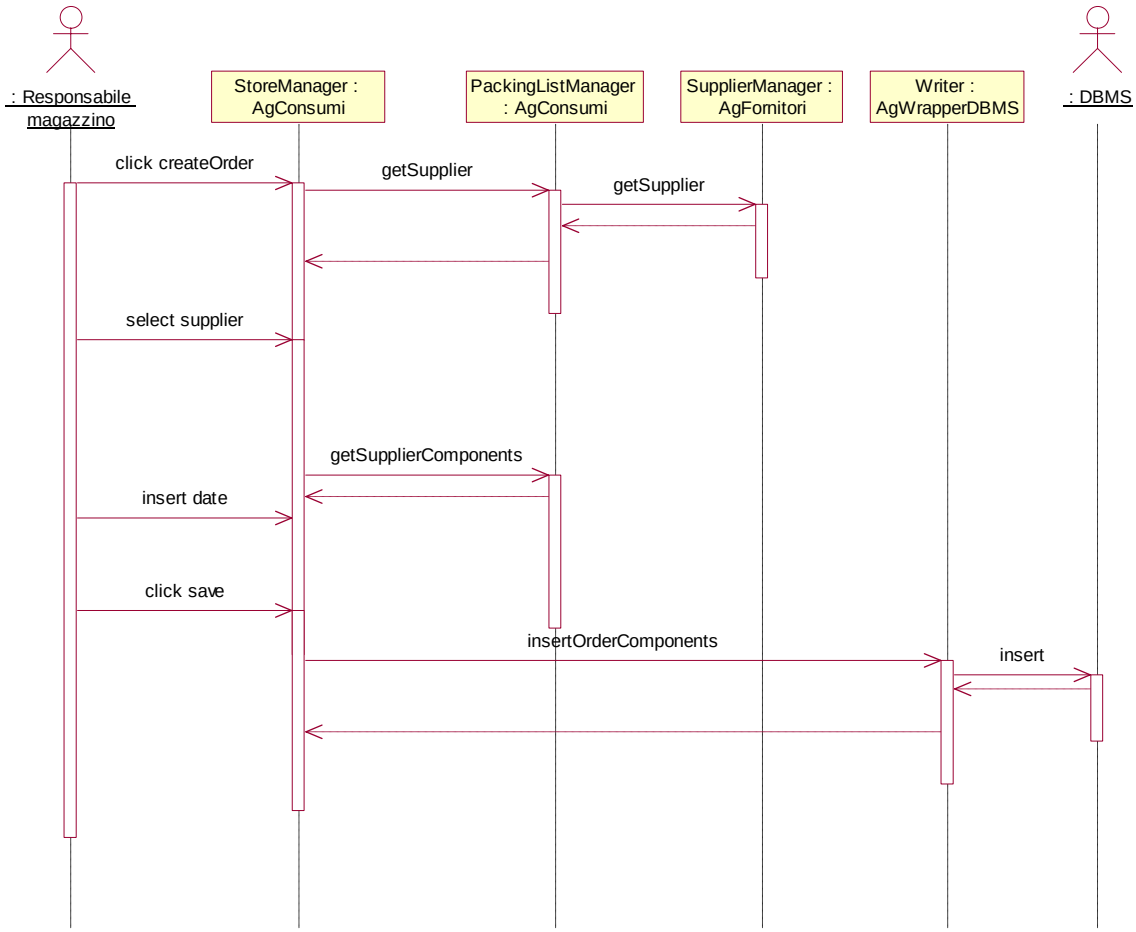
3.3.26 Modifica distinta di produzione

Il responsabile magazzino desidera modificare la distinta di produzione. Seleziona l'opzione 'visualizza lista componenti'. L'agente consumi, nel ruolo di packing list manager, mostra all'utente un form contenete la lista di tutti i componenti e la lista dei componenti presenti nella distinta di produzione con le relative quantità. Il responsabile magazzino effettua le modifiche desiderate e conferma. Il sistema chiede conferma delle modifiche effettuate. L'utente conferma nuovamente e l'agente consumi invia un messaggio al wrapper contenente i dati da aggiornare. Il wrapper comunica al DBMS i dati da aggiornare.



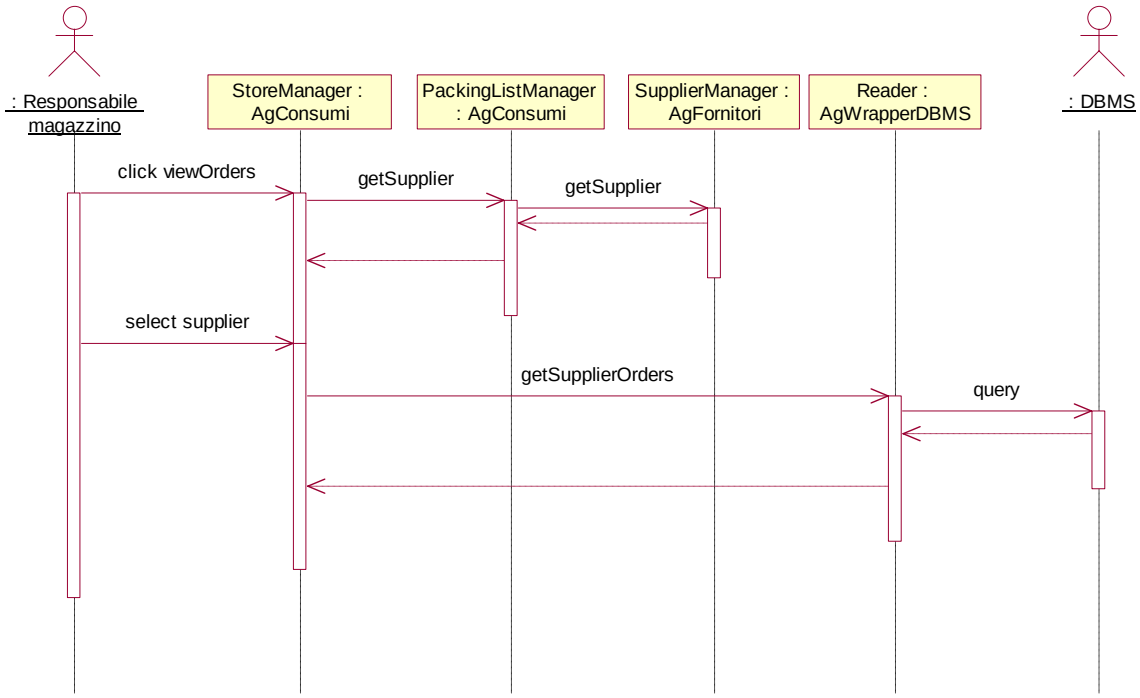
3.3.27 Inserimento nuovo ordine materie prime

Il responsabile magazzino desidera creare un ordine di componenti. Seleziona l'opzione 'nuovo ordine' dall'interfaccia dell'agente consumi. L'agente consumi, dal ruolo di store manager passa al ruolo di packinglist manager, da questo ruolo richiede all'agente fornitori la lista dei fornitori di componenti. L'agente fornitori invia la lista richiesta all'agente consumi. L'agente consumi torna al ruolo di store manager. L'agente consumi mostra al responsabile magazzino un form in cui è possibile selezionare un fornitore e scegliere la quantità di componenti da ordinare. Il responsabile magazzino sceglie, per ogni ordine, la data di consegna e le quantità di componenti da ordinare e seleziona l'opzione 'salva'. L'agente consumi invia i dati al wrapper che li invia al DBMS per l'inserimento.



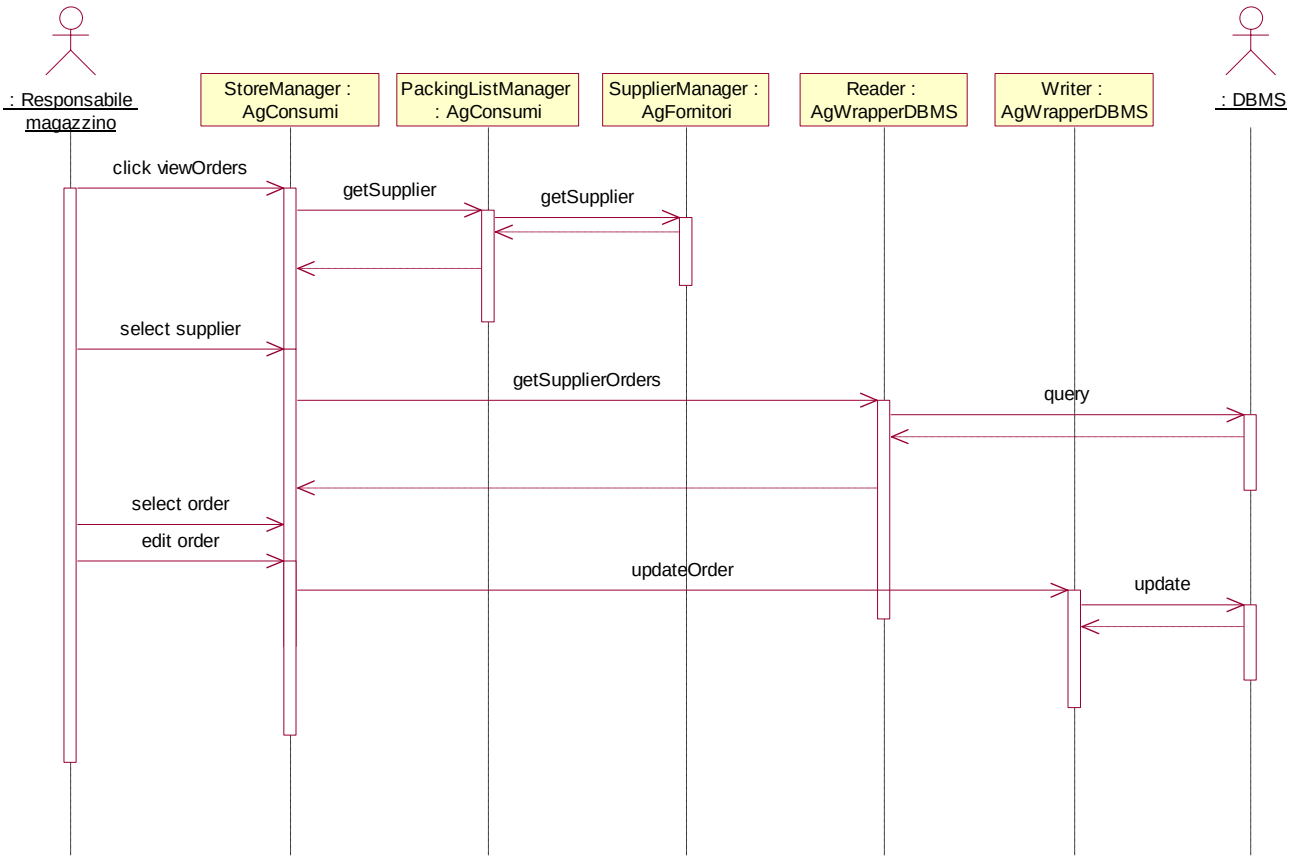
3.3.28 Visualizza ordini componenti

Il responsabile magazzino desidera visualizzare gli ordini di componenti effettuati per un certo fornitore. L'agente consumi richiede all'agente fornitori la lista dei fornitori dell'azienda. L'agente fornitori invia all'agente consumi la lista richiesta. L'agente consumi mostra all'utente la lista dei fornitori. Il responsabile magazzino seleziona il fornitore desiderato. L'agente consumi richiede al wrapper la lista degli ordini relativi al fornitore selezionato. Il wrapper effettua una quary al DBMS e restituisce la lista degli ordini all'agente consumi. L'agente consumi mostra all'utente un form contenente la lista degli ordini relativi al fornitore desiderato.



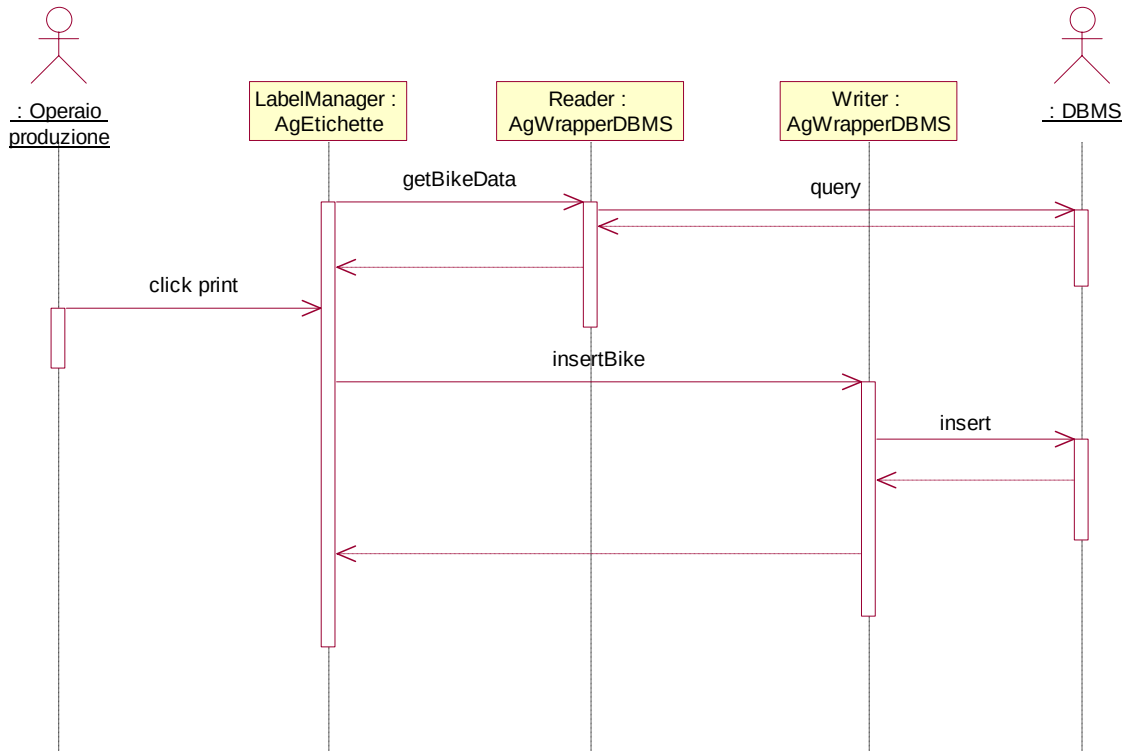
3.3.29 Modifica ordine componenti

Il responsabile magazzino desidera modificare un ordine componenti. L'utente seleziona l'opzione 'visualizza ordini componenti' dall'interfaccia dell'agente consumi. L'agente consumi richiede all'agente fornitori la lista dei fornitori di componenti dell'azienda. L'agente fornitori invia la lista dei fornitori all'agente consumi. L'agente consumi mostra all'utente un form contenente la lista dei fornitori. Il resopnsbile magazzino seleziona un fornitore. L'agente consumi richiede al wrapper la lista deglim ordini relativi al fornitore selezionato dall'utente. Il wrapper, nel ruolo di reader, esegue una query al DBMS e restituisce la lista degli ordini all'agente consumi. Quest'ultimo mostra all'utente un form contenete la lista degli ordini. L'utente seleziona un ordine e ne modifica il contenuto e seleziona l'opzione 'modifica'. Il sistema chiede conferma delle modifiche effettuate. Il responsabile magazzino conferma e l'agente consumi invia i dati da aggiornare al wrapper. Il wrapper, nel ruolo di writer, invia al DBMS i dati da aggiornare.



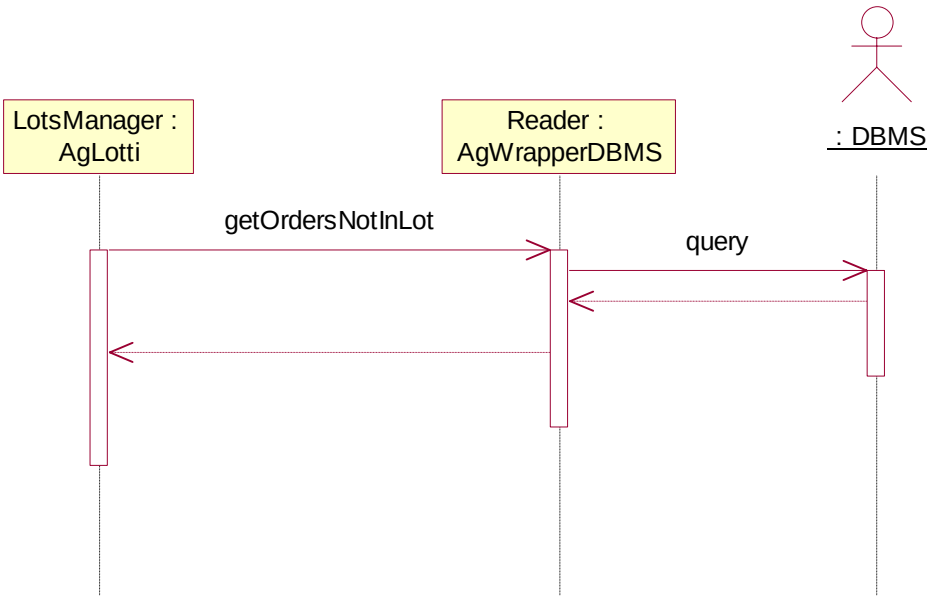
3.3.30 Creazione etichetta

All'avvio l'agente etichette, nel ruolo di label manager, chiede al wrapper i dati relativi alla produzione. Il wrapper, nel ruolo di reader, esegue una query al DBMS e restituisce i dati relativi alla produzione all'agente etichette. L'agente etichette mostra all'operaio produzione l'etichetta corrente da stampare. L'operaio produzione seleziona l'opzione 'stampa' dall'interfaccia dell'agente etichette. L'agente etichette invia al wrapper la bicicletta da inserire nell'archivio aell'azienda. Il wrapper, nel ruolo di writer, invia al DBMS i dati da salvare.



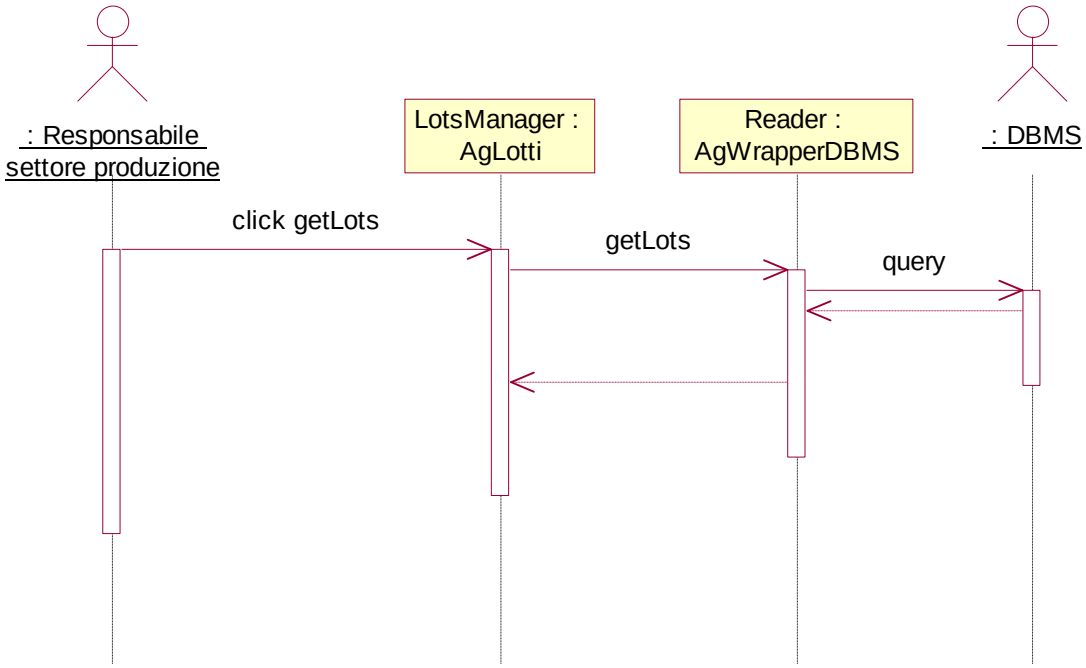
3.3.31 Avvio agente lotti

All’avvio del sietema l’agente lotti, nel ruolo di lots manager, chiede al wrapper la lista degli ordini non inseriti in lotti di produzione. Il wrapper, nel ruolo di reader, esegue una query al DBMS e restituisce la lista degli ordini all’agente lotti.



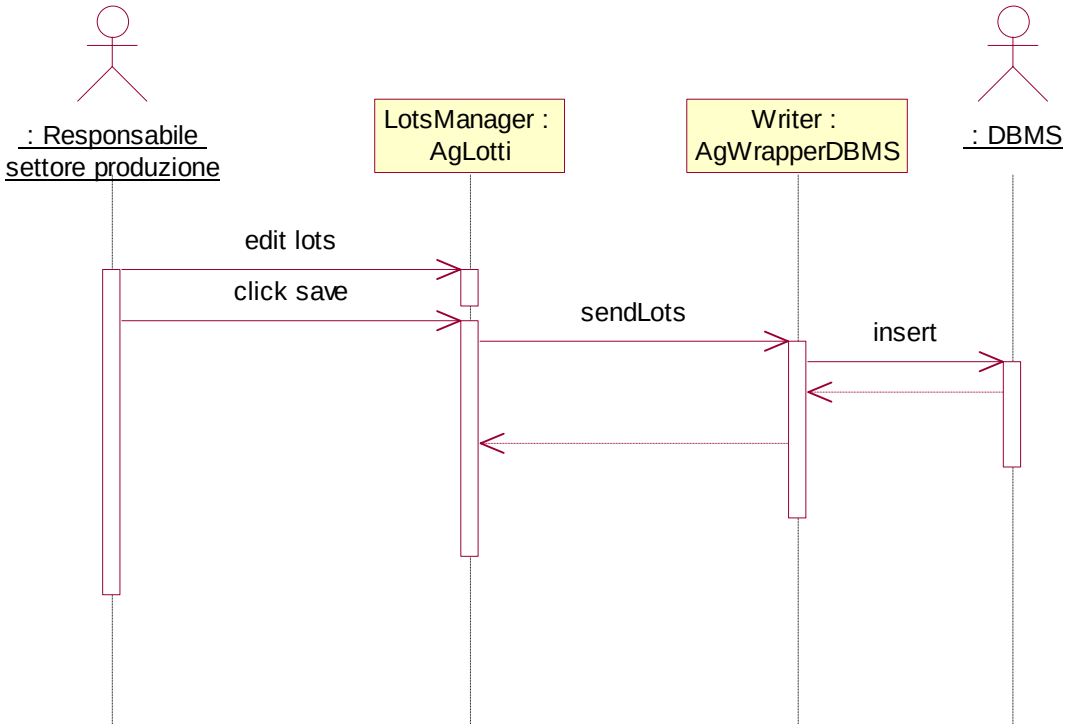
3.3.32 Recupero lottizzazione

Il responsabile settore produzione desidera visualizzare la lottizzazione non ancora schedulata. Seleziona l'opzione 'recupero lotti' dall'interfaccia dell'agente lotti. L'agente lotti, nel ruolo di lots manager, richiede al wrapper la lista dei lotti richiesta dall'utente. Il wrapper, nel ruolo di reader, esegue una query al DBMS e invia la lista dei lotti all'agente lotti.



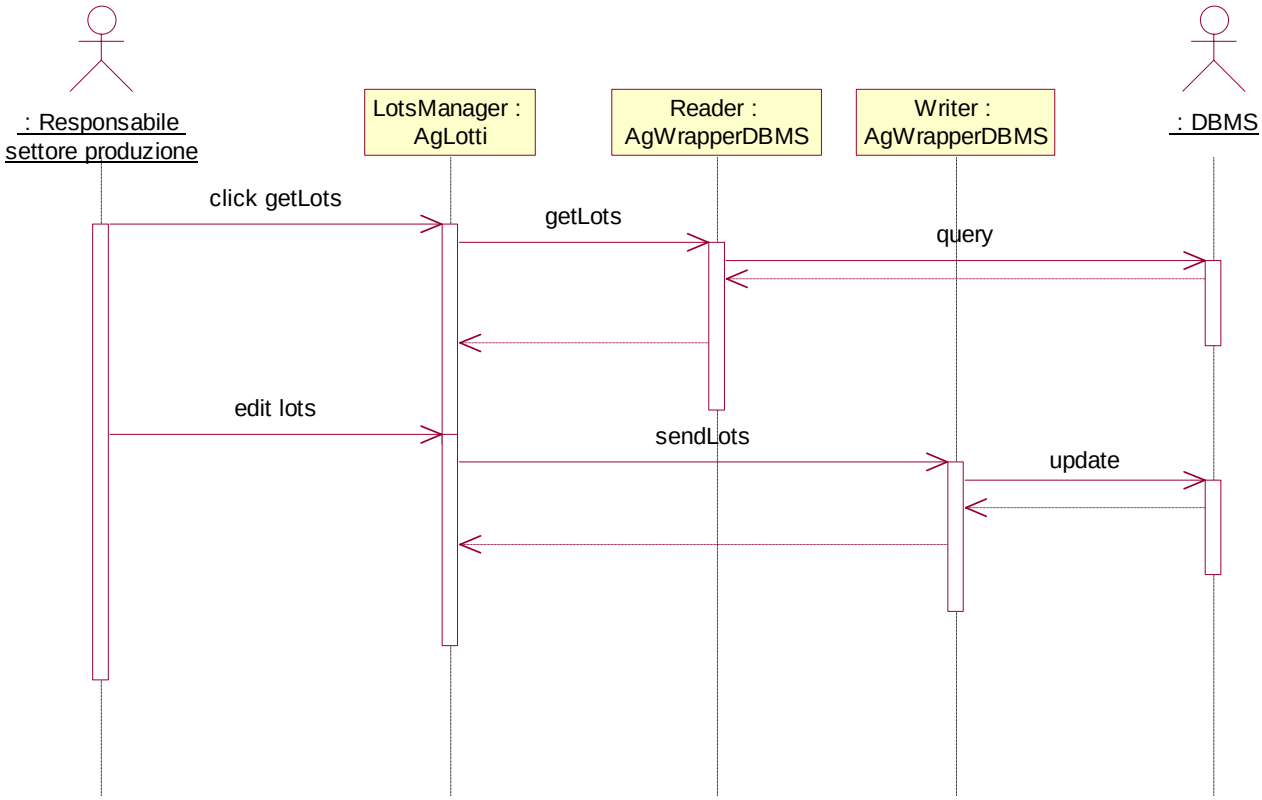
3.3.33 Inserisci lottizzazione

Il responsabile settore produzione, dopo aver creato i lotti di produzione desiderati, desidera salvarli nell'archivio dell'azienda. Seleziona l'opzione 'salva lotti' dall'interfaccia dell'agente lotti. Quest'utlimo, nel ruolo di lots manager, invia al wrapper i nuovi lotti creati. Il wrapper, nel ruolo di writer, invia al DBMS i dati da inserire.



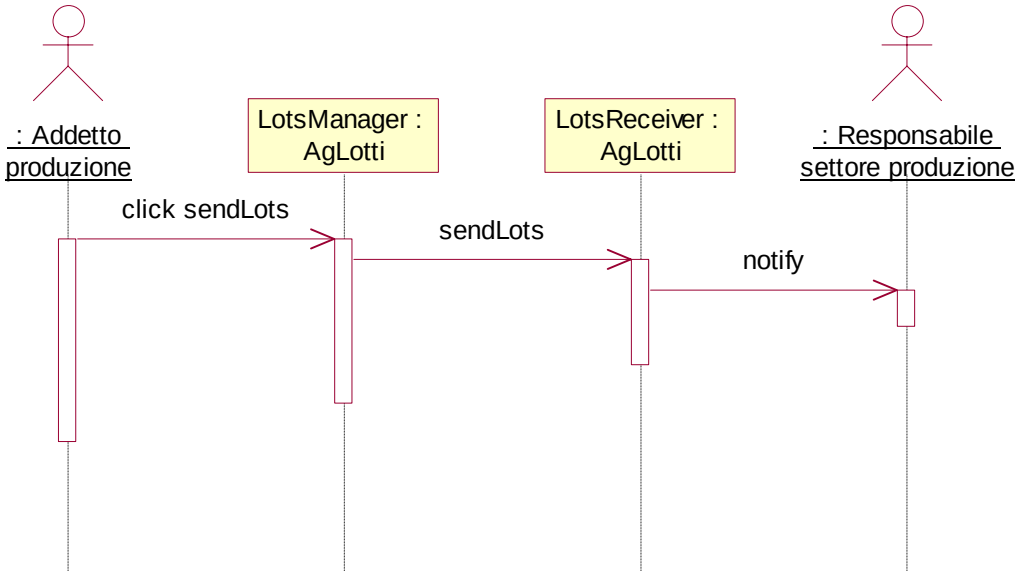
3.3.34 Modifica lottizzazione

Il responsabile settore produzione desidera midificare i lotti di produzione creati e archiviati. Seleziona l'opzione 'recupera lottizzazione' dall'interfaccia dell'agente lotti. L'agente lotti, nel ruolo di lots manager, richiede al wrapper la lista dei lotti desiderata dall'utente. Il wrapper, nel ruolo di reader, esegue una query al DBMS e restituisce la lista dei lotti all'agente lotti. Quest'ultimo mostra all'utente la lottizzazione desiderata e ne permette la modifica solo se il lotto non è già stato schedulato. L'utente effettua le modifiche desiderate e infine seleziona l'opzione 'salva'. L'agente lotti chiede conferma delle modifiche effettuate, il responsabile settore produzione conferma e l'agente lotti invia i dati da aggiornare al wrapper. Il wrapper invia al DBMS i dati relativi ai lotti da aggiornare.



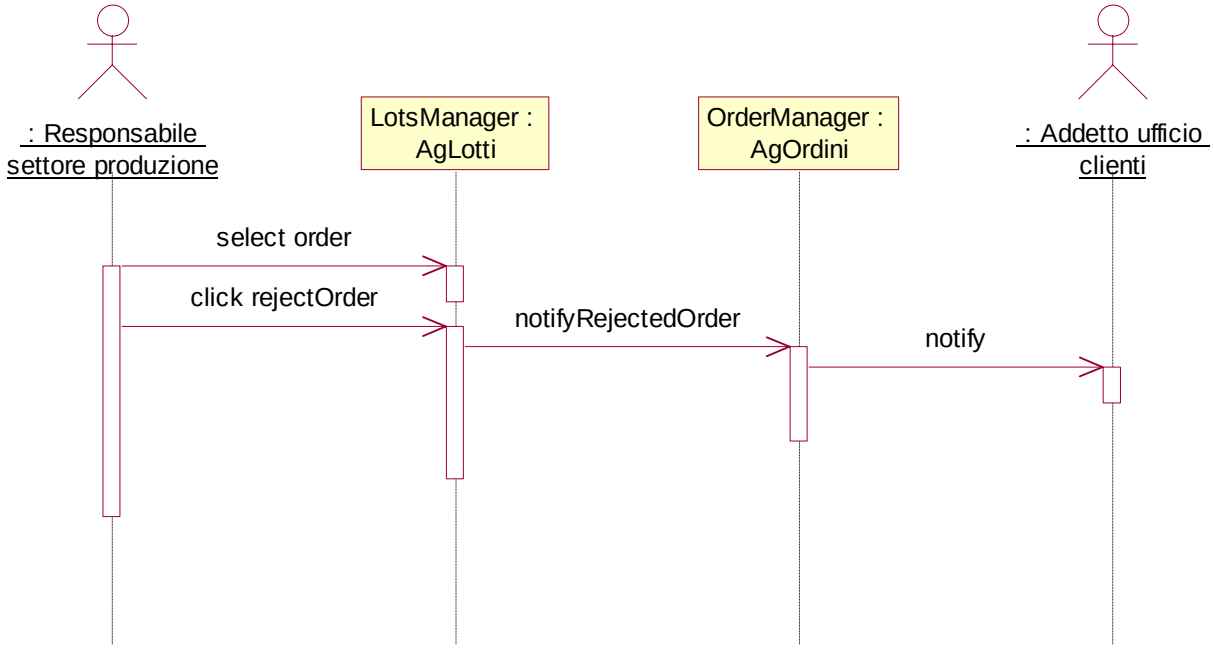
3.3.35 Invio Lottizzazione

L’addetto produzione, dopo aver creato lotti con numero di biciclette compresi tra 50 e 150 e data di scadenza vicina, invia al responasbile settore produzione la lottizzazione effettuata e gli ordini non inseriti in alcun lotto. L’addetto produzione seleziona l’opzione ‘invia lotti’ dall’interfaccia dell’agente lotti. L’agente lotti, nel ruolo di lots manager, invia i dati all’istanza dell’agente lotti utilizzata dal responsabile settore produzione che riceve i lotti nel ruolo di lots receiver. I lotti vengono quindi mostrati al responsabile settore produzione.



3.3.36 Rifiuto ordine

Il responsabile settore produzione desidera rifiutare un ordine. Seleziona l'ordine desiderato e seleziona l'opzione 'rifiuta ordine'. L'agente lotti invia all'agente ordini un messaggio contenente l'ordine da eliminare. L'agente ordini, nel ruolo di order manaqger, riceve il messaggio e notifica all'addetto ufficio clienti il rifiuto dell'ordine.



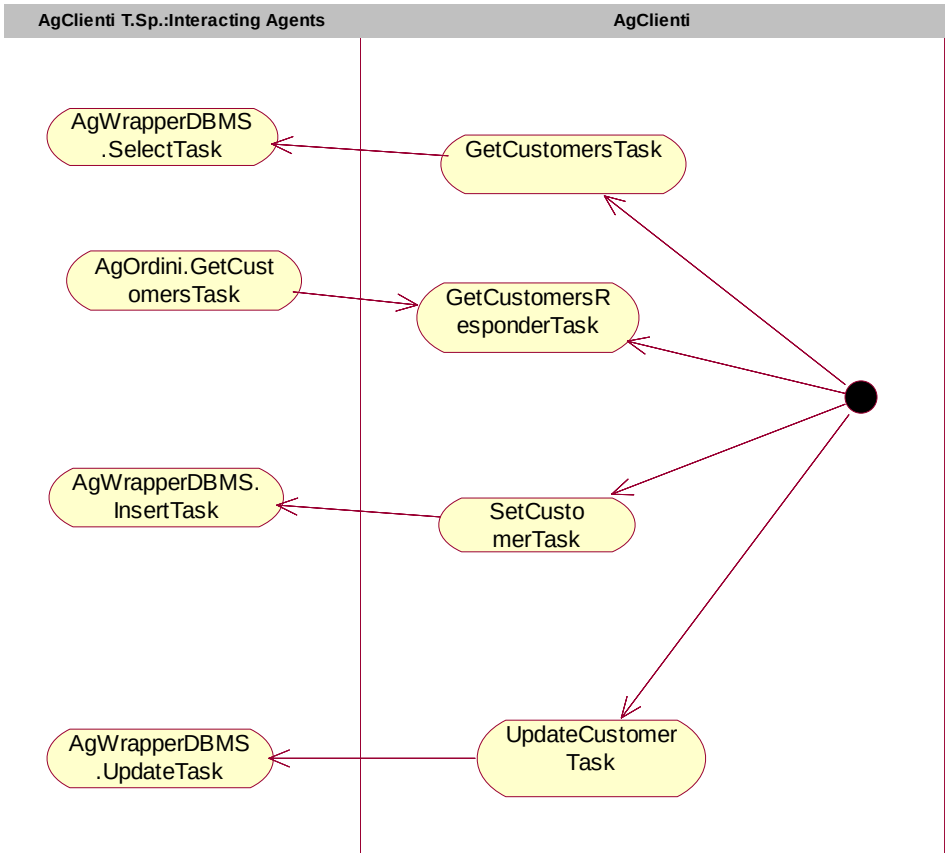
Name Role	*Agent which plays it*	*Description*	*Responsibilities*
LotsReceiver	AgLotti	Permette al responsabile settore produzione di ricevere la lottizzazione parziale effettuata dall’addetto produzione	Ricezione della lottizzazione parziale inviata dall’addetto produzione.
Logger	AgOrdini	Permette all'addetto ufficio clienti di autenticarsi	Autenticazione dell’addetto ufficio clienti.
Logger	AgLotti	Permette all'utente di autenticarsi	Autenticazione dipendenti settore produzione.
Logger	AgSchedulazione	Permette all'utente di autenticarsi.	Autenticazione del responsabile stabilimento.
Logger	AgGestioneDipendenti	Permette all'utente di autenticarsi.	Autenticazione dell’amministratore di sitema.
Logger	AgConsumi	Permette all'utente di autenticarsi.	Autenticazione del responsabile magazzino.
AuthenticationManager	AgAutenticazione	Riceve username e password dell'utente che desiera autenticarsi. Verifica la correttezza dei dati e restituisce all'agente il ruolo corrispondente.	Ricezione delle richieste di autenticazione, verifica dei dati di autenticazione e restituzione del ruolo corrispondente ai dati di autenticazione.
Reader	AgAutenticazione	Recupera i dati sui dipendenti.	Fornire i dati sui dipendenti.
EmployeesManager	AgGestioneDipendenti	Permette la gestione degli impiegati.	Inserimento e modifica dei dati sui dipendenti.

LabelManager	AgEtichette	Permette la visualizzazione e la stampa delle etichette	Visualzizzazione e stampa etichette. Salvataggio storia produzione.
LotsManager	AgLotti	Pemette all'utente la gestione dei lotti	Creazione e modifica lotti di produzione.
StoreManager	AgConsumi	Permette all'utente la gestione delle scorte di magazzino.	Creazione, modifica ed eliminazione componenti. Gestione approvvigionamenti.
SchedulingManager	AgSchedulazione	Permette all'utente di gestire la schedulazione dei lotti in uno stabilimento.	Schedulazione lotti. Fornire all’agente consumi i dati sulla produzione per l’approvvigionamento.
CustomerManager	AgClienti	Permette all'utente la gestione dei clienti.	Creazione e modifica clienti. Fornire all’agente ordini dati sui clienti.
OrderManager	AgOrdini	Permette all'autente la gestione degli ordini	Creazione, modifica ed elimianzione ordini.
PackingListManager	AgConsumi	Permette la gestione della distinta di produzione	Creazione e modifica distinta di produzione.
Reader	AgWrapperDBMS	Recupera i dati richiesti dal DBMS	Esecuzione di query di selezione al DBMS
Writer	AgWrapperDBMS	Scrive i dati sul DBMS.	Esecuzione di inserimenti, modifiche ed eliminazioni al DBMS.
SupplierManager	AgFornitori	Permette la gestione dei dati dei fornitori di componenti	Creazione e modifica formitori. Fornire dati sui fornitori all’gente consumi.

3.4 Tasks Specification phase

3.4.1 Agent: AgClienti

Si occupa della creazione, modifica ed eliminazione dei dati relativi ai clienti.



3.4.1.1 Task: GetCustomersResponderTask

Description — Fornisce la lista dei clienti. Utilizza il protocollo FIPA Query

Action — Riceve e risponde alla richiesta inviata dal behaviour *AgOrdini.GetCustomersTask*

Data — query_conv

Behavior — Ricevuto il query-ref, invia al behaviour *AgOrdini.GetCustomersTask* un agree ed un inform con la lista dei clienti richiesta.

3.4.1.2 Task: SetCustomerTask

Description — Invia al wrapper i dati relativi ad un cliente per l'inserimento nel database. Utilizza il protocollo FIPA Request

Action Invia al behaviour *AgWrapperDBMS.InsertTask* i dati del cliente da inserire.

Data agent_to_request

Behavior Invia un request al behaviour *AgWrapperDBMS.InsertTask* con i dati del cliente da inserire. Attende per l'arrivo di un agree e di un inform da parte del behaviur partecipante. All'arrivo dell'inform verrà mostrato all'utente un messaggio relativo all'inserimento dei dati.

3.4.1.3 Task: UpdateCustomerTask

Description Invia al wrapper i dati relativi ad un cliente per l'aggiornamento nel database. Utilizza il protocollo FIPA Request

Action Invia al behaviour *AgWrapperDBMS.UpdateTask* i dati del cliente da aggiornare.

Data agent_to_request

Behavior Invia un request al behaviour *AgWrapperDBMS.Updatek* con i dati del cliente da aggiornare. Attende per l'arrivo di un agree e di un inform da parte del behaviur partecipante. All'arrivo dell'inform verrà mostrato all'utente un messaggio relativo all'aggiornamento dei dati.

3.4.1.4 Task: GetCustomersTask

Description Richiede al wrapper la lista dei clienti. Utilizza il protocollo FIPA Query

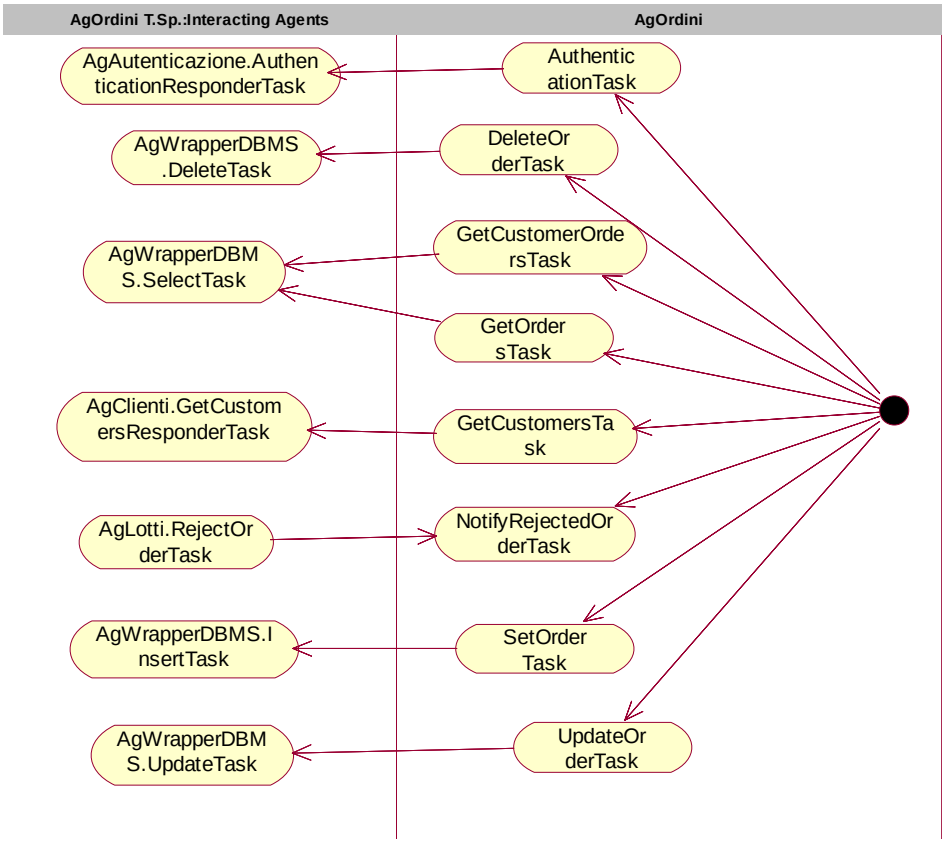
Action Richiede al behaviour *AgWrapperDBMS.SelectTask* la lista dei clienti.

Data

Behavior Invia un query-ref al behaviuor *AgWrapperDBMS.SelectTask* richiedendo la lista dei clienti. Attende per l'arrivo di un agree e di un inform con la lista dei clienti da parte del behaviur partecipante.

3.4.2 Agent: AgOrdini

Si occupa della creazione, modifica ed eliminazione degli ordini.



3.4.2.1 Task: AuthenticationTask

Description — Permette all'addetto uffico clienti di autenticarsi. Utilizza il protocollo FIPA Query.

Action — Invia al behaviour *AgAutenticazione.AuthenticationResponderTask* i dati dell'utente che si autentica.

Data — agent_to_query

Behavior — Invia un query-if al behaviour *AgAutenticazione.AuthenticationResponderTask.* Attende per un agree ed un inform di risposta contenente il ruolo del dipendente che si è autenticato.

3.4.2.2 Task: DeleteOrderTask

Description — Permette l'eliminazione di un ordine che non sia già stato inserito in un lotto di produzione. Utilizza il protocollo FIPA Request.

Action Invia al behaviour *AgWrapperDBMS.DeleteTask* l'ordine da eliminare.

Data agent_to_request

Behavior Invia un request al behaviour *AgWrapperDBMS.DeleteTask*. Attende per un agree ed un inform di risposta. Mostra all'utente un messaggio relativo all'eliminazione dell'ordine dal database.

3.4.2.3 Task: GetCustomerOrdersTask

Description Permette la visualizzazione degli ordini effettuati da un cliente. Utilizza il protocollo FIPA Query.

Action Invia al behaviour *AgWrapperDBMS.SelectTask* il cliente di cui desidera la lista degli ordini.

Data agent_to_query

Behavior Invia un query-ref al behaviour *AgWrapperDBMS.SelectTask*. Attende un agree ed un inform di risposta contenente la lista desiderata.

3.4.2.4 Task: GetCustomersTask

Description Richiede all'agente clienti la lista dei clienti. Utilizza il protocollo FIPA Query.

Action Invia al behaviour *AgClienti.GetCustomersResponderTask* la richiesta di invio della lista dei clienti.

Data agent_to_query

Behavior Invia un query-ref al behaviour *AgClienti.GetCustomersResponderTask*. Attende per un agree ed un inform di risposta contenente la lista desiderata.

3.4.2.5 Task: GetOrdersTask

Description Richiede al wrapper la lista degli ordini effettuati all'interno di un certo intervallo di tempo. Utilizza il protocollo FIPA Query.

Action Invia al behavioiur *AgWrapperDBMS.SelectTask* la richiesta di invio di ordini.

Data agent_to_query

Behavior Invia un query-ref al behaviour *AgWrapperDBMS.SelectTask*. Attende per un agree ed un inform di risposta contenente la lista desiderata.

3.4.2.6 Task: NotifyRejectedOrderTask

Description Comunica all'addetto ufficio clienti il rifiuto di un ordine da parte del settore produzione. Utilizza il protocollo FIPA Inform

Action Riceve dal behaviour *AgLotti.RejectOrderTask* il rifiuto di un ordine.

Data query_conv

Behavior Attende l'arrivo di un inform da parte del behaviuor *AgLotti.RejectOrderTask* contenente l'ordine rifiutato. Ricevuto l'inform verrà visualizzato all'utente un messaggio relativo all'ordine rifiutato.

3.4.2.7 Task: SetOrderTask

Description Permette all'utente l'inserimento di un ordine nel database. Utilizza il protocollo FIPA Request.

Action Invia al behaviour *AgWrapper.InsertTask* l'ordine da inserire nel database.

Data agent_to_request

Behavior Invia un request al behaviour *AgWrapper.InsertTask*. Attende per un agree e un inform di risposta. All'arrivo dell'inform viene visualizzato un messaggio relativo all'inserimento dell'ordine.

3.4.2.8 Task: UpdateOrderTask

Description Permette all'utente di aggiornare un ordine che non sia già stato inserito in un lotti di produzione. Utilizza il protocollo FIPA Request.

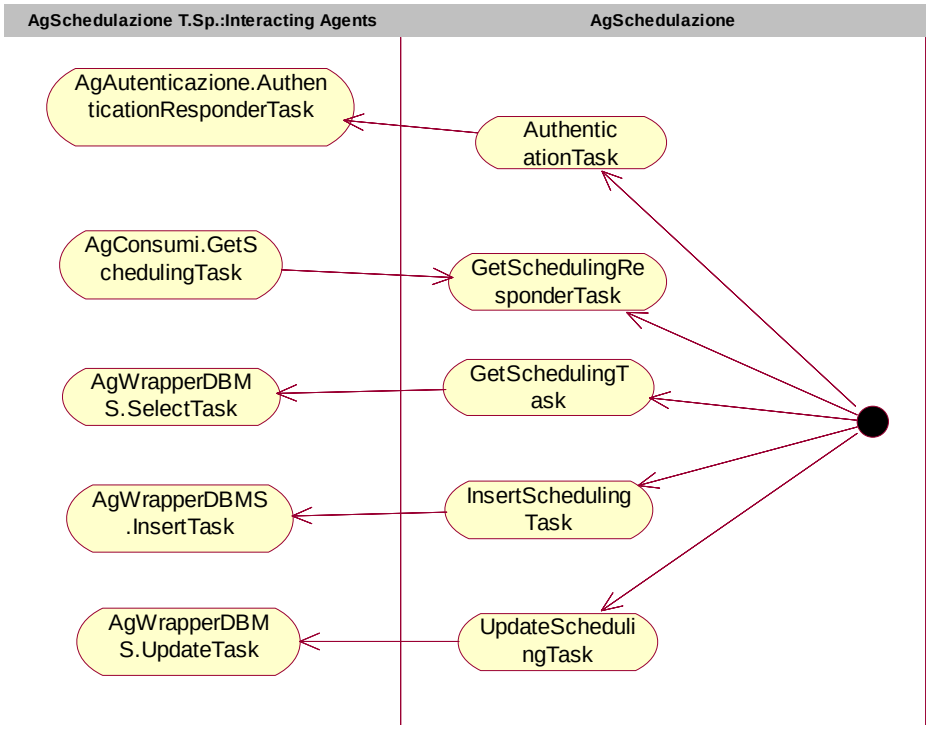
Action Invia al behaviour *AgWrapperDBMS.UpdateTask* l'ordine che si desidera modificare.

Data agent_to_request

Behavior Invia un request al behaviour *AgWrapperDBMS.UpdateTask*. Attende per un agree e un inform di risposta. All'arrivo dell'inform viene visualizzato un messaggio relativo all'aggiornamento dell'ordine.

3.4.3 Agent: AgSchedulazione

Si occupa della gestione della schedulazione dei lotti di produzione.



3.4.3.1 Task: AuthenticationTask

Description — Permette al responsabile stabilimento di autenticarsi. Utilizza il protocollo FIPA Query.

Action — Invia al behaviour *AgAutenticazione.AuthenticationResponderTask* i dati dell'utente che si autentica.

Data — agent_to_query

Behavior — Invia un query-if al behaviour *AgAutenticazione.AuthenticationResponderTask*. Attende per un agree ed un inform di risposta contenente il ruolo del dipendente che si è autenticato.

3.4.3.2 Task: GetSchedulingResponderTask

Description — Fornisce la schedulazione per gli approvvigionamenti. Utilizza il protocollo FIPA Query

Action	Riceve e risponde alla richiesta inviata dal behaviour *AgConsumi.GetSchedulingTask*
Data	query_conv
Behavior	Ricevuto il query-ref, invia al behaviour *AgConsumi.GetSchedulingTask* un agree ed un inform con la schedulazione richiesta.

3.4.3.3 Task: GetSchedulingTask

Description	Richiede al wrapper la schedulazione. Utilizza il protocollo FIPA Query.
Action	Invia al behaviour *AgWrapperDBMS.SelectTask* la richiesta per l'invio della schedulazione.
Data	agent_to_query
Behavior	Invia un query-ref al behaviour *AgWrapperDBMS.SelectTask*. Attende per un agree ed un inform di risposta contenente la schedulazione richiesta.

3.4.3.4 Task: InsertSchedulingTask

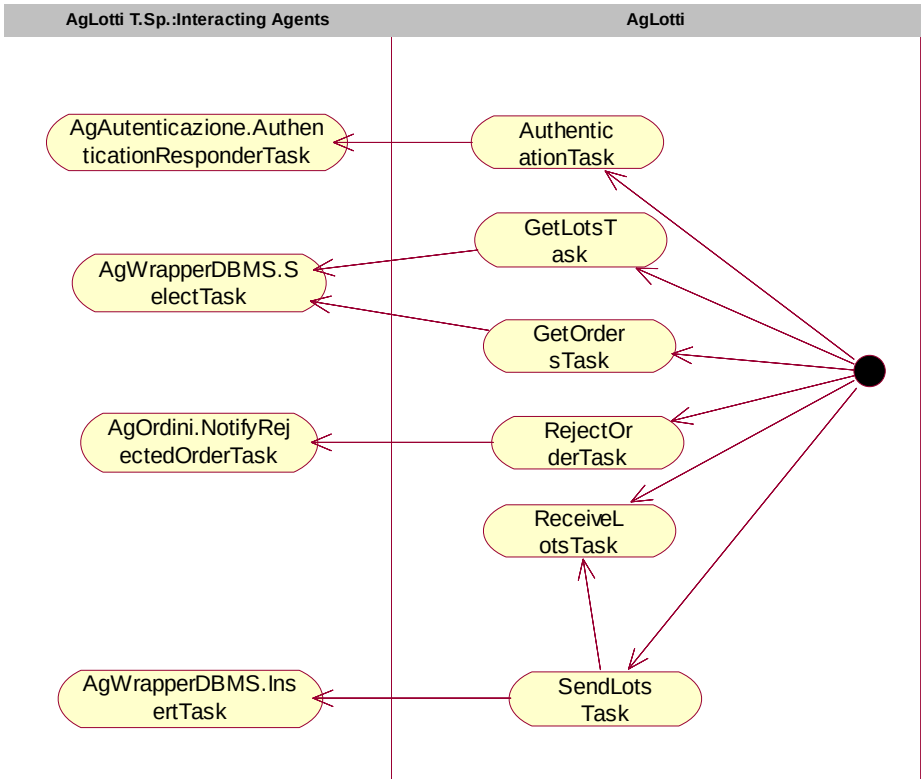
Description	Invia al wrapper la schedulazione da inserire nel database. Utilizza il protocollo FIPA Request.
Action	Invia al behaviour *AgWrapperDBMS.InsertTask* la schedulazione da inserire.
Data	agent_to_request
Behavior	Invia un request al behaviour *AgWrapperDBMS.InsertTask*. Attende per un agree ed un inform di riposta. Ricevuto l'inform viene mostrato all'utente un messaggio relativo vall'inserimento dei dati

3.4.3.5 Task: UpdateSchedulingTask

Description	Invia al wrapper la schedulazione da aggiornare nel database. Utilizza il protocollo FIPA Request.
Action	Invia al behaviour *AgWrapperDBMS.UpdateTask* la schedulazione da aggiornare.
Data	agent_to_request
Behavior	Invia un request al behaviour *AgWrapperDBMS.UpdateTask*. Attende per un agree ed un inform di riposta. Ricevuto l'inform viene mostrato all'utente un messaggio relativo all'aggiornamento dei dati

3.4.4 Agent: AgLotti

Si occupa della creazione, modifica ed eliminazione dei lotti produzione.



3.4.4.1 Task: AuthenticationTask

Description — Permette all'addetto produzione e al responsabile settore produzione di autenticarsi. Utilizza il protocollo FIPA Query.

Action — Invia al behaviour *AgAutenticazione.AuthenticationResponderTask* i dati dell'utente che si autentica.

Data — agent_to_query

Behavior — Invia un query-if al behaviour *AgAutenticazione.AuthenticationResponderTask.* Attende per un agree ed un inform di risposta contenente il ruolo del dipendente che si è autenticato.

3.4.4.2 Task: GetLotsTask

Description	Richiede al wrapper la lista dei lotti non schedulati. Utilizza il protocollo FIPA Query.
Action	Invia al behavioiur *AgWrapperDBMS.SelectTask* la richiesta di invio di lotti.
Data	agent_to_query
Behavior	Invia un query-ref al behaviour *AgWrapperDBMS.SelectTask*. Attende per un agree ed un inform di risposta contenente la lista desiderata.

3.4.4.3 Task: GetOrdersTask

Description	Richiede al wrapper la lista degli ordini non inseriti in lotti di produzione. Utilizza il protocollo FIPA Query.
Action	Invia al behavioiur *AgWrapperDBMS.SelectTask* la richiesta di invio di ordini.
Data	agent_to_query
Behavior	Invia un query-ref al behaviour *AgWrapperDBMS.SelectTask*. Attende per un agree ed un inform di risposta contenente la lista desiderata.

3.4.4.4 Task: RejectOrderTask

Description	Permette al responsabile settore produzione di notificare il rifiuto di un ordine all'addetto ufficio clienti. Utilizza il protocollo FIPA Inform.
Action	Invia l'ordine da eliminare al behaviour *AgOrdini.NotifyRejectedOrder*.
Data	inform_conv
Behavior	Invia un inform al behaviour *AgOrdini.NotifyRejectedOrder*.

3.4.4.5 Task: SendLotsTask

Description	Permette all'addetto produzione di inviare la lottizzazione parziale da lui effettuata al responsabile settore produzione. Inoltre permette al responsabile settore produzione di inviare la lottizzazione definitiva al wrapper per l'inserimento nel database. Utilizza il protocollo FIPA Request.
Action	Invia al behaviour *AgLotti.ReceiveLots* la lottizzazione parziale.

Invia al behaviuor *AgWrapperDBMS.InsertTask* la lista dei lotti da inserire nel database.

Data agent_to_request

Behavior Invia un request al behaviour *AgLotti.ReceiveLots*.
Invia un request al behaviuor *AgWrapperDBMS.InsertTask* contenente la lista dei lotti da salvare. Attende per un agree e un inform di risposta. Viene visualizzato all'utente un messaggio relativo all'inserimento dei dati.

3.4.4.6 Task: ReceiveLotsTask

Description Riceve la lottizzazione parziale effettuata dall'addetto produzione. Utilizza il protocollo FIPA Request.

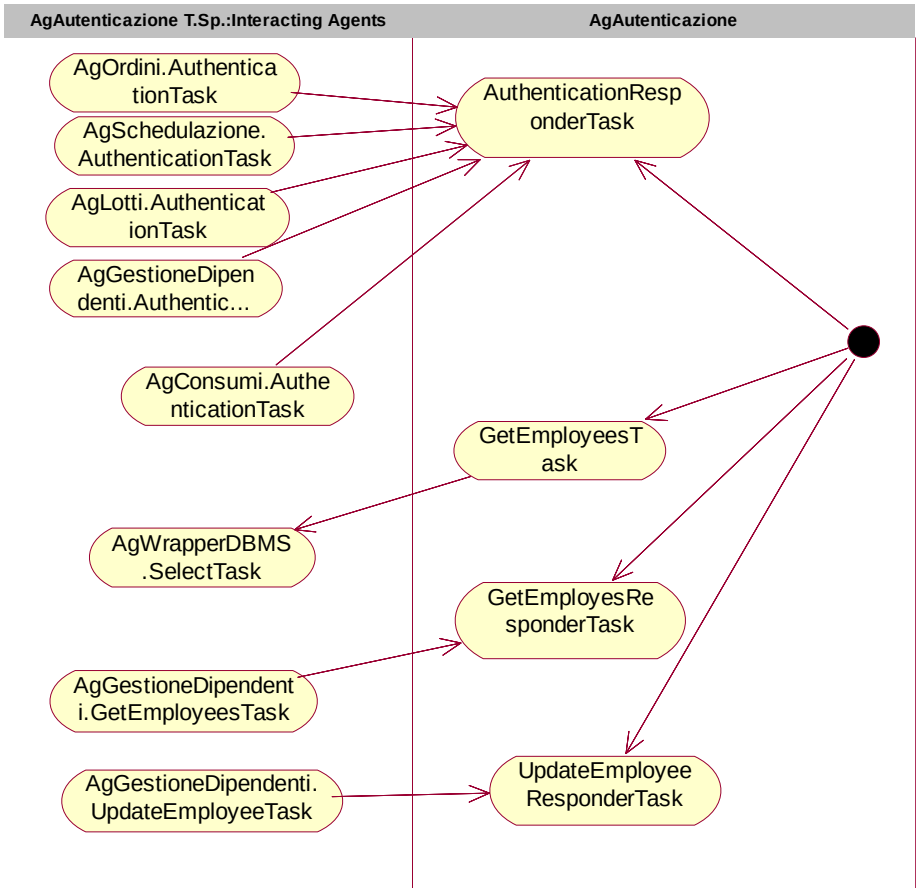
Action Riceve la lottizzazione parziale effettuata dall'addetto produzione dal task *AgLotti.SendLots*.

Data agent_to_query

Behavior Riceve un request dal behaviour *AgLotti.SendLots*.

3.4.5 Agent: AgAutenticazione

Si occupa della gestione delle autenticazioni dei dipendenti.



3.4.5.1 Task: AuthenticationResponderTask

Description — Riceve le richieste di autenticazione dagli altri agenti presenti nel sistema. Utilizza il protocollo FIPA Query.

Action — Riceve dai behaviuors *AgOrdini.AuthenticationTask*, *AgSchedulazione .AuthenticationTask*, *AgLotti.AuthenticationTask*, *AgGestioneDipendenti.AuthenticationTask* e *AgConsumi*.AuthenticationTask le richieste di verifica username e password.

Data — query_conv

Behavior — Attende da uno dei beviours comunicanti un query-if relativo all'autenticazione di un dipendente. Controlla tra la lista dei dipendenti se sono presenti username e password ricevuti e in caso affermativo restituisce il ruolo corrispondente.

3.4.5.2 Task: GetEmployeesTask

Description	Richiede al wrapper la lista dei dipendenti. Utilizza il protocollo FIPA Query
Action	Richiede al behaviour *AgWrapperDBMS.SelectTask* la lista dei dipendenti.
Data	
Behavior	Invia un query-ref al behaviuor *AgWrapperDBMS.SelectTask* richiedendo la lista dei dipendenti. Attende per l'arrivo di un agree e di un inform con la lista dei dipendenti da parte del behaviur partecipante.

3.4.5.3 Task: GetEmployesResponderTask

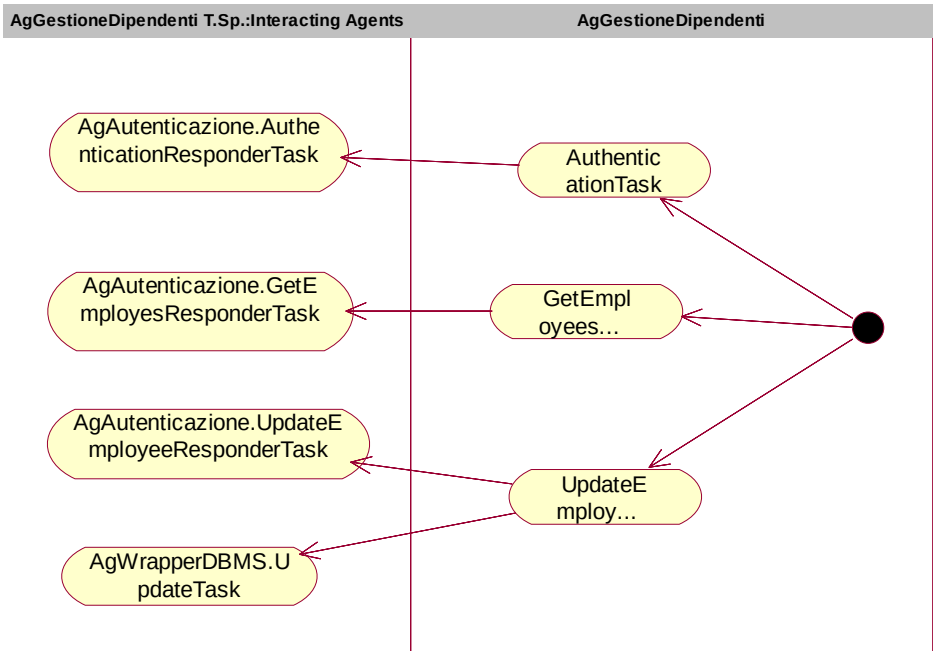
Description	Fornisce la lista dei dipendenti. Utilizza il protocollo FIPA Query
Action	Riceve e risponde alla richiesta inviata dal behaviour *AgConsumi.GetEmployeesTask*
Data	query_conv
Behavior	Ricevuto il query-ref, invia al behaviour *AgConsumi.GetEmployeesTask* un agree ed un inform con la lista dei dipendenti richiesta.

3.4.5.4 Task: UpdateEmployeeResponderTask

Description	Riceve l'aggiornamento dei dati relativi ad un dipendente dall'agente gestione dipendenti. Utilizza il protocollo FIPA Request.
Action	Riceve i dati da modificare dal behaviour *AgGestioneDipendenti.UpdateEmployeeTask*.
Data	query_conv
Behavior	Riceve un request dal behaviour *AgGestioneDipendenti.UpdateEmployeeTask*. Risponde con un agree, modifica la lista dei dipendenti dell'agente autenticazione e risponde con un inform.

3.4.6 Agent: AgGestioneDipendenti

Si occupa della gestione dei dati dei dipendenti



3.4.6.1 Task: AuthenticationTask

Description — Permette all'amministratore di sistema di autenticarsi. Utilizza il protocollo FIPA Query.

Action — Invia al behaviour *AgAutenticazione.AuthenticationResponderTask* i dati dell'utente che si autentica.

Data — agent_to_query

Behavior — Invia un query-if al behaviour *AgAutenticazione.AuthenticationResponderTask*. Attende per un agree ed un inform di risposta contenente il ruolo del dipendente che si è autenticato.

3.4.6.2 Task: GetEmployeesTask

Description — Richiede all'agente autenticazione la lista dei dipendenti. Utilizza il protocollo FIPA Query.

Action — Invia al behaviour *AgAutenticazione.GetEmployeesResponderTask* la richiesta di invio della lista dei dipendenti.

Data agent_to_query

Behavior Invia un query-ref al behaviour *AgAutenticazione.GetEmployeesResponderTask.* Attende un agree ed un inform di risposta contenente la lista dei dipendenti richiesta.

3.4.6.3 Task: UpdateEmployeeTask

Description Invia al wrapper l'aggiornamento dei dati di un dipwndente. Utilizza il protocollo FIPA Request.

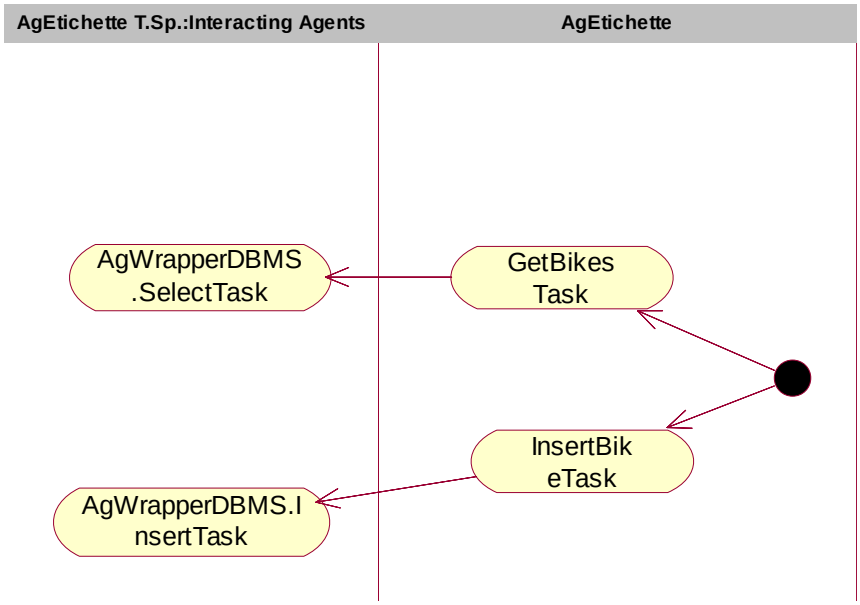
Action Invia al behaviour *AgWrapperDBMS.UpdateTask* I dati relativi al dipendente da aggiornare.

Data agent_to_request

Behavior Invia un requast al behaviour *AgWrapperDBMS.UpdateTask* contenente I dati del dipendente da eliminare. Attende un agree e un inform di risposta. Mostra all'utente un messaggio relativo all'aggiornamento dei dati.

3.4.7 Agent: AgEtichette

Si occupa della visualizzazione e stampa delle etichette



3.4.7.1 Task: GetBikesTask

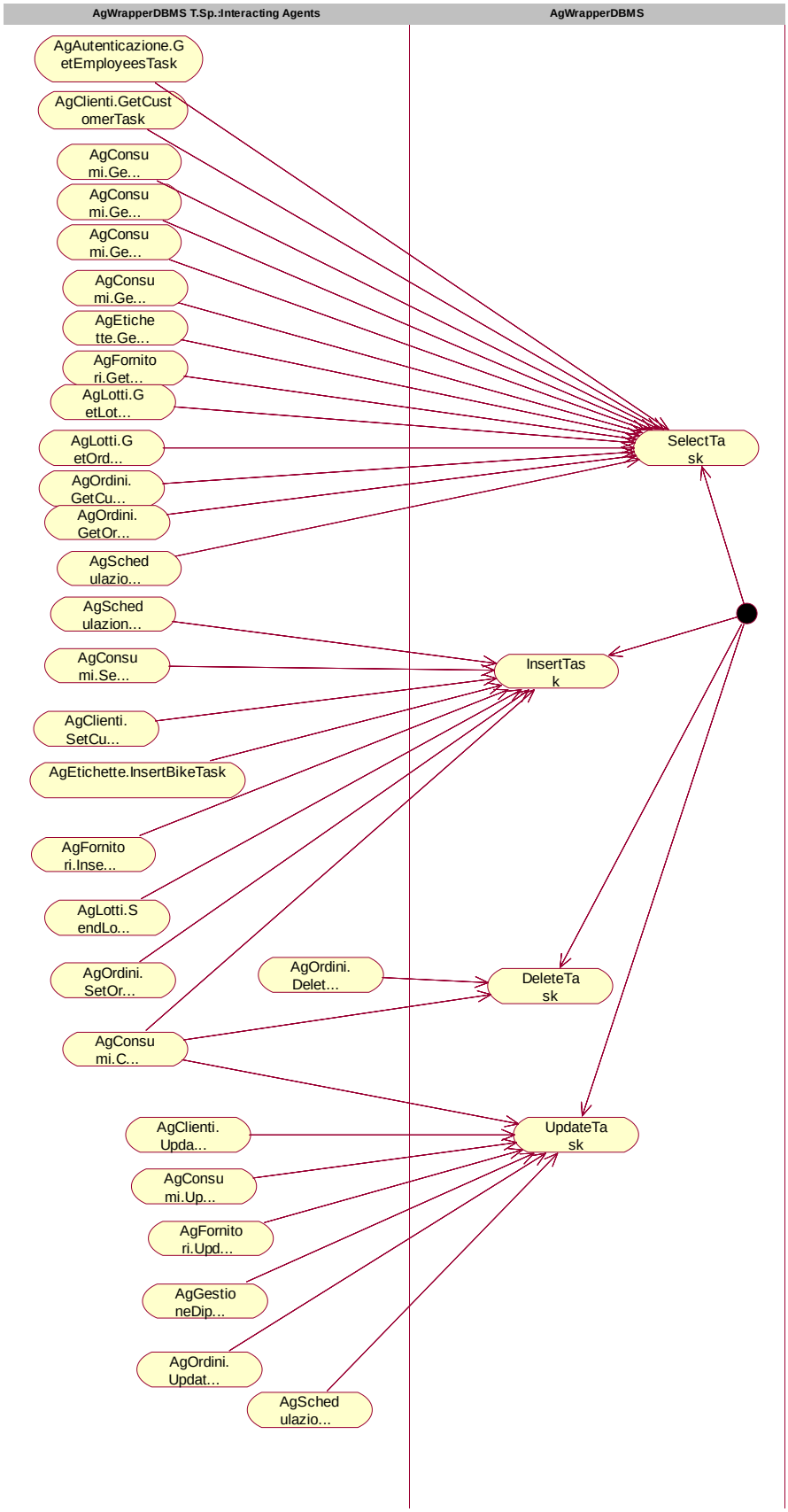
Description	Richiede al wrapper i dati sulla produzione del giorno corrente. Utilizza il protocollo FIPA Query.
Action	Invia al behaviuor *AgWrapper.SelectTask* la richiesta di invio della produzione.
Data	agent_to_query
Behavior	Invia al behaviour *AgWrapper.SelectTask* un request. Attende per un agree e un inform di risposta contenente le informazioni richieste.

3.4.7.2 Task: InsertBikeTask

Description	Invia al wrapper i dati relativi ad una bicicletta prodotta. Utilizza il protocollo FIPA Request.
Action	Invia al behaviour *AgWrapperDBMS.InsertTask* i dati sulla bicicletta da inserire nel database.
Data	agent_to_request
Behavior	Invia un request al behaviour *AgWrapperDBMS.InsertTask* contenente i dati sulla biciletta d inserire nel database. Attende per un agree ed un inform di risposta.

3.4.8 Agent: AgWrapperDBMS

Si occupa dell'interfacciamento degli agenti con il DBMS.



3.4.8.1 Task: SelectTask

Description Riceve le richieste di invio dati dal DBMS. Utilizza il protocollo FIPA Query.

Action Riceve dai behaviours partecipanti le richieste di invio dati.
Data query_conv

Behavior Riceve da uno dei behaviours partecipanti un query-ref contenente la richiesta di dati desiderata. Risponde con una gree. Effuttua la query di selezione corrispondente. Manda un inform al behaviour partecipante inviando i dati richiesti.

3.4.8.2 Task: InsertTask

Description Riceve le richieste di inserimento dati nel DBMS. Utilizza il protocollo FIPA Request.

Action Riceve dai behaviours partecipanti le richieste di inserimento dati.
Data request_conv

Behavior Riceve da uno dei behaviours partecipanti un request contenente i dati da inserire. Risponde con una gree. Effuttua l'insert nel DBMS. Manda un inform al behaviour partecipante.

3.4.8.3 Task: UpdateTask

Description Riceve le richieste di aggiornamento dati nel DBMS. Utilizza il protocollo FIPA Request.

Action Riceve dai behaviours partecipanti le richieste di aggiornamento dati.
Data request_conv

Behavior Riceve da uno dei behaviours partecipanti un request contenente i dati da aggiornare. Risponde con una gree. Effuttua l'update nel DBMS. Manda un inform al behaviour partecipante.

3.4.8.4 Task: DeleteTask

Description Riceve le richieste di elimianzione dati nel DBMS. Utilizza il protocollo FIPA

Request.

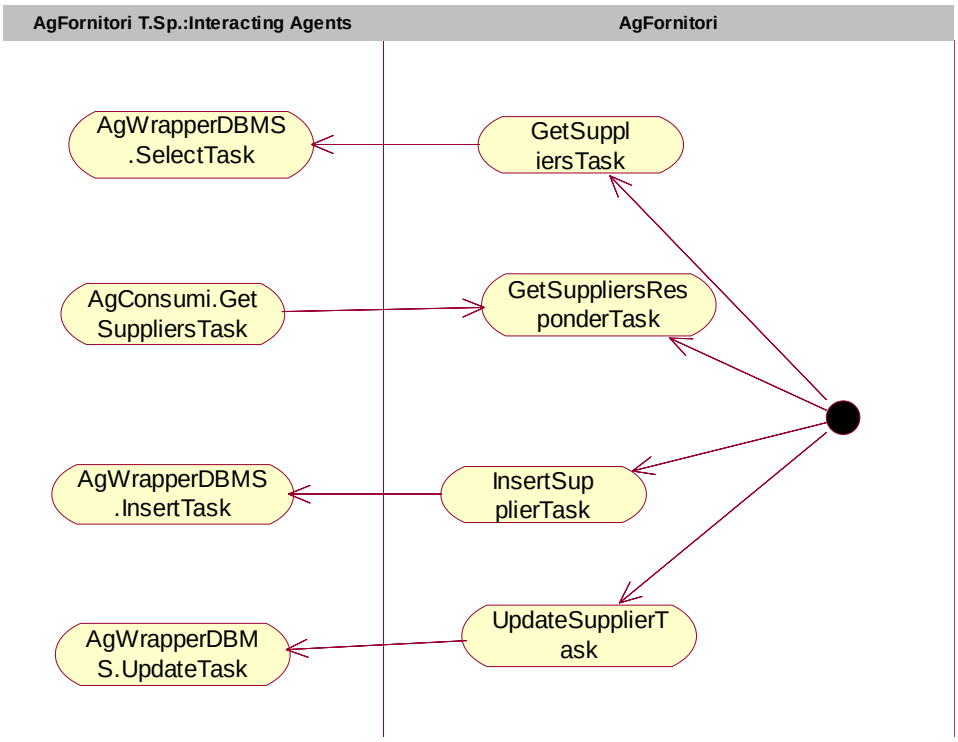
Action Riceve dai behaviours partecipanti le richieste di eliminazione dati.

Data request_conv

Behavior Riceve da uno dei behaviours partecipanti un request contenente i dati da eliminare. Risponde con una gree. Effuttua il delete nel DBMS. Manda un inform al behaviour partecipante.

3.4.9 Agent: AgFornitori

Si occupa della creazione, modifica ed eliminazione dei dati sui fornitori.



3.4.9.1 Task: GetSuppliersResponderTask

Description Fornisce la lista dei fornitori. Utilizza il protocollo FIPA Query

Action Riceve e risponde alla richiesta inviata dal behaviour *AgConsumi.GetSuppliersTask*

Data query_conv

Behavior Ricevuto il query-ref, invia al behaviour *AgConsumi.GetSuppliersTask* un agree ed un inform con la lista dei fornitori richiesta.

3.4.9.2 Task: GetSuppliersTask

Description Richiede al wrapper la lista dei fornitori. Utilizza il protocollo FIPA Query

Action Richiede al behaviour *AgWrapperDBMS.SelectTask* la lista dei fornitori.

Data

Behavior Invia un query-ref al behaviuor *AgWrapperDBMS.SelectTask* richiedendo la lista dei fornitori. Attende per l'arrivo di un agree e di un inform con la lista dei fornitori da parte del behaviur partecipante.

3.4.9.3 Task: InsertSupplierTask

Description Invia al wrapper i dati relativi ad un fornitore per l'inserimento nel database. Utilizza il protocollo FIPA Request

Action Invia al behaviour *AgWrapperDBMS.InsertTask* i dati del fornitore da inserire.

Data agent_to_request

Behavior Invia un request al behaviour *AgWrapperDBMS.InsertTask* con i dati del fornitore da inserire. Attende per l'arrivo di un agree e di un inform da parte del behaviur partecipante. All'arrivo dell'inform verrà mostrato all'utente un messaggio relativo all'inserimento dei dati.

3.4.9.4 Task: UpdateSupplierTask

Description Invia al wrapper i dati relativi ad un fornitore per l'aggiornamento nel database. Utilizza il protocollo FIPA Request

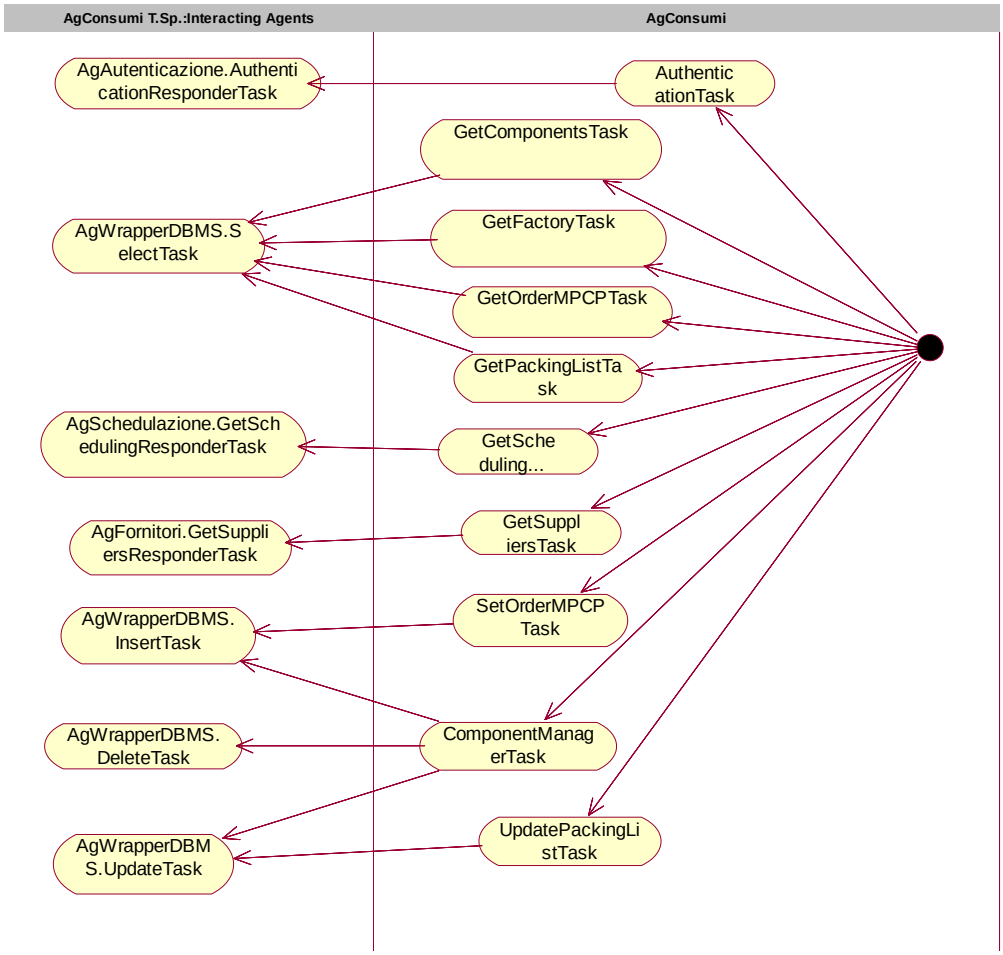
Action Invia al behaviour *AgWrapperDBMS.UpdateTask* i dati del fornitore da aggiornare.

Data agent_to_request

Behavior Invia un request al behaviour *AgWrapperDBMS.Updatek* con i dati del fornitore da aggiornare. Attende per l'arrivo di un agree e di un inform da parte del behaviur partecipante. All'arrivo dell'inform verrà mostrato all'utente un messaggio relativo all'aggiornamento dei dati.

3.4.10 Agent: AgConsumi

Si occupa della gestione degli approvvigionamenti. Permette la creazione, modifica ed elimazione dei dati su componenti, distinta di produzione ed ordine materiali.



3.4.10.1 Task: AuthenticationTask

Description — Permette al responsabile magazzino di autenticarsi. Utilizza il protocollo FIPA Query.

Action — Invia al behaviour *AgAutenticazione.AuthenticationResponderTask* i dati dell'utente che si autentica.

Data — agent_to_query

Behavior — Invia un query-if al behaviour *AgAutenticazione.AuthenticationResponderTask*. Attende per un agree ed un inform di risposta contenente il ruolo del dipendente che si è autenticato.

3.4.10.2 Task: GetFactoryTask

Description Richiede al wrapper I dati relativi ad uno stabilimento. Utilizza il protocollo FIPA Query.

Action Invia al behaviour *AgWrapperDBMS.SelectTask* la richiesta di invio dei dati relativi ad uno stabilimento.

Data agent_to_query

Behavior Invia un request al behaviour *AgWrapperDBMS.SelectTask*. Attende per un agree ed un inform di risposta contenente I dati richiesti.

3.4.10.3 Task: GetOrderMPCPTask

Description Richiede al wrapper l'invio degli ordini componenti eseguiti da un fornitore. Utilizza il protocollo FIPA Query.

Action Invia al behaviour *AgWrapperDBMS.SelectTask* la richiesta di invio dgli ordini componenti riferiti ad un certo fornitore.

Data agent_to_query

Behavior Invia al behaviour *AgWrapperDBMS.SelectTask* un request contenente il fornitore di cui si desiderano gli ordini. Attende per un agree ed un inform contenente i dati richiesti.

3.4.10.4 Task: GetPackingListTask

Description Richiede al wrapper l'invio della distinta di produzione di un modello di biciclette. Utilizza il protocollo FIPA Query.

Action Invia al behaviour *AgWrapperDBMS.SelectTask* la richiesta di invio della distinta di produzione di un modello di bicicletta.

Data agent_to_query

Behavior Invia al behaviour *AgWrapperDBMS.SelectTask* un request contenente il modello di cui si vuole la distinta di produzione. Attende per un agree ed un inform contenente la distinta di produzione richiesta.

3.4.10.5 Task: GetSchedulingTask

Description Richiede all'agente schedulazione la produzione effettuata in un certo intervallo di tempo per l'approvvigionamento materie prime e componenti pronte. Utilizza il protocollo FIPA Query.

Action Invia al behaviour *AgSchedulazione.GetSchedulingResponderTask* la richiesta di invio dati.

Data agent_to_query

Behavior Invia un query-ref al behaviour *AgSchedulazione.GetSchedulingResponderTask* richiedendo la schedulazione. Attende per un agree e un inform contenente i dati richiesti.

3.4.10.6 Task: GetSuppliersTask

Description Richiede all'agente fornitori la lista dei fornitori per i componenti utilizzati in un certo stabiliemento. Utilizza il protocollo FIPA Query.

Action Invia al behaviour *AgFornitori.GetSuppliersResponderTask* la richiesta di invio della lista dei fornitori.

Data agent_to_query

Behavior Invia un request al behaviour *AgFornitori.GetSuppliersResponderTask* contenente la richiesta di invio della lista dei fornitori. Attende per un agree e un inform contenente la lista dei fornitori richiesta.

3.4.10.7 Task: SetOrderMPCP

Description Invia al wrapper un ordine componenti relativo ad un fornitore per l'inserimento nel database. Utilizza il rpotocollo FIPA Request.

Action Invia al behaviour *AgWrapperDBMS.InsertTask* l'ordine da inserire.

Data agent_to_request

Behavior Invia un request al behaviour *AgWrapperDBMS.InsertTask* con l'ordine da inserire. Attende per un agree e un inform di conferma dell'inserimento. Mostra all'utente un messaggio relativo asll'inserimento dei dati.

3.4.10.8 Task: UpdatePackingListTask

Description Invia al wrapper l'aggiornamento della distinta di produzione relativa ad un certo modello di bicicletta prodotta in un determinato stabilimento. Utilizza il protocollo Request.

Action Invia al behaviour *AgWrapperDBMS.UpdateTask* I dati relativi all'aggiornamento della distinta di produzione.

Data agent_to_request

Behavior Invia un request al behaviuor *AgWrapperDBMS.UpdateTask* contenente i dati da aggiornare. Attende per un agree ed un inform di conferma aggiornamento dati. Mostra un messaggio all'utente relativo all'aggiornamento dei dati.

3.4.10.9 Task: GetComponentsTask

Description Richiede al wrapper l'invio della lista dei componenti. Utilizza il protocollo FIPA Query.

Action Invia al behaviour *AgWrapperDBMS.SelectTask* la richiesta di invio della lista dei componenti.

Data

Behavior Invia un request al behaviour *AgWrapperDBMS.SelectTask*. Attende per un agree ed un inform contenente i dati richiesti.

3.4.10.10 Task: ComponentManagerTask

Description Invia al wrapper i dati relativi all'inserimento, la modifica e la cancellazione di componenti. Utilizza il protocollo FIPA Request.

Action Invia ad uno dei behaviours partecipanti la richiesta di inserimento, modifica o eliminazione dati.

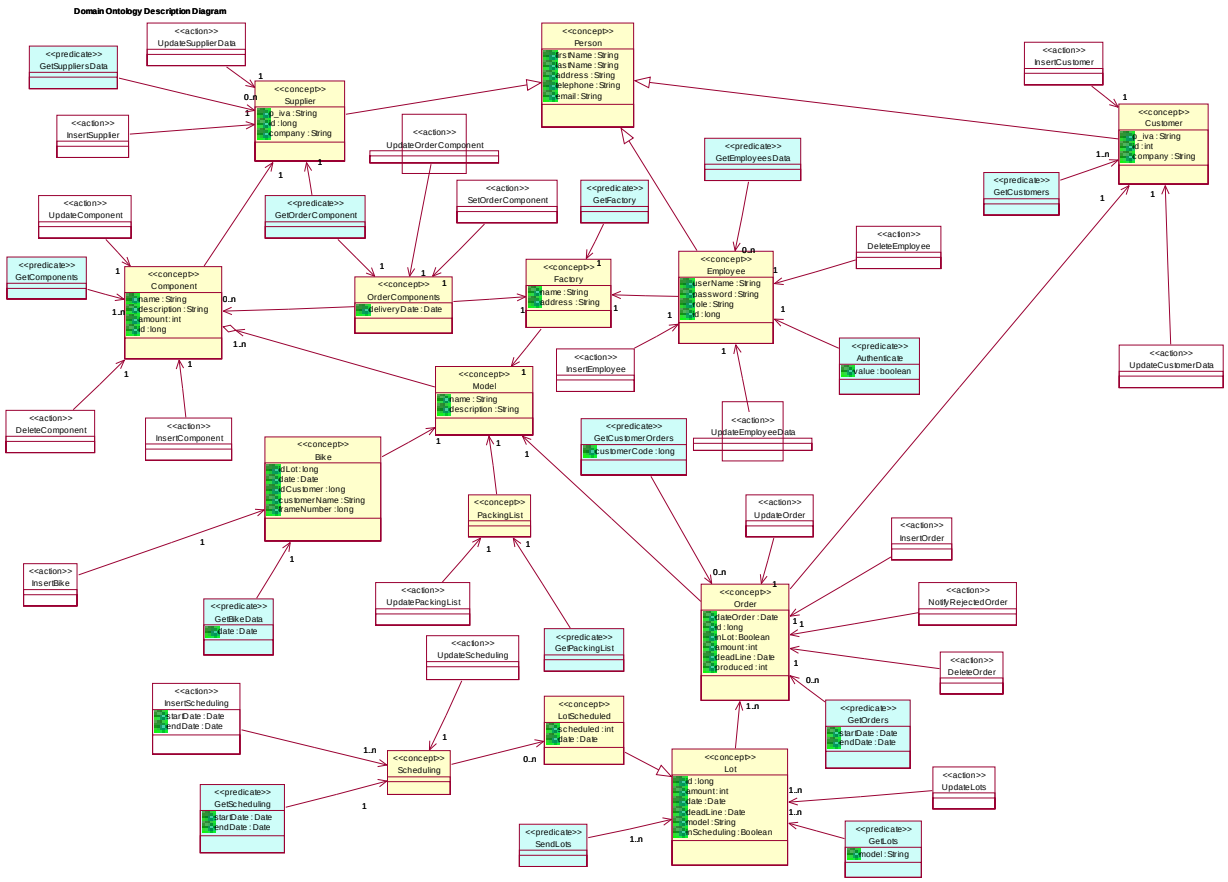
Data

Behavior Invia un request ad uno dei behaviuor partecipanti con i dati da inserire, modificare o eliminare. Attende per un agree e un inform. Mostra all'utente un messaggio sull'inserimento, l'aggiornamento o la modifica dei dati.

3.5 Ontology Description phase

3.5.1 Domain

Questo diagramma rappresenta l'ontologia del dominio. Le entità dell'ontologia (concetti, predicati, azioni) sono rappresentati come classi.



3.5.1.1 Elements of the domain ontology diagram (DOD)

Data name	*Stereotype*	*Field Name*	*Field Type*	*Description*
Person	concept	firstName	String	Concetto che rappresenta una persona. I suoi campi rappresentano nome, cognome, indirizzo, telefono ed e-mail della persona.
		lastName	String	
		address	String	
		telephone	String	
		email	String	
UpdateScheduling	action			Azione utilizzata per aggiornare i dati relativi alla schedulazione.
InsertScheduling	action	startDate	Date	Azione utilizzata per l'inserimento di una nuova schedulazione.
		endDate	Date	
LotScheduled	concept	scheduled	int	Concetto che rappresenta una frazione di un lotto sachedulata in un giorno. Estende il concetto lotto. I sui campi rappresentano la data di schedulazione e la quantità schedulata.
		date	Date	
Scheduling	concept			Concetto che rappresenta la schedulazione. Contiene al suo interno una lista di LotScheduled.
GetScheduling	predicate	startDate	Date	Predicato utilizzato per ottenere dati sulla schedulazione.
		endDate	Date	
SendLots	predicate			Predicato utilizzato per l'invio di dati relativi ai lotti.
GetLots	predicate	model	String	Predicato utilizzato per ottenere dati relativi ai lotti. Ha un

				campo model che specifica il modello di biciletta a cui il lotto si riferisce.
UpdateLots	action			Azione utilizzata per aggiornare dati sui lotti.
GetCustomers	predicate			Predicato utilizzato per ottenere dati dui clienti.
UpdateCustomerData	action			Azione utilizzata per aggiornre i dati relativi ad un cliente.
InsertCustomer	action			Azione utilizzata per inserire nel databse un nuovo cliente.
Authenticate	predicate	value	boolean	Predicato utilizzato per l'autenticazione dei dipendenti.
InsertEmployee	action			Azione utilizzata per l'inserimento di un nuovo dipendente.
UpdateEmployeeData	action			Azione utilizzata per l'aggiornamento dei dati di un dipendente.
GetEmployeesData	predicate			Predicato utilizzato per ottenere i dati dei dipendenti.
DeleteEmployee	action			Azione utilizzate per eliminare i dati di un dipendente.
GetOrders	predicate	startDate endDate	Date Date	Predicato utilizzato per ottenere gli ordini effettuati in un certo intervallo di tempo. I sui campi, startDate ed endDate, rappresentano rispettivamente le date di inizio e fine dell'intervallo di tempo.
DeleteOrder	action			Azione utilizzata per eliminare un certo ordine.
UpdateOrder	action			Azione utilizzata per aggiornare un certo ordine.

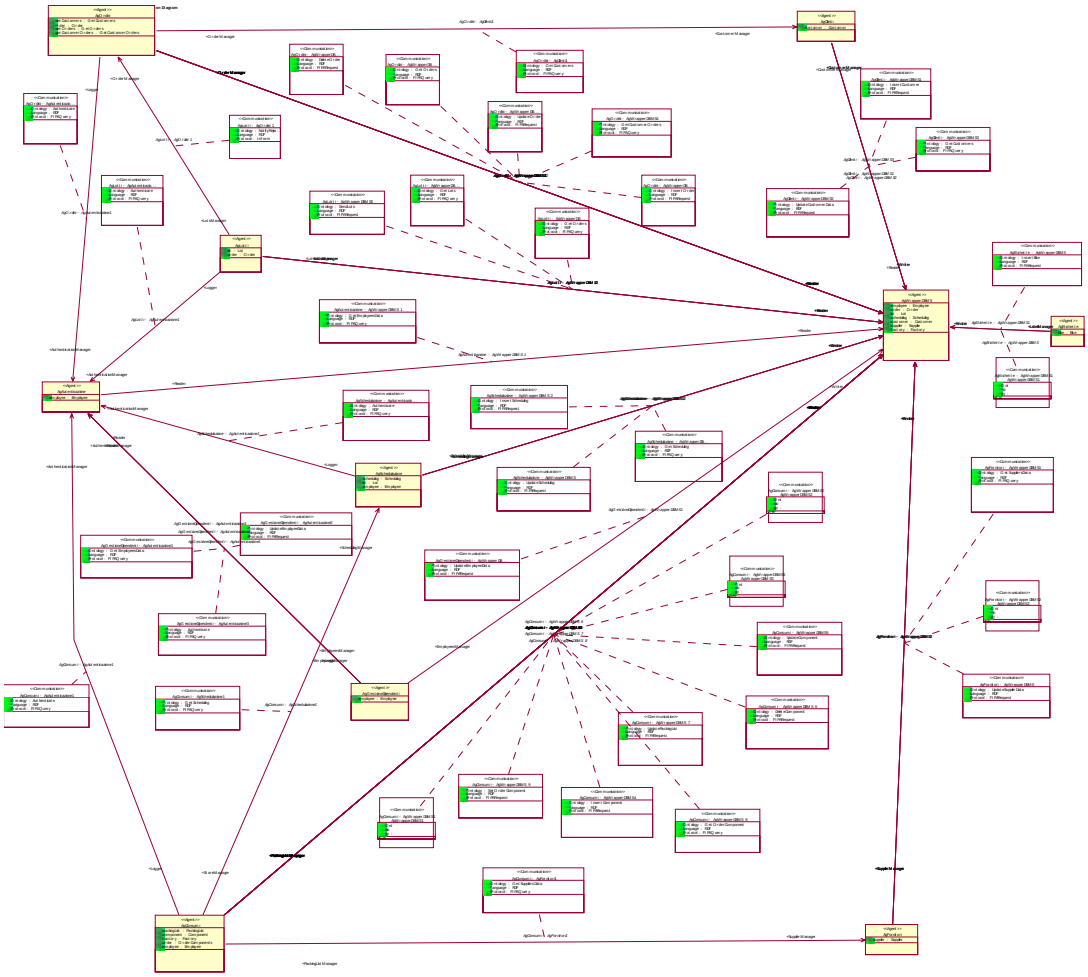
InsertOrder	action			Azione utilizzate per inserire un nuovo ordine.
Lot	concept	id amount date deadLine model inScheduling	long int Date Date String Boolean	Concetto che rappresenta un lotto di produzione. I suoi campi rappresentano la data di creazione, la data di scadenza e il modello. Inoltre il lotto può essere schedulato oppure no.
NotifyRejectedOrder	action			Azione utilizzata per notificare il rifiuto di un ordine.
Customer	concept	p_iva id company	String int String	Concetto che rappresenta un cliente. Estende il concetto persona aggiungendo i campi id, partita Iva e compagnia.
GetCustomerOrders	predicate	customerCode	long	Preedicato utilizzato per ottenere gli ordini effettuati da un cliente. Il campo customerCode è l'id del cliente.
InsertBike	action			Azione per l'inserimento di una bicicletta prodotta.
GetBikeData	predicate	date	Date	Predicato utilizzato per ottenere la produzione in un certo giorno.
GetPackingList	predicate			Predicato utilizzato per ottenere la distinta di produzione di un certo modello.
UpdatePackingList	action			Azione utilizzata per aggiornare i dati di una distinta di produzione.
Employee	concept	userName password role id	String String String long	Concetto che rappresenta un impiegato. Estende il concetto persona e aggiunge i campi username e password, utilizzati nella fase di autenticazione, l'id

				e il ruolo del dipendente.
GetFactory	predicate			Predicato utilizzato per ottenere i dati di un certo stabilimento.
UpdateSupplierData	action			Azione utilizzata per aggiornare i dati di un fornitore.
InsertSupplier	action			Azione utilizzata per inserire I dati di un fornitore.
GetSuppliersData	predicate			Predicato utilizzato, per ottenre i dati dei fornitori.
Order	concept	dateOrder id inLot amount deadLine produced	Date long Boolean int Date int	Concetto che rappresenta un ordine. I suoi campi rappresentano l'id dell'ordine, la data di creazione, la data di consegna, la quantità ed il modello di biciclette. Inoltre un ordine può essere in un lotto di produzione oppure no.
Bike	concept	idLot date idCustomer customerName frameNumber	long Date long String long	Concetto che rappresenta una bicicletta. I suoi campi rappresentano il numero di telaio, la data di produzione, il nome del cliente e l'id del lotto in cui è stata prodotta.
PackingList	concept			Concetto che rappresenta la distinta di produzione di un modello di bicicletta.
SetOrderComponent	action			Azione per inserire un ordine componenti a un fornitore.
UpdateOrderComponent	action			Azione utilizzata per aggiornare I dati relativi ad un ordine componenti.
GetOrderComponent	predicate			Predicato utilizzato per ottenere la lista degli ordini componenti relativi ad un certo fornitore.

Factory	concept	name address	String String	Concetto che rappresenta uno stabilimento. I suoi campi reppresentano il nome e l'indirizzo dello stabilimento.
UpdateComponent	action			Azione utilizzata per aggiornare I dati relativi ad un componente.
DeleteComponent	action			Azione utilizzata per eliminare un componente.
InsertComponent	action			Azione utilizzata per inserire un nuovo componente.
Supplier	concept	p_iva id company	String long String	Concetto che rappresenta un fornitore di componenti. Estende il concetto persona aggiungendo i campi id, partita Iva e compagnia.
Model	concept	name description	String String	Concetto che rappresenta un modello di bicicletta. I suoi campi sono il nome del modello e la descrizione delle caratteristiche del modello. Il modello contiene al suo interno una lista di componenti con le relative quantità.
OrderComponents	concept	deliveryDate	Date	Concetto che rappresenta un ordine di componenti. Ha un campo deliveryDate che rappresenta la dat di consegna.
Component	concept	name description amount id	String String int long	Concetto che rappresenta un componente. I suoi campi sono l'id del componente, il nome, la descrizione e la quantità.
GetComponents	predicate			Predicato utilizzato per ottenere i dati sui componenti.

3.5.2 Communication

This diagram describes the involved entities (agents and actors), their knowledge and their communication relationship. Each agent is identified by a class, its knowledge is identified by an attribute. An association is introduced for each message of a certain agent.



3.5.2.1 Agent knowledge

Agent	*Field Name*	*Data Type*
AgLotti	lot	Lot
	order	Order
AgGestioneDipendenti	employee	Employee
AgEtichette	bike	Bike
AgClienti	customer	Customer
AgOrdini	getCustomers	GetCustomers
	order	Order
	getOrders	GetOrders
	getCustomerOrders	GetCustomerOrders
AgFornitori	supplier	Supplier
AgAutenticazione	employee	Employee
AgWrapperDBMS	employee	Employee
	order	Order
	lot	Lot
	scheduling	Scheduling
	customer	Customer
	supplier	Supplier
	factory	Factory
AgConsumi	packingList	PackingList
	component	Component
	factory	Factory
	order	OrderComponents
	employee	Employee

AgSchedulazione	scheduling	Scheduling
	lot	Lot
	employee	Employee

3.5.2.2 Message content

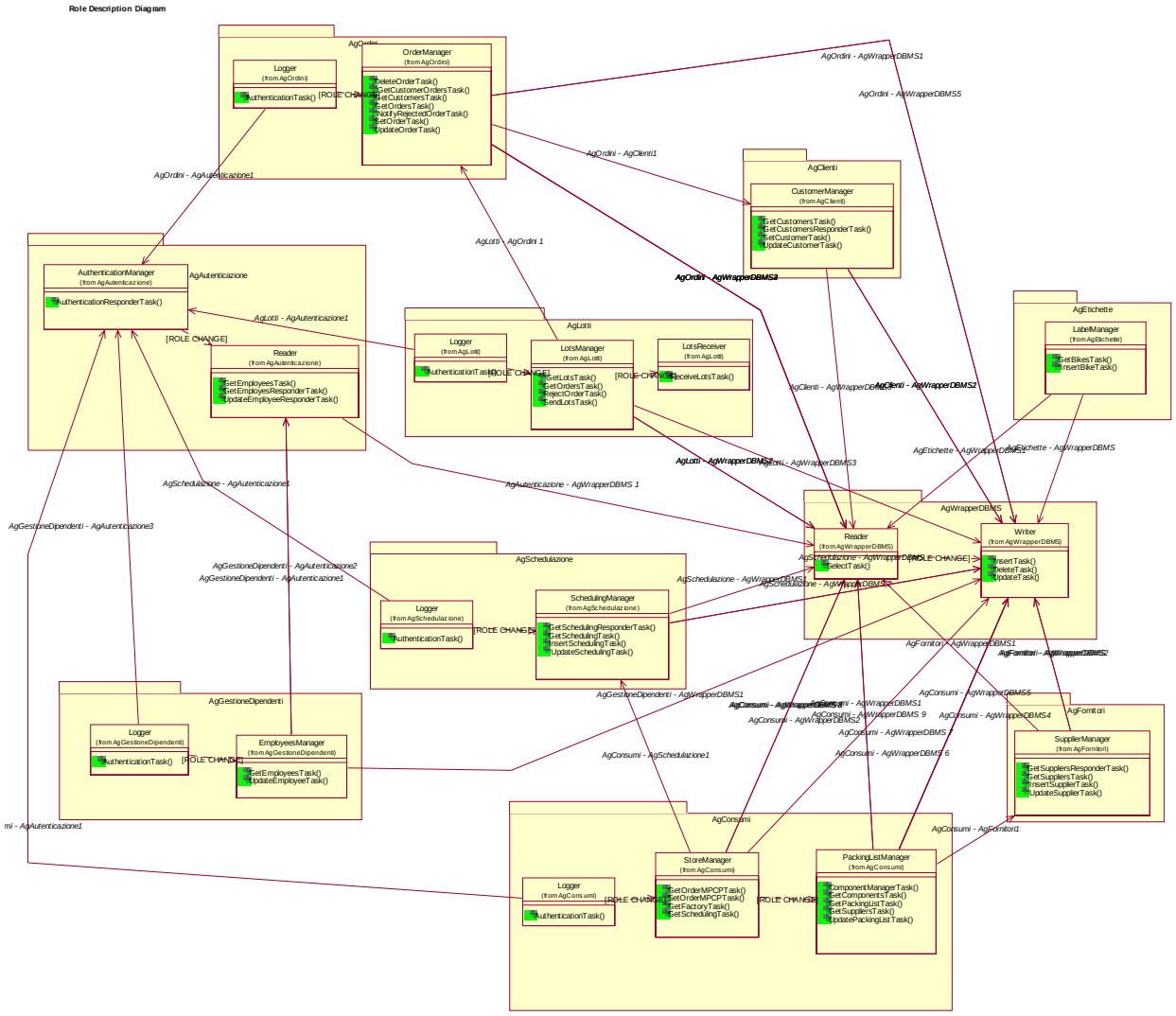
Name	MsgID	From Agent	To Agent	Language	Protocol	Data
AgLotti - AgOrdini 1	38	AgLotti	AgOrdini	RDF	Inform	NotifyRejectedOrder
AgOrdini - AgClienti1	3	AgOrdini	AgClienti	RDF	FIPA Query	GetCustomers
AgLotti - AgAutenticazione1	13	AgLotti	AgAutenticazione	RDF	FIPA Query	Authenticate
AgGestioneDipendenti - AgAutenticazione1	17	AgGestione Dipendenti	AgAutenticazione	RDF	FIPA Query	GetEmployees Data
AgGestioneDipendenti - AgAutenticazione2	4	AgGestione Dipendenti	AgAutenticazione	RDF	FIPA Request	UpdateEmployeeData
AgGestioneDipendenti - AgAutenticazione3	29	AgGestione Dipendenti	AgAutenticazione	RDF	FIPA Query	Authenticate
AgOrdini - AgAutenticazione1	5	AgOrdini	AgAutenticazione	RDF	FIPA Query	Authenticate
AgLotti - AgWrapperDBMS1	14	AgLotti	AgWrapperDBMS	RDF	FIPA Query	GetLots
AgLotti - AgWrapperDBMS2	15	AgLotti	AgWrapperDBMS	RDF	FIPA Query	GetOrders
AgLotti - AgWrapperDBMS3	16	AgLotti	AgWrapperDBMS	RDF	FIPA Request	SendLots
AgGestioneDipendenti - AgWrapperDBMS1	18	AgGestione Dipendenti	AgWrapperDBMS	RDF	FIPA Request	UpdateEmployeeData
AgEtichette - AgWrapperDBMS1	19	AgEtichette	AgWrapperDBMS	RDF	FIPA Query	GetBikeData
AgEtichette - AgWrapperDBMS	33	AgEtichette	AgWrapperDBMS	RDF	FIPA Request	InsertBike
AgAutenticazione -	39	AgAutentica	AgWrappe	RDF	FIPA	GetEmployees

AgWrapperDBMS 1		zione	rDBMS		Query	Data
AgClienti - AgWrapperDBMS1	1	AgClienti	AgWrapperDBMS	RDF	FIPA Request	InsertCustomer
AgClienti - AgWrapperDBMS2	2	AgClienti	AgWrapperDBMS	RDF	FIPA Request	UpdateCustomerData
AgClienti - AgWrapperDBMS3	40	AgClienti	AgWrapperDBMS	RDF	FIPA Query	GetCustomers
AgOrdini - AgWrapperDBMS1	6	AgOrdini	AgWrapperDBMS	RDF	FIPA Request	InsertOrder
AgOrdini - AgWrapperDBMS2	7	AgOrdini	AgWrapperDBMS	RDF	FIPA Request	UpdateOrder
AgOrdini - AgWrapperDBMS3	8	AgOrdini	AgWrapperDBMS	RDF	FIPAQuery	GetOrders
AgOrdini - AgWrapperDBMS4	9	AgOrdini	AgWrapperDBMS	RDF	FIPA Query	GetCustomerOrders
AgOrdini - AgWrapperDBMS5	10	AgOrdini	AgWrapperDBMS	RDF	FIPA Request	DeleteOrder
AgFornitori - AgWrapperDBMS1	20	AgFornitori	AgWrapperDBMS	RDF	FIPA Query	GetSuppliersData
AgFornitori - AgWrapperDBMS2	21	AgFornitori	AgWrapperDBMS	RDF	FIPA Request	InsertSupplier
AgFornitori - AgWrapperDBMS	32	AgFornitori	AgWrapperDBMS	RDF	FIPA Request	UpdateSupplierData
AgConsumi - AgAutenticazione1	22	AgConsumi	AgAutenticazione	RDF	FIPA Query	Authenticate
AgConsumi - AgFornitori1	28	AgConsumi	AgFornitori	RDF	FIPA Query	GetSuppliersData
AgConsumi - AgWrapperDBMS1	23	AgConsumi	AgWrapperDBMS	RDF	FIPA Query	GetPackingList
AgConsumi - AgWrapperDBMS2	24	AgConsumi	AgWrapperDBMS	RDF	FIPA Query	GetComponents
AgConsumi - AgWrapperDBMS3	25	AgConsumi	AgWrapperDBMS	RDF	FIPA Query	GetFactory
AgConsumi -	26	AgConsumi	AgWrappe	RDF	FIPA	InsertCompon

AgWrapperDBMS4			rDBMS		Request	ent
AgConsumi - AgWrapperDBMS5	27	AgConsumi	AgWrappe rDBMS	RDF	FIPA Request	UpdateCompo nent
AgConsumi - AgWrapperDBMS 6	30	AgConsumi	AgWrappe rDBMS	RDF	FIPA Request	DeleteCompo nent
AgConsumi - AgWrapperDBMS 7	31	AgConsumi	AgWrappe rDBMS	RDF	FIPA Request	UpdatePackin gList
AgConsumi - AgWrapperDBMS 8	36	AgConsumi	AgWrappe rDBMS	RDF	FIPA Query	GetOrderCom ponent
AgConsumi - AgWrapperDBMS 9	37	AgConsumi	AgWrappe rDBMS	RDF	FIPA Request	SetOrderCom ponent
AgSchedulazione - AgAutenticazione1	11	AgSchedulaz ione	AgAutentic azione	RDF	FIPA Query	Authenticate
AgSchedulazione - AgWrapperDBMS1	12	AgSchedulaz ione	AgWrappe rDBMS	RDF	FIPA Query	GetScheduling
AgSchedulazione - AgWrapperDBMS 2	34	AgSchedulaz ione	AgWrappe rDBMS	RDF	FIPA Request	InsertScheduli ng
AgSchedulazione - AgWrapperDBMS	35	AgSchedulaz ione	AgWrappe rDBMS	RDF	FIPA Request	UpdateSchedu ling
AgConsumi - AgSchedulazione1	41	AgConsumi	AgSchedul azione	RDF	FIPA Query	GetScheduling

3.6 Roles Description phase

This diagram describes the lifecycle of each agent, looking at roles which it can play, at collaboration that it needs, and the communication in which it participates. Each communication must be matched with a standard FIPA protocols.



3.6.1 Communication

From	To	Name	Refer to C.O.D. msg (MsgID)
AgOrdini:Logger	AgAutenticazione: AuthenticationManager	AgOrdini - AgAutenticazione1	5
AgLotti:Logger	AgAutenticazione: AuthenticationManager	AgLotti - AgAutenticazione1	13
AgSchedulazione: Logger	AgAutenticazione: AuthenticationManager	AgSchedulazione - AgAutenticazione1	11
AgGestioneDipendenti: Logger	AgAutenticazione: AuthenticationManager	AgGestioneDipendenti - AgAutenticazione3	29
AgConsumi:Logger	AgAutenticazione: AuthenticationManager	AgConsumi - AgAutenticazione1	22
AgGestioneDipendenti: EmployeesManager	AgAutenticazione: Reader	AgGestioneDipendenti - AgAutenticazione2	4
AgGestioneDipendenti: EmployeesManager	AgAutenticazione: Reader	AgGestioneDipendenti - AgAutenticazione1	17
AgConsumi: StoreManager	AgSchedulazione: SchedulingManager	AgConsumi - AgSchedulazione1	41
AgLotti:LotsManager	AgOrdini: OrderManager	AgLotti - AgOrdini 1	38
AgOrdini: OrderManager	AgClienti: CustomerManager	AgOrdini - AgClienti1	3
AgAutenticazione: Reader	AgWrapperDBMS: Reader	AgAutenticazione - AgWrapperDBMS 1	39
AgEtichette: LabelManager	AgWrapperDBMS: Reader	AgEtichette - AgWrapperDBMS1	19
AgLotti:LotsManager	AgWrapperDBMS: Reader	AgLotti - AgWrapperDBMS1	14
AgLotti:LotsManager	AgWrapperDBMS: Reader	AgLotti - AgWrapperDBMS2	15
AgConsumi: StoreManager	AgWrapperDBMS: Reader	AgConsumi - AgWrapperDBMS 8	36

AgConsumi: StoreManager	AgWrapperDBMS: Reader	AgConsumi AgWrapperDBMS3	- 25
AgSchedulazione: SchedulingManager	AgWrapperDBMS: Reader	AgSchedulazione AgWrapperDBMS1	- 12
AgClienti: CustomerManager	AgWrapperDBMS: Reader	AgClienti AgWrapperDBMS3	- 40
AgConsumi: PackingListManager	AgWrapperDBMS: Reader	AgConsumi AgWrapperDBMS1	- 23
AgConsumi: PackingListManager	AgWrapperDBMS: Reader	AgConsumi AgWrapperDBMS2	- 24
AgOrdini: OrderManager	AgWrapperDBMS: Reader	AgOrdini AgWrapperDBMS2	- 7
AgOrdini: OrderManager	AgWrapperDBMS: Reader	AgOrdini AgWrapperDBMS3	- 8
AgOrdini: OrderManager	AgWrapperDBMS: Reader	AgOrdini AgWrapperDBMS4	- 9
AgOrdini: OrderManager	AgWrapperDBMS: Writer	AgOrdini AgWrapperDBMS1	- 6
AgClienti: CustomerManager	AgWrapperDBMS: Writer	AgClienti AgWrapperDBMS1	- 1
AgSchedulazione: SchedulingManager	AgWrapperDBMS: Writer	AgSchedulazione AgWrapperDBMS 2	- 34
AgGestioneDipendenti: EmployeesManager	AgWrapperDBMS: Writer	AgGestioneDipendenti AgWrapperDBMS1	- 18
AgEtichette: LabelManager	AgWrapperDBMS: Writer	AgEtichette AgWrapperDBMS	- 33
AgLotti:LotsManager	AgWrapperDBMS: Writer	AgLotti AgWrapperDBMS3	- 16
AgConsumi: PackingListManager	AgWrapperDBMS: Writer	AgConsumi AgWrapperDBMS4	- 26
AgConsumi: StoreManager	AgWrapperDBMS: Writer	AgConsumi AgWrapperDBMS 9	- 37
AgSchedulazione: SchedulingManager	AgWrapperDBMS: Writer	AgSchedulazione AgWrapperDBMS	- 35

AgClienti: CustomerManager	AgWrapperDBMS: Writer	AgClienti AgWrapperDBMS2	- 2
AgConsumi: PackingListManager	AgWrapperDBMS: Writer	AgConsumi AgWrapperDBMS5	- 27
AgConsumi: PackingListManager	AgWrapperDBMS: Writer	AgConsumi AgWrapperDBMS 6	- 30
AgConsumi: PackingListManager	AgWrapperDBMS: Writer	AgConsumi AgWrapperDBMS 7	- 31
AgOrdini: OrderManager	AgWrapperDBMS: Writer	AgOrdini AgWrapperDBMS5	- 10
AgConsumi: PackingListManager	AgFornitori: SupplierManager	AgConsumi AgFornitori1	- 28
AgFornitori: SupplierManager	AgWrapperDBMS: Reader	AgFornitori AgWrapperDBMS1	- 20
AgFornitori: SupplierManager	AgWrapperDBMS: Writer	AgFornitori AgWrapperDBMS	- 32
AgFornitori: SupplierManager	AgWrapperDBMS: Writer	AgFornitori AgWrapperDBMS2	- 21

3.6.2 Dependencies

Dependent/Client	Dependee/server	Kind of dependency	Description	Attached Note
Logger	OrderManager	[ROLE CHANGE]		
Logger	SchedulingManager	[ROLE CHANGE]		
Logger	LotsManager	[ROLE CHANGE]		
LotsManager	LotsReceiver	[ROLE CHANGE]		
AuthenticationManager	Reader	[ROLE CHANGE]		
Logger	EmployeesManager	[ROLE		

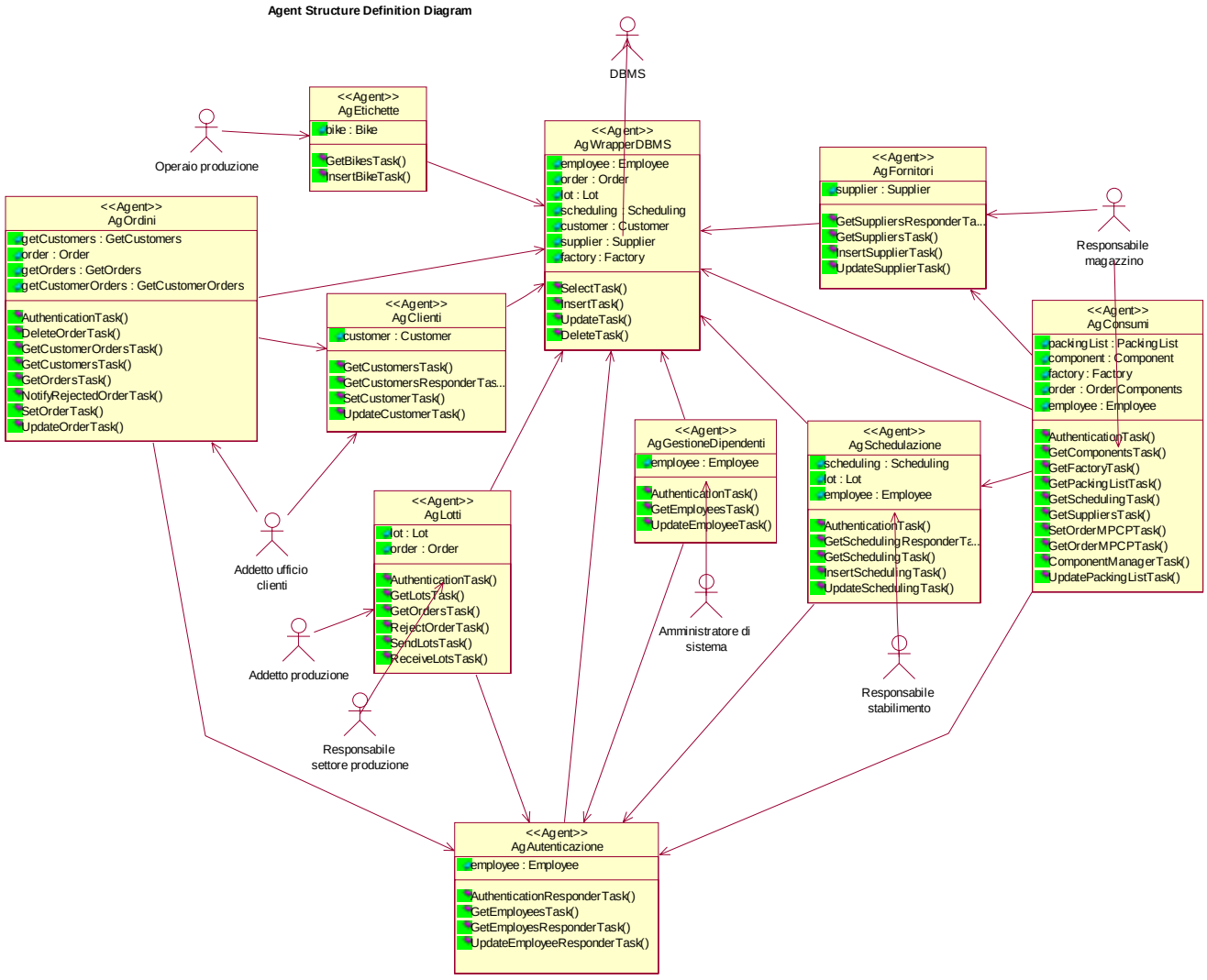
		CHANGE]
Reader	Writer	[ROLE
		CHANGE]
Logger	StoreManager	[ROLE
		CHANGE]
StoreManager	PackingListManager	[ROLE
		CHANGE]

3.7 Protocol Description phase

Per le comunicazioni tra gli agenti si è scelto di utilizzare rotocolli FIPA standard.

3.8 Multi-Agent Structure Definition phase

Questo diagramma è creato automaticamente dai precedenti e rappresenta l'intero sistema con gli agenti e gli attori coinvolti. Ogni agente è rappresentato come una classe, i propri task sono rappresentati come metodi e la propria ontologia come campi. Una associazione può rappresentare una comunicazione o una dipendenza.



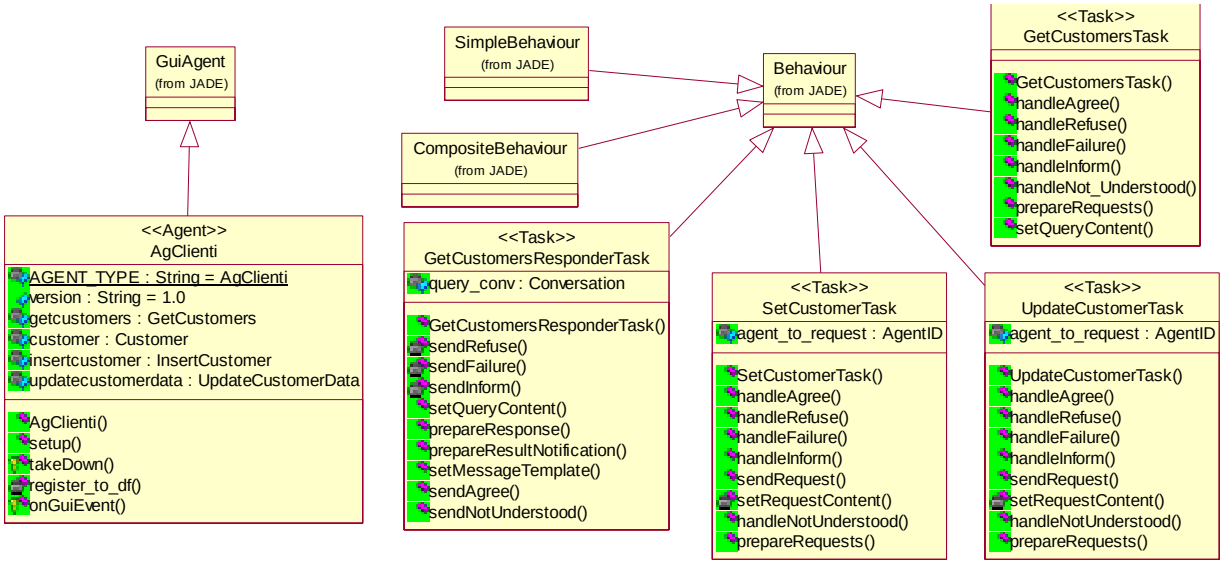
3.9 Multi-Agent Behavior Description phase

In questo diagramma rappresentiamo il flusso di eventi, i metodi invocati e i messaggi scambiati tra gli agenti.

3.10 Single Agent Structure Definition phase

3.10.1 Agent: AgClienti

Si occupa della creazione, modifica ed eliminazione dei dati relativi ai clienti.



3.10.1.1 Attributes

Class	Attribute	Data Type	Description
AgClienti	AGENT_TYPE	String	
	version	String	
	getcustomers	GetCustomers	
	customer	Customer	
	insertcustomer	InsertCustomer	
	updatecustomerdata	UpdateCustomerData	
SimpleBehaviour			
CompositeBehaviour			
GetCustomersResponderTask	query_conv	Conversation	

SetCustomerTask	agent_to_request	AgentID
UpdateCustomerTask	agent_to_request	AgentID
GetCustomersTask		
GuiAgent		

3.10.1.2 Methods

Class	Method	Parameters	Description
AgClienti	AgClienti	platform	Costruttore dell'agente
	setup	name ownership	Metodo invocato automaticamente per l'inizializzazione dell'agente
	takeDown		Metodo invocato automaticamente quando termina l'esecuzione dell'agente
	register_to_df		Metodo per la registrazione al DF
	onGuiEvent		Metodo per la gestione degli eventi gella GUI
SimpleBehaviour			
CompositeBehaviour			
GetCustomersRespond erTask	GetCustomersRespond erTask	data data	Costruttore del task
	sendRefuse sendFailure sendInform sendAgree sendNotUnderstood	data	I metodi sendXXX settano la performativa XXX

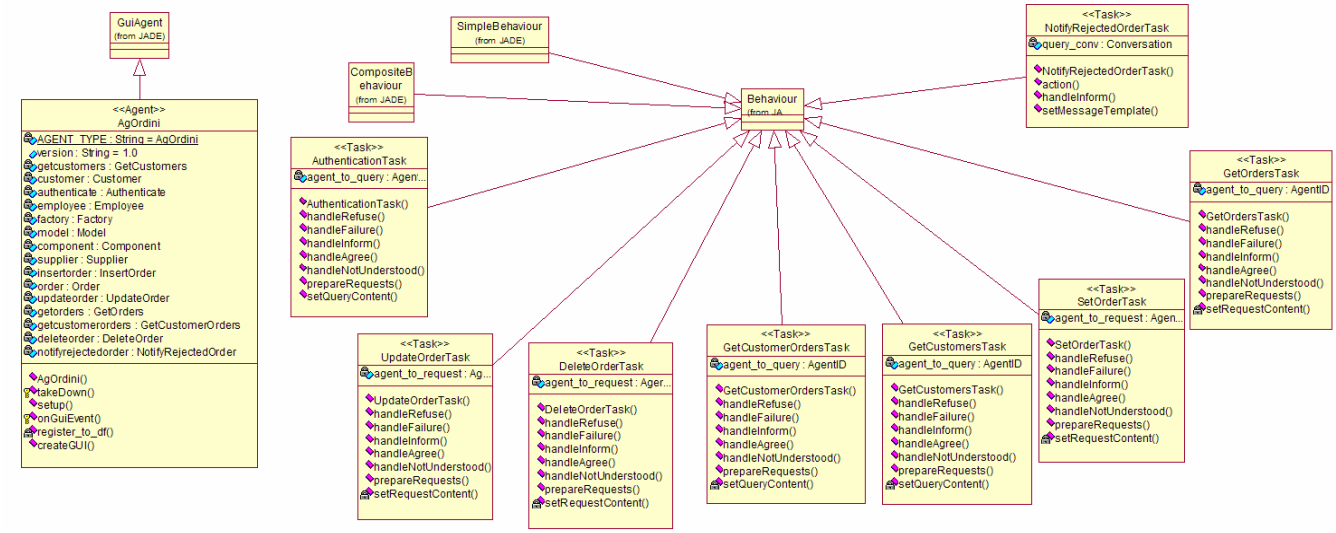
	setQueryContent		Utilizzato per settare il contenuto della query
	prepareResponse		Invia l'AGREE
	prepareResultNotification		Invia l'INFORM
	setMessageTemplate		Setta il mesage template
SetCustomerTask	SetCustomerTask	conv	Costruttore del task
	handleAgree	conv	I metodi handleXXX
	handleRefuse	conv	gestiscono la ricezione
	handleFailure	conv	di performative XXX
	handleInform	data	
	handleNotUnderstood		
	sendRequest		Setta la performativa REQUEST
	setRequestContent		Setta il contenuto del messaggio
	prepareRequests		Invia il REQUEST
UpdateCustomerTask	UpdateCustomerTask	conv	Costruttore del task
	handleAgree	conv	I metodi handleXXX
	handleRefuse	conv	gestiscono la ricezione
	handleFailure	conv	di performative XXX
	handleInform	data	
	handleNotUnderstood		
	sendRequest		Setta la performativa REQUEST
	setRequestContent		Setta il contenuto del messaggio
	prepareRequests		Invia il REQUEST
GetCustomersTask	GetCustomersTask	conv	Costruttore del task

	handleAgree	conv	I metodi handleXXX
	handleRefuse	conv	gestiscono la ricezione
	handleFailure		di performative XXX
	handleInform		
	handleNot_Understood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio

GuiAgent

3.10.2 Agent: AgOrdini

Si occupa della creazione, modifica ed eliminazione degli ordini.



3.10.2.1 Attributes

Class	Attribute	Data Type	Description
AgOrdini	AGENT_TYPE	String	
	version	String	
	getcustomers	GetCustomers	
	customer	Customer	
	authenticate	Authenticate	

	employee	Employee
	factory	Factory
	model	Model
	component	Component
	supplier	Supplier
	insertorder	InsertOrder
	order	Order
	updateorder	UpdateOrder
	getorders	GetOrders
	getcustomerorders	GetCustomerOrders
	deleteorder	DeleteOrder
	notifyrejectedorder	NotifyRejectedOrder
SimpleBehaviour		
CompositeBehaviour		
AuthenticationTask	agent_to_query	AgentID
DeleteOrderTask	agent_to_request	AgentID
GetCustomerOrdersTask	agent_to_query	AgentID
GetCustomersTask	agent_to_query	AgentID
GetOrdersTask	agent_to_query	AgentID
NotifyRejectedOrderTask	query_conv	Conversation
SetOrderTask	agent_to_request	AgentID
UpdateOrderTask	agent_to_request	AgentID
GuiAgent		

3.10.2.2 Methods

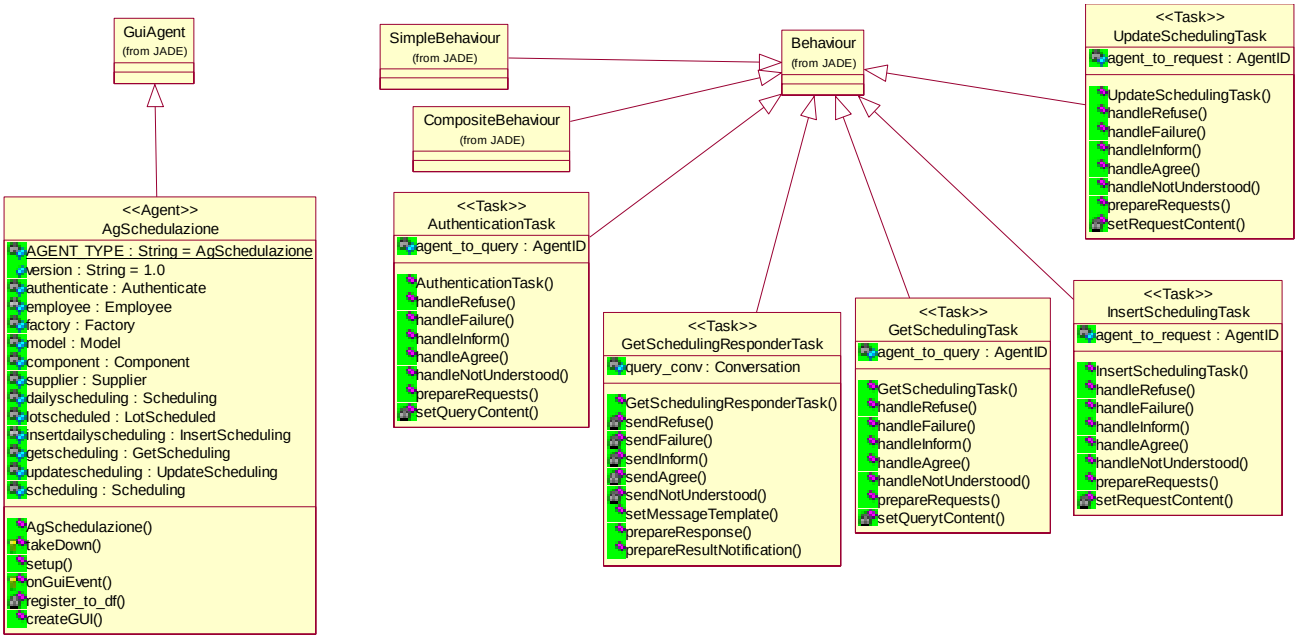
Class	Method	Parameters	Description
AgOrdini	AgOrdini	platform	Costruttore dell'agente
	setup	name ownership	Metodo invocato automaticamente per l'inizializzazione dell'agente
	takeDown		Metodo invocato automaticamente quando termina l'esecuzione dell'agente
	register_to_df		Metodo per la registrazione al DF
	onGuiEvent		Metodo per la gestione degli eventi gella GUI
	createGUI		Metodo per la creazione e l'inizializzazione della GUI
SimpleBehaviour			
CompositeBehaviour			
AuthenticationTask	AuthenticationTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
DeleteOrderTask	DeleteOrderTask	conv	Costruttore del task

	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia il REQUEST
	setRequestContent		Setta il contenuto del messaggio
GetCustomerOrdersTask	GetCustomerOrdersTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetCustomersTask	GetCustomersTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetOrdersTask	GetOrdersTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX

	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
NotifyRejectedOrderTask	NotifyRejectedOrderTask		Costruttore del task
	action		Attiva il task
	handleInform		Gestisce la ricezione dell'INFORM
	setMessageTemplate		Setta il message template
SetOrderTask	SetOrderTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la REQUEST
	setRequestContent		Setta il contenuto del messaggio
UpdateOrderTask	UpdateOrderTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la REQUEST
	setRequestContent		Setta il contenuto del messaggio
GuiAgent			

3.10.3 Agent: AgSchedulazione

Si occupa della gestione della schedulazione dei lotti di produzione.



3.10.3.1 Attributes

Class	Attribute	Data Type	Description
AgSchedulazione	AGENT_TYPE	String	
	version	String	
	authenticate	Authenticate	
	employee	Employee	
	factory	Factory	
	model	Model	
	component	Component	
	supplier	Supplier	
	dailyscheduling	Scheduling	
	lotscheduled	LotScheduled	
	insertdailyscheduling	InsertScheduling	
	getscheduling	GetScheduling	
	updatescheduling	UpdateScheduling	

	scheduling	Scheduling
SimpleBehaviour		
CompositeBehaviour		
AuthenticationTask	agent_to_query	AgentID
GetSchedulingRespond erTask	query_conv	Conversation
GetSchedulingTask	agent_to_query	AgentID
InsertSchedulingTask	agent_to_request	AgentID
UpdateSchedulingTask	agent_to_request	AgentID
GuiAgent		

3.10.3.2 Methods

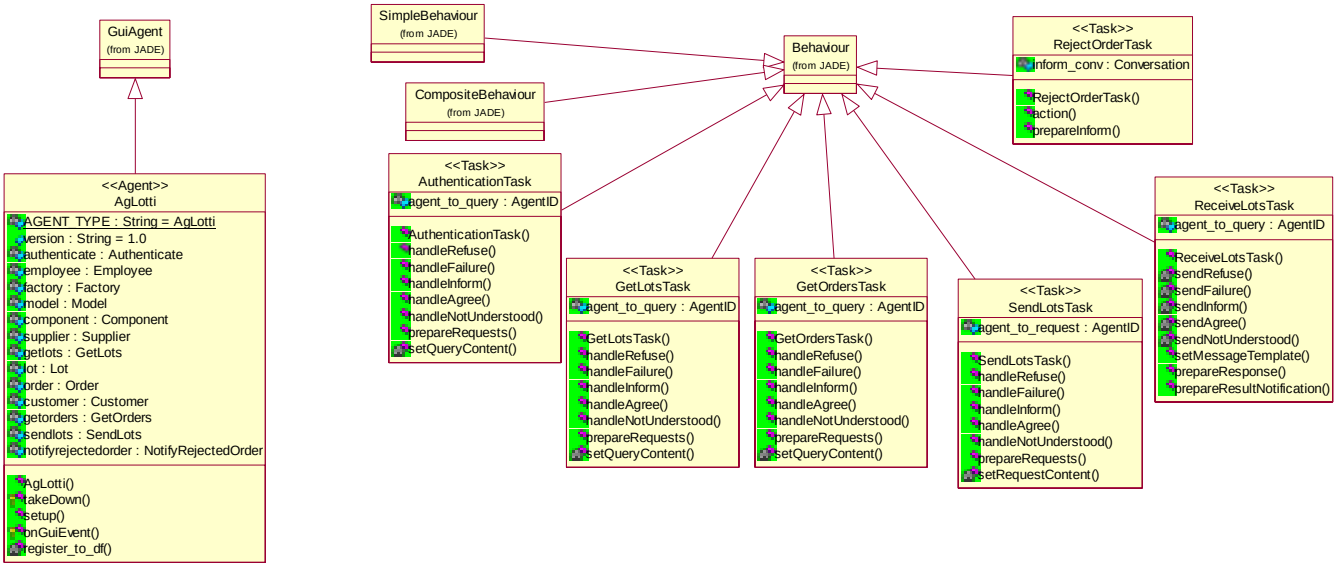
Class	Method	Parameters	Description
AgSchedulazione	AgSchedulazione	platform name ownership	Costruttore dell'agente
	setup		Metodo invocato automaticamente per l'inizializzazione dell'agente
	takeDown		Metodo invocato automaticamente quando termina l'esecuzione dell'agente
	register_to_df		Metodo per la registrazione al DF
	onGUIEvent		Metodo per la gestione degli eventi gella GUI
	createGUI		Metodo per la

			creazione e l'inizializzazione della GUI
SimpleBehaviour			
CompositeBehaviour			
AuthenticationTask	AuthenticationTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetSchedulingRespond erTask	GetSchedulingRespond erTask	data data	Costruttore del task
	sendRefuse	data	I metodi sendXXX
	sendFailure		settano la performativa
	sendInform		XXX
	sendAgree		
	sendNotUnderstood		
	prepareResponse		Invia l'AGREE
	prepareResultNotificati on		Invia l'INFORM
	setMessageTemplate		Setta il mesage template
GetSchedulingTask	GetSchedulingTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione

	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQuerytContent		Setta il contenuto del messaggio
InsertSchedulingTask	InsertSchedulingTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la REQUEST
	setRequestContent		Setta il contenuto del messaggio
UpdateSchedulingTask	UpdateSchedulingTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la REQUEST
	setRequestContent		Setta il contenuto del messaggio
GuiAgent			

3.10.4 Agent: AgLotti

Si occupa della creazione, modifica ed eliminazione dei lotti produzione.



3.10.4.1 Attributes

Class	Attribute	Data Type	Description
AgLotti	AGENT_TYPE	String	
	version	String	
	authenticate	Authenticate	
	employee	Employee	
	factory	Factory	
	model	Model	
	component	Component	
	supplier	Supplier	
	getlots	GetLots	
	lot	Lot	
	order	Order	
	customer	Customer	
	getorders	GetOrders	
	sendlots	SendLots	

	notifyrejectedorder	NotifyRejectedOrder
SimpleBehaviour		
CompositeBehaviour		
AuthenticationTask	agent_to_query	AgentID
GetLotsTask	agent_to_query	AgentID
GetOrdersTask	agent_to_query	AgentID
RejectOrderTask	inform_conv	Conversation
SendLotsTask	agent_to_request	AgentID
ReceiveLotsTask	agent_to_query	AgentID
GuiAgent		

3.10.4.2 Methods

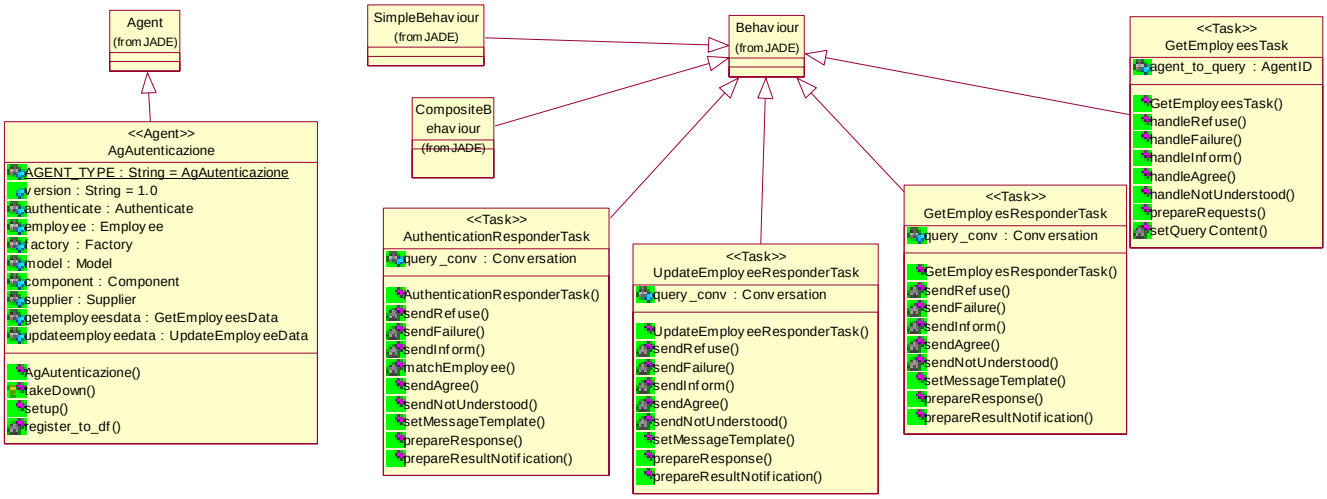
Class	Method	Parameters	Description
AgLotti	AgLotti	platform	Costruttore dell'agente
	setup	name ownership	Metodo invocato automaticamente per l'inizializzazione dell'agente
	takeDown		Metodo invocato automaticamente quando termina l'esecuzione dell'agente
	register_to_df		Metodo per la registrazione al DF
	onGuiEvent		Metodo per la gestione

			degli eventi gella GUI
	createGUI		Metodo per la creazione e l'inizializzazione della GUI
SimpleBehaviour			
CompositeBehaviour			
AuthenticationTask	AuthenticationTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetLotsTask	GetLotsTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetOrdersTask	GetOrdersTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY

	setQueryContent		Setta il contenuto del messaggio
RejectOrderTask	RejectOrderTask		Costruttore del task
	action		Attiva il task
	prepareInform		Invia l'INFORM
SendLotsTask	SendLotsTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la REQUEST
	setRequestContent		Setta il contenuto del messaggio
ReceiveLotsTask	ReceiveLotsTask	data	Costruttore del task
	sendRefuse	data	I metodi sendXXX settano la performativa XXX
	sendFailure	data	
	sendInform		
	sendAgree		
	sendNotUnderstood		
	prepareResponse		Invia l'AGREE
	prepareResultNotification		Invia l'INFORM
	setMessageTemplate		Setta il mesage template
GuiAgent			

3.10.5 Agent: AgAutenticazione

Si occupa della gestione delle autenticazioni dei dipendenti.



3.10.5.1 Attributes

Class	Attribute	Data Type	Description
AgAutenticazione	AGENT_TYPE	String	
	version	String	
	authenticate	Authenticate	
	employee	Employee	
	factory	Factory	
	model	Model	
	component	Component	
	supplier	Supplier	
	getemployeesdata	GetEmployeesData	
	updateemployeedata	UpdateEmployeeData	
SimpleBehaviour			
CompositeBehaviour			
AuthenticationResponderTask	query_conv	Conversation	
GetEmployeesTask	agent_to_query	AgentID	

GetEmployesResponderTask	query_conv	Conversation
UpdateEmployeeResponderTask	query_conv	Conversation

3.10.5.2 Methods

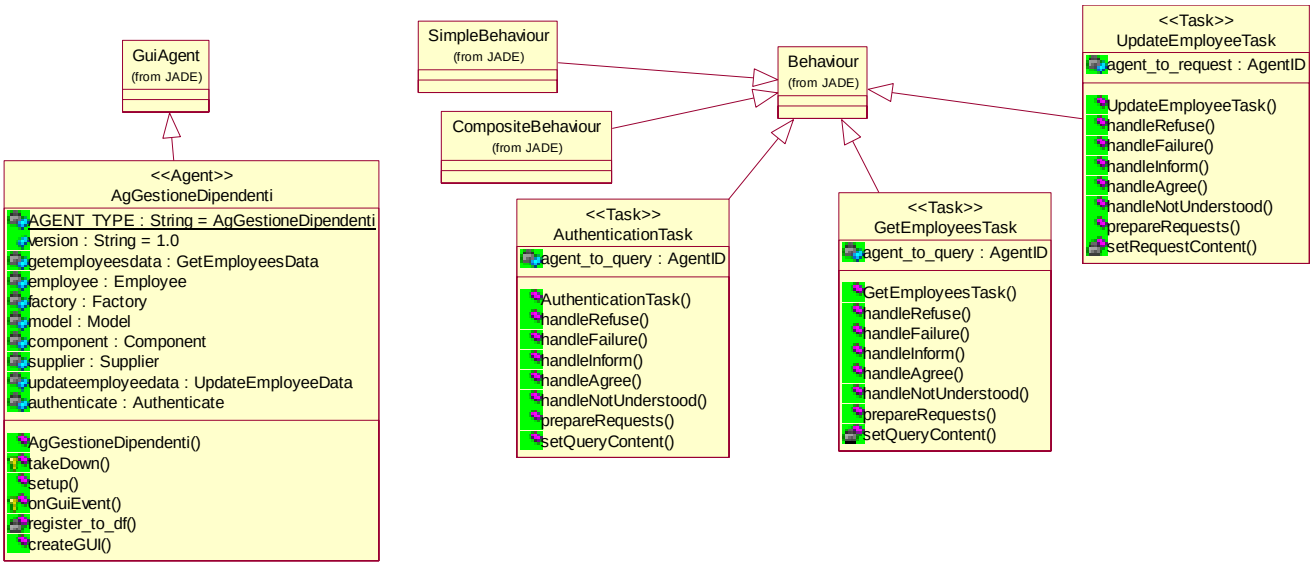
Class	Method	Parameters	Description
AgAutenticazione	AgAutenticazione	platform	Costruttore dell'agente
	setup	name ownership	Metodo invocato automaticamente per l'inizializzazione dell'agente
	takeDown		Metodo invocato automaticamente quando termina l'esecuzione dell'agente
	register_to_df		Metodo per la registrazione al DF
SimpleBehaviour			
CompositeBehaviour			
AuthenticationResponderTask	AuthenticationResponderTask	data	Costruttore del task
		data	
	sendRefuse sendFailure sendInform sendAgree sendNotUnderstood	data	I metodi sendXXX settano la performativa XXX
	matchEmployee		Verifica i dati di autenticazione
	prepareResponse		Invia l'AGREE

	prepareResultNotification		Invia l'INFORM
	setMessageTemplate		Setta il mesage template
GetEmployeesTask	GetEmployeesTask	id	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform	conv	
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetEmployesResponderTask	GetEmployesResponderTask	data	Costruttore del task
		data	
	sendRefuse	data	I metodi sendXXX settano la performativa XXX
	sendFailure		
	sendInform		
	sendAgree		
	sendNotUnderstood		
	setMessageTemplate		Setta il mesage template
	prepareResponse		Invia l'AGREE
	prepareResultNotification		Invia l'INFORM
UpdateEmployeeResponderTask	UpdateEmployeeResponderTask	data	Costruttore del task
		data	
	sendRefuse	data	I metodi sendXXX settano la performativa XXX
	sendFailure		
	sendInform		
	sendAgree		

	sendNotUnderstood	
	setMessageTemplate	Setta il message template
	prepareResponse	Invia l'AGREE
	prepareResultNotification	Invia l'INFORM

3.10.6 Agent: AgGestioneDipendenti

Si occupa della gestione dei dati dei dipendenti



3.10.6.1 Attributes

Class	Attribute	Data Type	Description
AgGestioneDipendenti	AGENT_TYPE	String	
	version	String	
	getemployeesdata	GetEmployeesData	
	employee	Employee	
	factory	Factory	
	model	Model	
	component	Component	
	supplier	Supplier	
	updateemployeedata	UpdateEmployeeData	

	authenticate	Authenticate
SimpleBehaviour		
CompositeBehaviour		
AuthenticationTask	agent_to_query	AgentID
GetEmployeesTask	agent_to_query	AgentID
UpdateEmployeeTask	agent_to_request	AgentID
GuiAgent		

3.10.6.2 Methods

Class	Method	Parameters	Description
AgGestioneDipendenti	AgGestioneDipendenti	platform name ownership	Costruttore dell'agente
	Setup		Metodo invocato automaticamente per l'inizializzazione dell'agente
	takeDown		Metodo invocato automaticamente quando termina l'esecuzione dell'agente
	register_to_df		Metodo per la registrazione al DF
	onGuiEvent		Metodo per la gestione degli eventi gella GUI
	createGUI		Metodo per la creazione e l'inizializzazione della GUI
SimpleBehaviour			

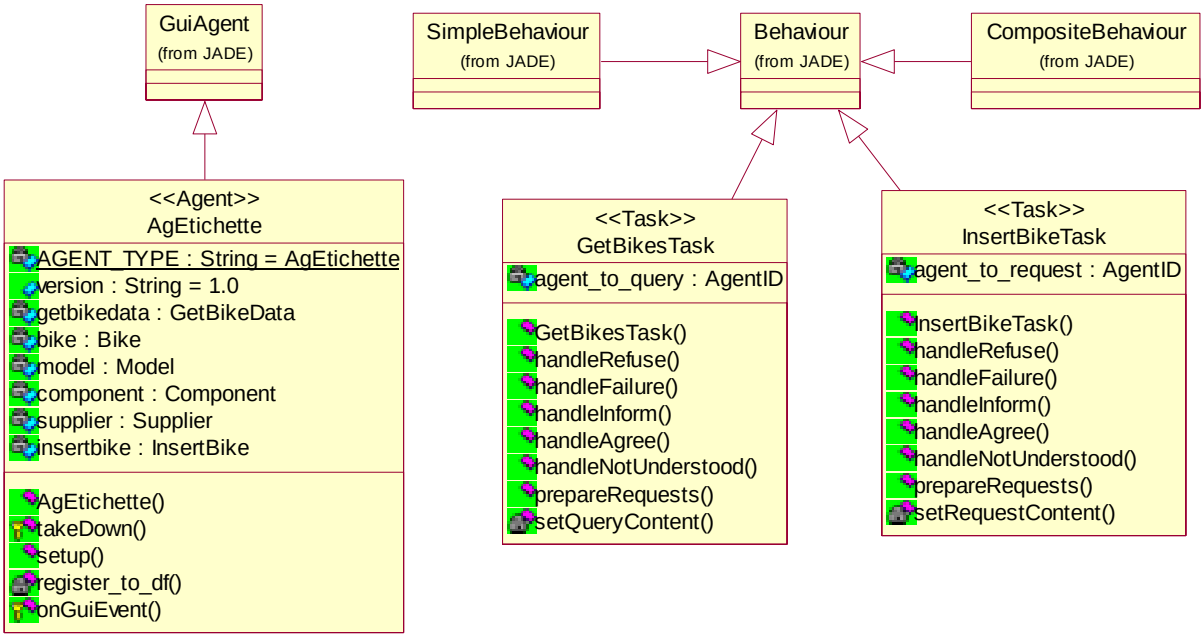
CompositeBehaviour

AuthenticationTask	AuthenticationTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetEmployeesTask	GetEmployeesTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
UpdateEmployeeTask	UpdateEmployeeTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setRequestContent		Setta il contenuto del messaggio

GuiAgent

3.10.7 Agent: AgEtichette

Si occupa della visualizzazione e stampa delle etichette



3.10.7.1 Attributes

Class	Attribute	Data Type	Description
AgEtichette	AGENT_TYPE	String	
	version	String	
	getbikedata	GetBikeData	
	bike	Bike	
	model	Model	
	component	Component	
	supplier	Supplier	
	insertbike	InsertBike	
SimpleBehaviour			
CompositeBehaviour			
GetBikesTask	agent_to_query	AgentID	
InsertBikeTask	agent_to_request	AgentID	

GuiAgent

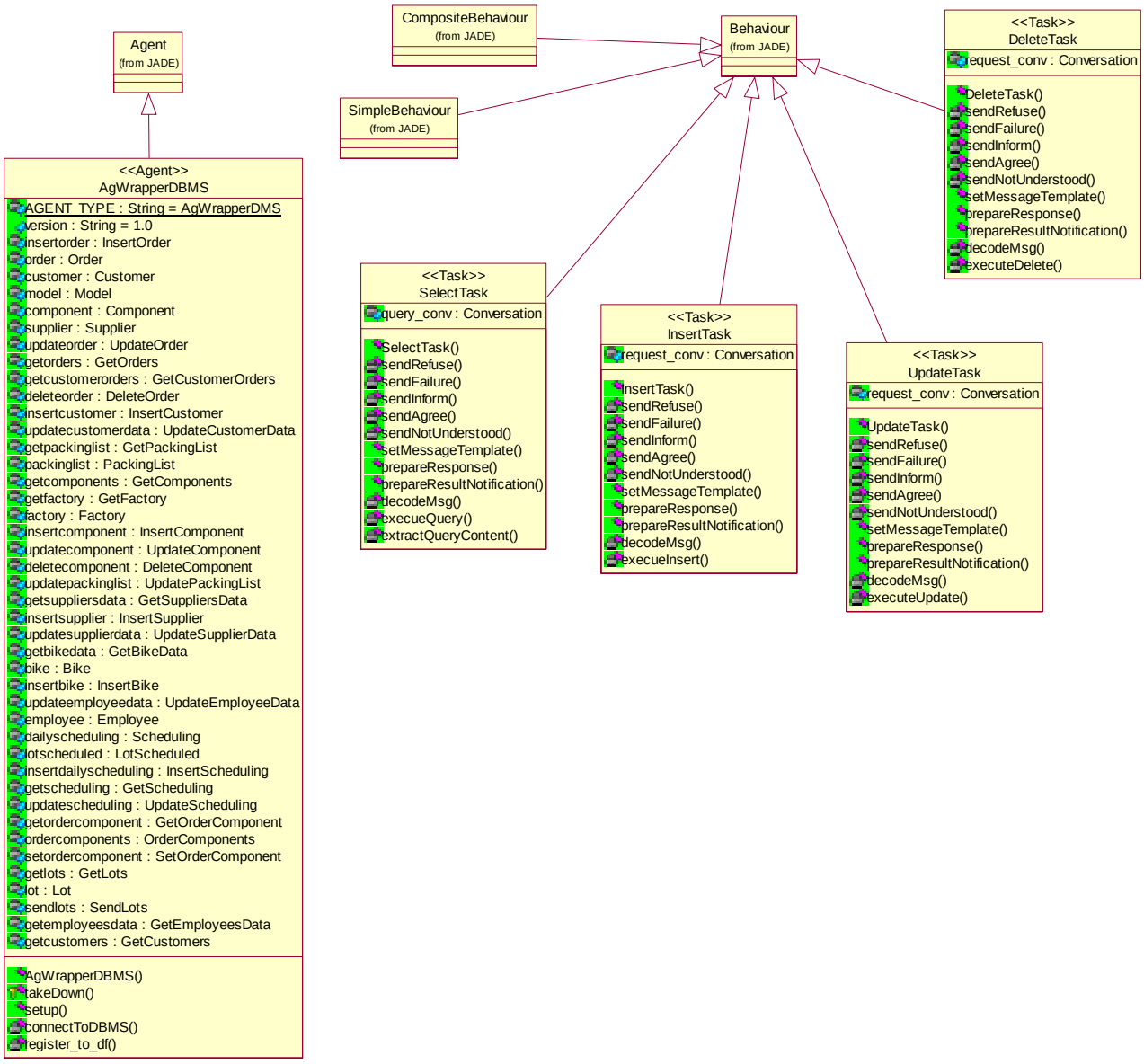
3.10.7.2 Methods

Class	Method	Parameters	Description
AgEtichette	AgEtichette	platform	Costruttore dell'agente
	setup	name ownership	Metodo invocato automaticamente per l'inizializzazione dell'agente
	takeDown		Metodo invocato automaticamente quando termina l'esecuzione dell'agente
	register_to_df		Metodo per la registrazione al DF
	onGuiEvent		Metodo per la gestione degli eventi gella GUI
			Metodo per la creazione e l'inizializzazione della GUI
SimpleBehaviour			
CompositeBehaviour			
GetBikesTask	GetBikesTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio

InsertBikeTask	InsertBikeTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia il REQUEST
	setRequestContent		Setta il contenuto del messaggio
GuiAgent			

3.10.8 Agent: AgWrapperDBMS

Si occupa dell'interfacciamento degli agenti con il DBMS.



3.10.8.1 Attributes

Class	Attribute	Data Type	Description
AgWrapperDBMS	AGENT_TYPE	String	
	version	String	
	insertorder	InsertOrder	

order	Order
customer	Customer
model	Model
component	Component
supplier	Supplier
updateorder	UpdateOrder
getorders	GetOrders
getcustomerorders	GetCustomerOrders
deleteorder	DeleteOrder
insertcustomer	InsertCustomer
updatecustomerdata	UpdateCustomerData
getpackinglist	GetPackingList
packinglist	PackingList
getcomponents	GetComponents
getfactory	GetFactory
factory	Factory
insertcomponent	InsertComponent
updatecomponent	UpdateComponent
deletecomponent	DeleteComponent
updatepackinglist	UpdatePackingList
getsuppliersdata	GetSuppliersData
insertsupplier	InsertSupplier
updatesupplierdata	UpdateSupplierData
getbikedata	GetBikeData
bike	Bike
insertbike	InsertBike
updateemployeedata	UpdateEmployeeData
employee	Employee
dailyscheduling	Scheduling
lotscheduled	LotScheduled
insertdailyscheduling	InsertScheduling
getscheduling	GetScheduling
updatescheduling	UpdateScheduling
getordercomponent	GetOrderComponent

ordercomponents	OrderComponents	
setordercomponent	SetOrderComponent	
getlots	GetLots	
lot	Lot	
sendlots	SendLots	
getemployeesdata	GetEmployeesData	
getcustomers	GetCustomers	

SimpleBehaviour

CompositeBehaviour

SelectTask	query_conv	Conversation
InsertTask	request_conv	Conversation
UpdateTask	request_conv	Conversation
DeleteTask	request_conv	Conversation

3.10.8.2 Methods

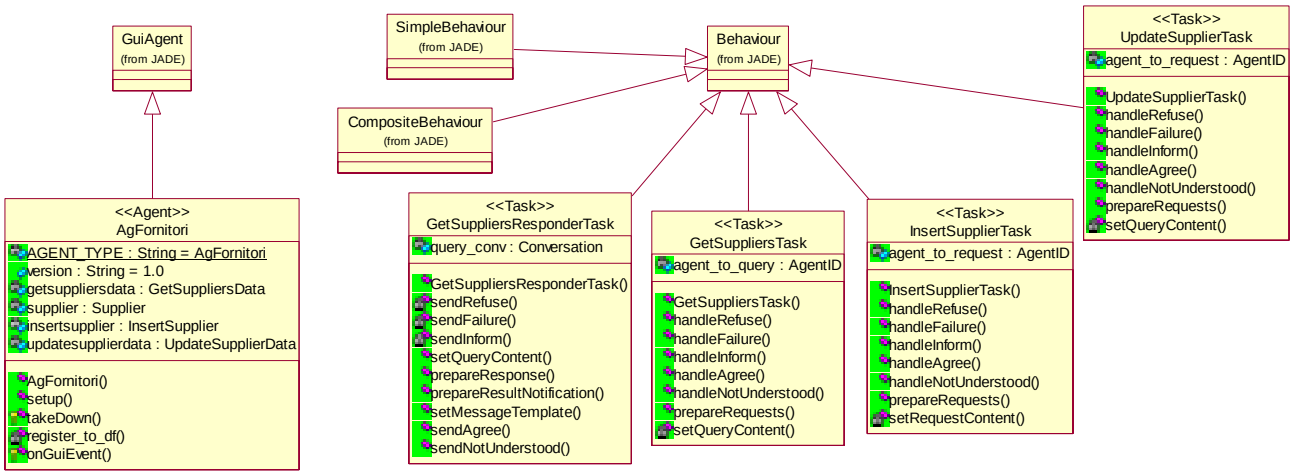
Class	Method	Parameters	Description
AgWrapperDBMS	AgWrapperDBMS	platform	Costruttore dell'agente
	setup	name ownership	Metodo invocato automaticamente per l'inizializzazione dell'agente
	takeDown		Metodo invocato automaticamente quando termina l'esecuzione dell'agente
	register_to_df		Metodo per la registrazione al DF

	connectToDBMS		Metodo utilizzato per la connessione dell'agente al DBMS
SimpleBehaviour			
CompositeBehaviour			
SelectTask	SelectTask	data	Costruttore del task
	sendRefuse	data	I metodi sendXXX settano la performativa XXX
	sendFailure	data	
	sendInform		
	sendAgree		
	sendNotUnderstood		
	setMessageTemplate		Setta il message template
	prepareResponse		Invia l'AGREE
	prepareResultNotification		Invia l'INFORM
	decodeMsg		Decodifica il messaggio ricevuto
	execueQuery		Esegue la query al DBMS
	extractQueryContent		Estrae il contenuto della query.
InsertTask	InsertTask	data	Costruttore del task
	sendRefuse	data	I metodi sendXXX settano la performativa XXX
	sendFailure	data	
	sendInform		
	sendAgree		
	sendNotUnderstood		
	setMessageTemplate		Setta il message template
	prepareResponse		Invia l'AGREE
	prepareResultNotification		Invia l'INFORM
	decodeMsg		Decodifica il

	execueInsert		messaggio ricevuto Esegue l'insert al DBMS
UpdateTask	UpdateTask	data	Costruttore del task
	sendRefuse	data	I metodi sendXXX
	sendFailure	data	settano la performativa
	sendInform		XXX
	sendAgree		
	sendNotUnderstood		
	setMessageTemplate		Setta il message template
	prepareResponse		Invia l'AGREE
	prepareResultNotification		Invia l'INFORM
	decodeMsg		Decodifica il messaggio ricevuto
	executeUpdate		Esegue l'update al DBMS
DeleteTask	DeleteTask	data	Costruttore del task
	sendRefuse	data	I metodi sendXXX
	sendFailure	data	settano la performativa
	sendInform		XXX
	sendAgree		
	sendNotUnderstood		
	setMessageTemplate		Setta il message template
	prepareResponse		Invia l'AGREE
	prepareResultNotification		Invia l'INFORM
	decodeMsg		Decodifica il messaggio ricevuto
	executeDelete		Esegue il delete al DBMS

3.10.9 Agent: AgFornitori

Si occupa della creazione, modifica ed eliminazione dei dati sui fornitori.



3.10.9.1 Attributes

Class	Attribute	Data Type	Description
AgFornitori	AGENT_TYPE	String	
	version	String	
	getsuppliersdata	GetSuppliersData	
	supplier	Supplier	
	insertsupplier	InsertSupplier	
	updatesupplierdata	UpdateSupplierData	
SimpleBehaviour			
CompositeBehaviour			
GetSuppliersResponderTask	query_conv	Conversation	
GetSuppliersTask	agent_to_query	AgentID	
InsertSupplierTask	agent_to_request	AgentID	
UpdateSupplierTask	agent_to_request	AgentID	
GuiAgent			

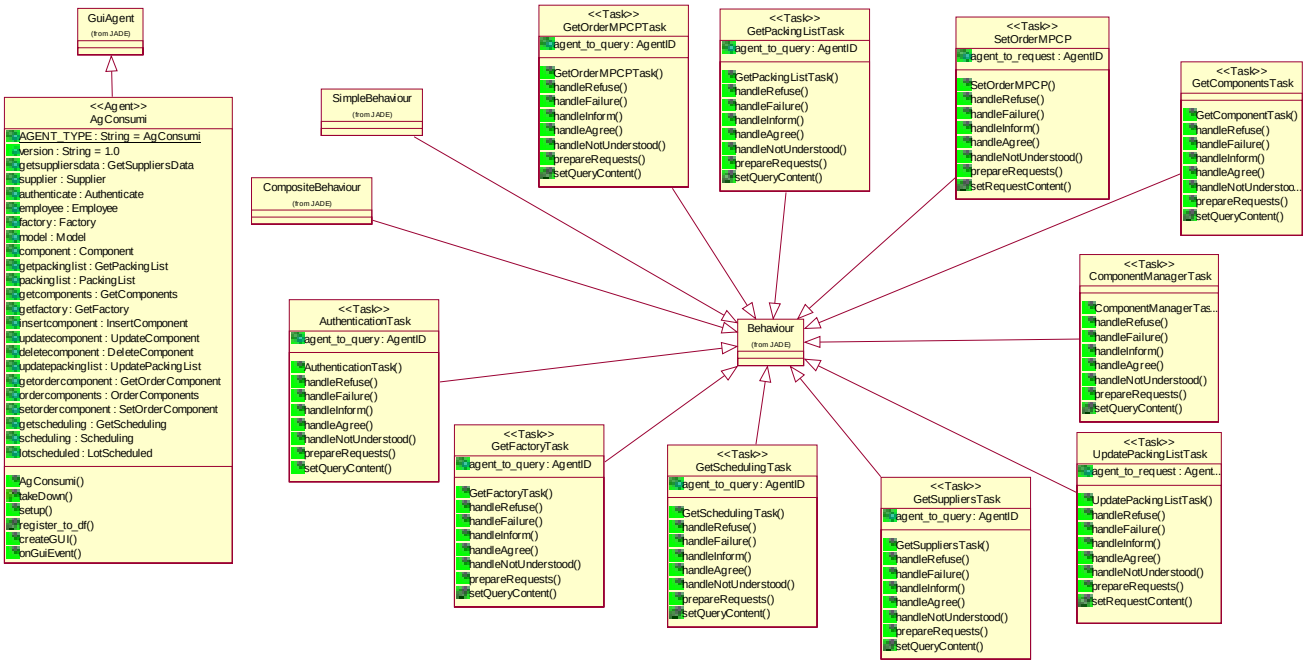
3.10.9.2 Methods

Class	Method	Parameters	Description
AgFornitori	AgFornitori	platform	Costruttore dell'agente
	setup	name ownership	Metodo invocato automaticamente per l'inizializzazione dell'agente
	takeDown		Metodo invocato automaticamente quando l'esecuzione dell'agente termina
	register_to_df		Metodo per la registrazione al DF
	onGuiEvent		Metodo per la gestione degli eventi gella GUI
	createGUI		Metodo per la creazione e l'inizializzazione della GUI
SimpleBehaviour			
CompositeBehaviour			
GetSuppliersResponder Task	GetSuppliersResponder Task	data data	Costruttore del task
	sendRefuse sendFailure sendInform sendAgree sendNotUnderstood	data	I metodi sendXXX settano la performativa XXX
	setQueryContent		Setta il contenuto della query
	prepareResponse		Invia l'AGREE
	prepareResultNotification		Invia l'INFORM

	setMessageTemplate		Setta il message template
GetSuppliersTask	GetSuppliersTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
InsertSupplierTask	InsertSupplierTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la REQUEST
	setRequestContent		Setta il contenuto del messaggio
UpdateSupplierTask	UpdateSupplierTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la REQUEST
	setQueryContent		Setta il contenuto del messaggio
GuiAgent			

3.10.10 Agent: AgConsumi

Si occupa della gestione degli approvvigionamenti. Permette la creazione, modifica ed elimazione dei dati su componenti, distinta di produzione ed ordine materiali.



3.10.10.1 Attributes

Class	Attribute	Data Type	Description
AgConsumi	AGENT_TYPE	String	
	version	String	
	getsuppliersdata	GetSuppliersData	
	supplier	Supplier	
	authenticate	Authenticate	
	employee	Employee	
	factory	Factory	
	model	Model	
	component	Component	
	getpackinglist	GetPackingList	
	packinglist	PackingList	

	getcomponents	GetComponents
	getfactory	GetFactory
	insertcomponent	InsertComponent
	updatecomponent	UpdateComponent
	deletecomponent	DeleteComponent
	updatepackinglist	UpdatePackingList
	getordercomponent	GetOrderComponent
	ordercomponents	OrderComponents
	setordercomponent	SetOrderComponent
	getscheduling	GetScheduling
	scheduling	Scheduling
	lotscheduled	LotScheduled
SimpleBehaviour		
CompositeBehaviour		
AuthenticationTask	agent_to_query	AgentID
GetFactoryTask	agent_to_query	AgentID
GetOrderMPCPTask	agent_to_query	AgentID
GetPackingListTask	agent_to_query	AgentID
GetSchedulingTask	agent_to_query	AgentID
GetSuppliersTask	agent_to_query	AgentID
SetOrderMPCP	agent_to_request	AgentID
UpdatePackingListTask	agent_to_request	AgentID
GetComponentsTask		
ComponentManagerTask		
GuiAgent		

3.10.10.2 Methods

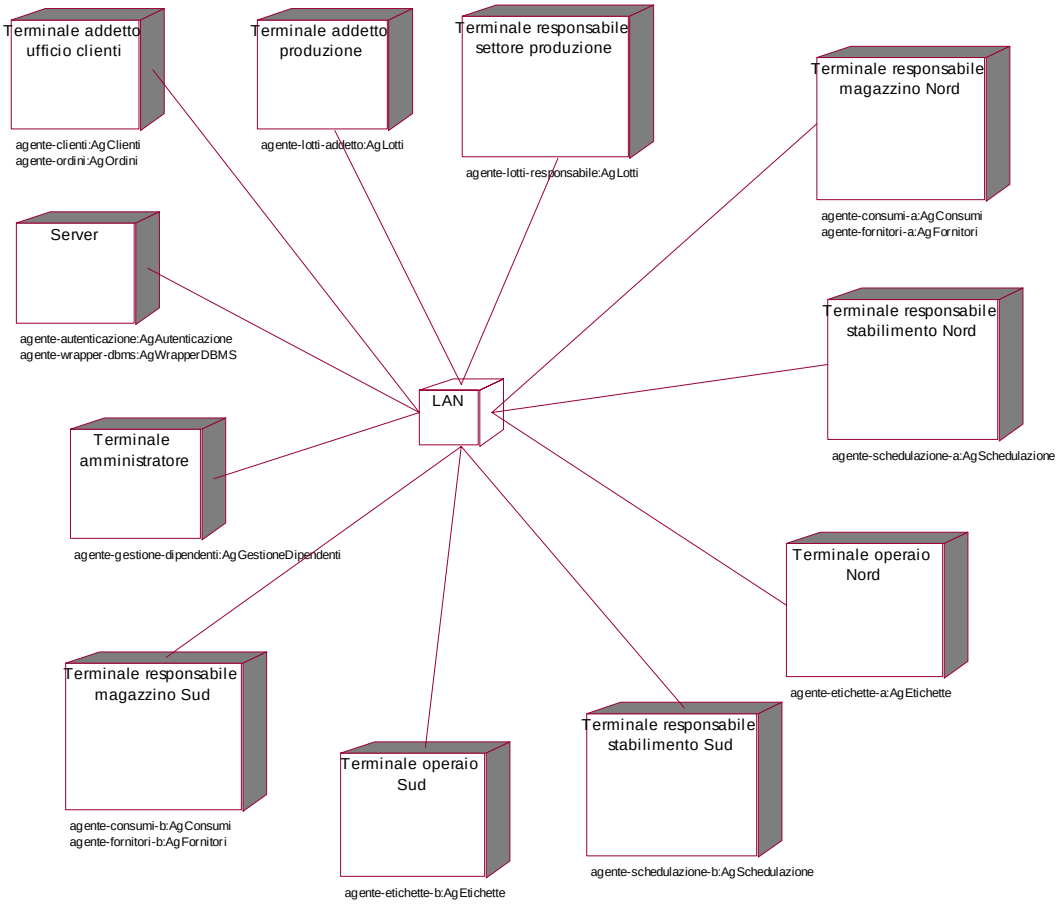
Class	Method	Parameters	Description
AgConsumi	AgConsumi	platform	Costruttore dell'agente
	Setup	name ownership	Metodo invocato automaticamente per l'inizializzazione dell'agente
	takeDown		Metodo invocato automaticamente quando termina l'esecuzione dell'agente
	register_to_df		Metodo per la registrazione al DF
	createGUI		Metodo per la gestione degli eventi gella GUI
	onGuiEvent		Metodo per la creazione e l'inizializzazione della GUI
SimpleBehaviour			
CompositeBehaviour			
AuthenticationTask	AuthenticationTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetFactoryTask	GetFactoryTask	conv	Costruttore del task

	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetOrderMPCPTask	GetOrderMPCPTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetPackingListTask	GetPackingListTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
GetSchedulingTask	GetSchedulingTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY

	setQueryContent		Setta il contenuto del messaggio
GetSuppliersTask	GetSuppliersTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
SetOrderMPCP	SetOrderMPCP	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la REQUEST
	setRequestContent		Setta il contenuto del messaggio
UpdatePackingListTask	UpdatePackingListTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX
	handleFailure	conv	gestiscono la ricezione
	handleInform		di performative XXX
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la REQUEST
	setRequestContent		Setta il contenuto del messaggio
GetComponentsTask	GetComponentTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX

	handleFailure	conv	gestiscono la ricezione di performative XXX
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la QUERY
	setQueryContent		Setta il contenuto del messaggio
ComponentManagerTask	ComponentManagerTask	conv	Costruttore del task
	handleRefuse	conv	I metodi handleXXX gestiscono la ricezione di performative XXX
	handleFailure	conv	
	handleInform		
	handleAgree		
	handleNotUnderstood		
	prepareRequests		Invia la REQUEST
	setQueryContent		Setta il contenuto del messaggio
GuiAgent			

3.11 Deployment Configuration phase



Terminale addetto ufficio clienti

- agente-clienti:AgClienti
- agente-ordini:AgOrdini

Terminale addetto produzione

- agente-lotti-addetto:AgLotti

Terminale responsabile settore produzione

- agente-lotti-responsabile:AgLotti

Server

- agente-autenticazione:AgAutenticazione
- agente-wrapper-dbms:AgWrapperDBMS

Terminale amministratore

- agente-gestione-dipendenti:AgGestioneDipendenti

Terminale operaio Sud

- agente-etichette-b:AgEtichette

Terminale responsabile stabilimento Nord

- agente-schedulazione-a:AgSchedulazione

Terminale responsabile magazzino Nord

- agente-consumi-a:AgConsumi
- agente-fornitori-a:AgFornitori

Terminale operaio Nord

- agente-etichette-a:AgEtichette

Terminale responsabile stabilimento Sud

- agente-schedulazione-b:AgSchedulazione

Terminale responsabile magazzino Sud

- agente-consumi-b:AgConsumi
- agente-fornitori-b:AgFornitori

PROGETTO DATABASE

4 Progetto database

4.1 Progetto concettuale

4.1.1 Descrizione verbale

In questa fase procediamo alla descrizione dell'archivio ed identifichiamo i principali elementi che in esso si relazionano. Si utilizza la seguente notazione per individuare gli elementi del dominio:

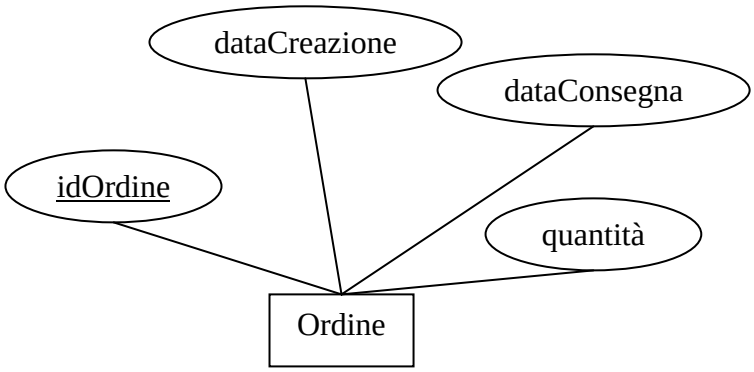
- le entità, rappresentate da sostantivi (in rosso)
- gli i attributi (delle entità) sono identificabili come proprietà delle entità (in verde).
- le relazioni sono verbi che collegano più entità (in blu)

Si intende realizzare un archivio per un sistema di gestione di un'industria di biciclette.
Ogni ordine è caratterizzato da un identificatore, dalla data di creazione, dalla data di consegna e dalla quantità. Ogni ordine è effettuato da un cliente. Ogni ordine è inserito in un lotto. Ogni ordine fa riferimento ad un modello.
Ogni cliente è caratterizzato da un identificatore, nome, cognome, nome della compagnia a cui appartiene, numero di telefono, indirizzo e-mail, partita Iva, indirizzo.
Ogni lotto è caratterizzato da un identificatore, data di creazione, data di scadenza, la quantità di biciclette da produrre. Ogni lotto è prodotto in una o più schedulazioni giornaliere.
Ogni schedulazione è caratterizzata dalla quantità di un lotto da schedulare e dalla data di schedulazione.
Ogni stabilimento è caratterizzato da un identificatore, da un indirizzo e da un nome.
Ogni modello è caratterizzato da un nome che lo identifica e da una descrizione. Ogni modello è prodotto in uno stabilimento. Ogni modello è composto da un certo numero di componenti.
Ogni bicicletta è caratterizzata da un numero di telaio e dalla data di produzione. Ogni bicicletta è prodotta all'interno di un lotto. Ogni bicicletta è rappresentativa di un modello. Ogni bicicletta è ordinata da un cliente.
Ogni componente è caratterizzato da un identificatore, da un nome e da una descrizione. Ogni componente compare in uno o più modelli. Ogni componente è fornito da un fornitore.
Ogni fornitore è caratterizzato da un identificatore, nome, cognome, nome della compagnia a cui appartiene, numero di telefono, indirizzo e-mail, partita Iva, indirizzo.
Ogni componente è inserito in più ordini di componenti con una quantità calcolata in base alla produzione.

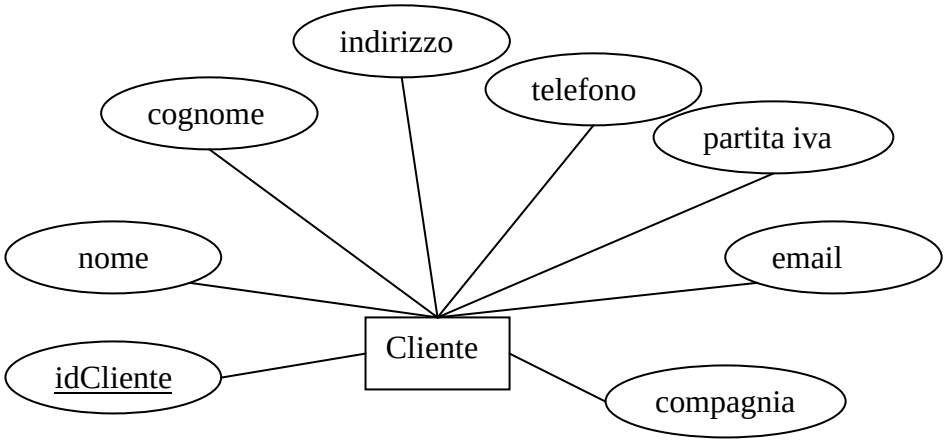
Ogni ordine di componenti è caratterizzato dalla data di emissione e dalla data di consegna richiesta. Ogni ordine di componenti fa riferimento ad uno stabilimento.
Ogni impiegato è caratterizzato da un identificatore, dal proprio nome, dal cognome, dall'indirizzo, dal numero di telefono, da un indirizzo e-mail, da un ruolo, dall'username e dalla password. Ogni impiegato lavora in uno stabilimento.

4.1.2 ERD

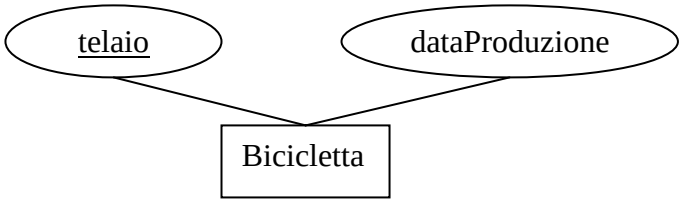
Entità ordine:



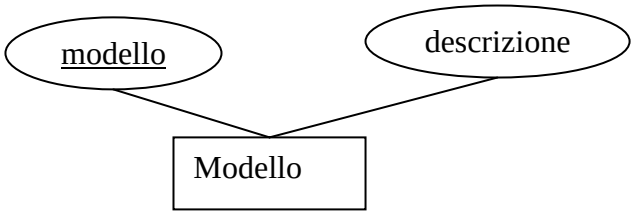
Entità cliente:



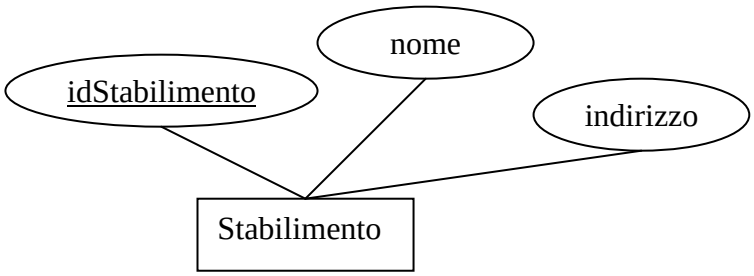
Entità bicicletta:



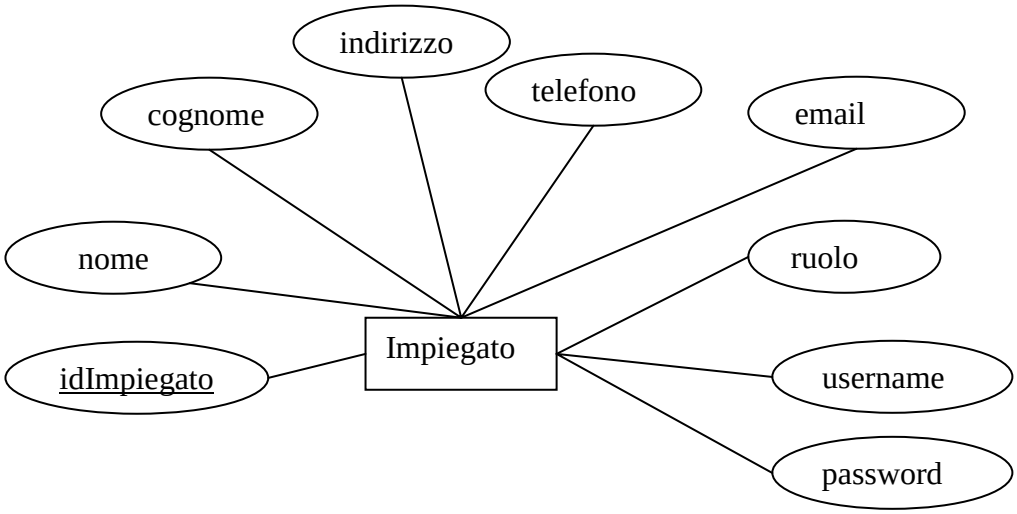
Entità modello:



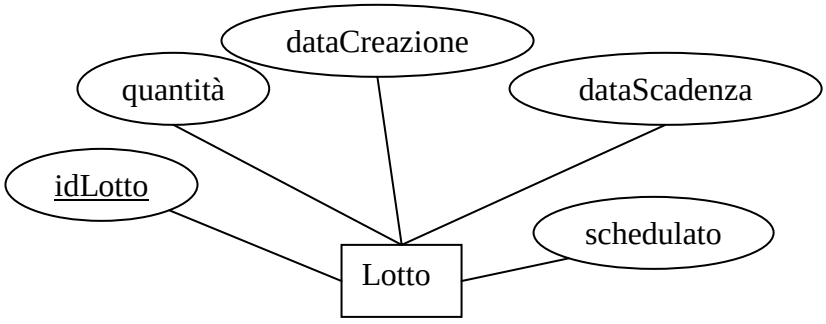
Entità stabilimento:



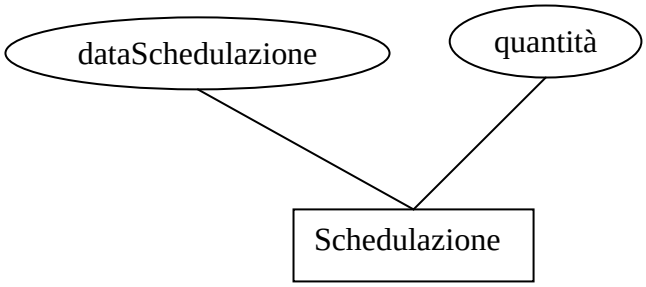
Entità impiegato:



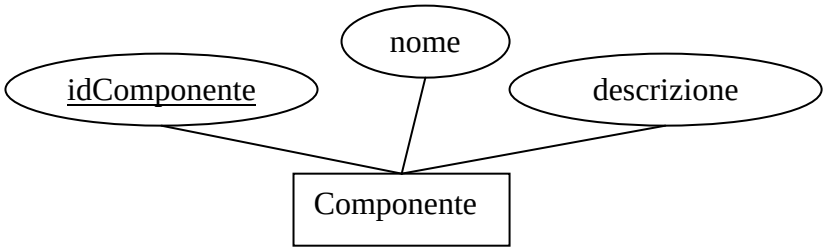
Entità lotto:



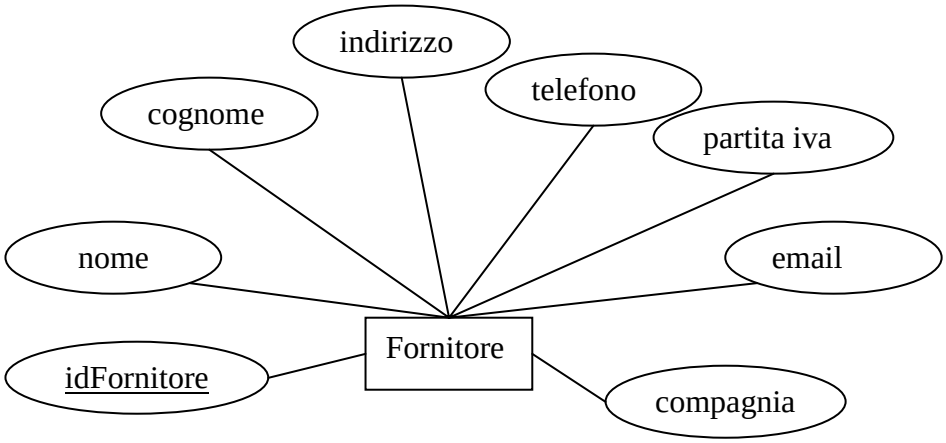
Entità schedulazione:



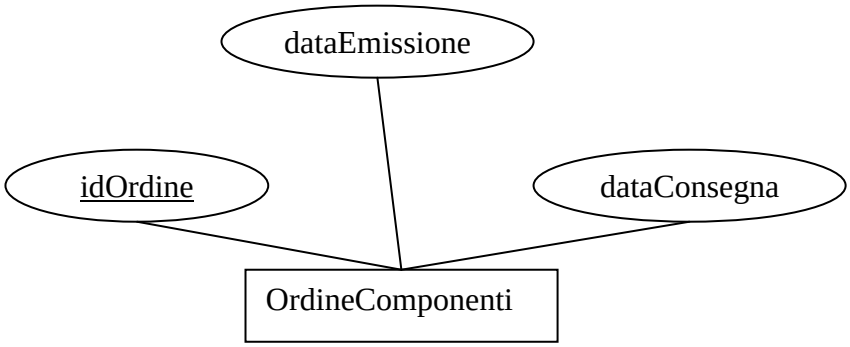
Entità componente:



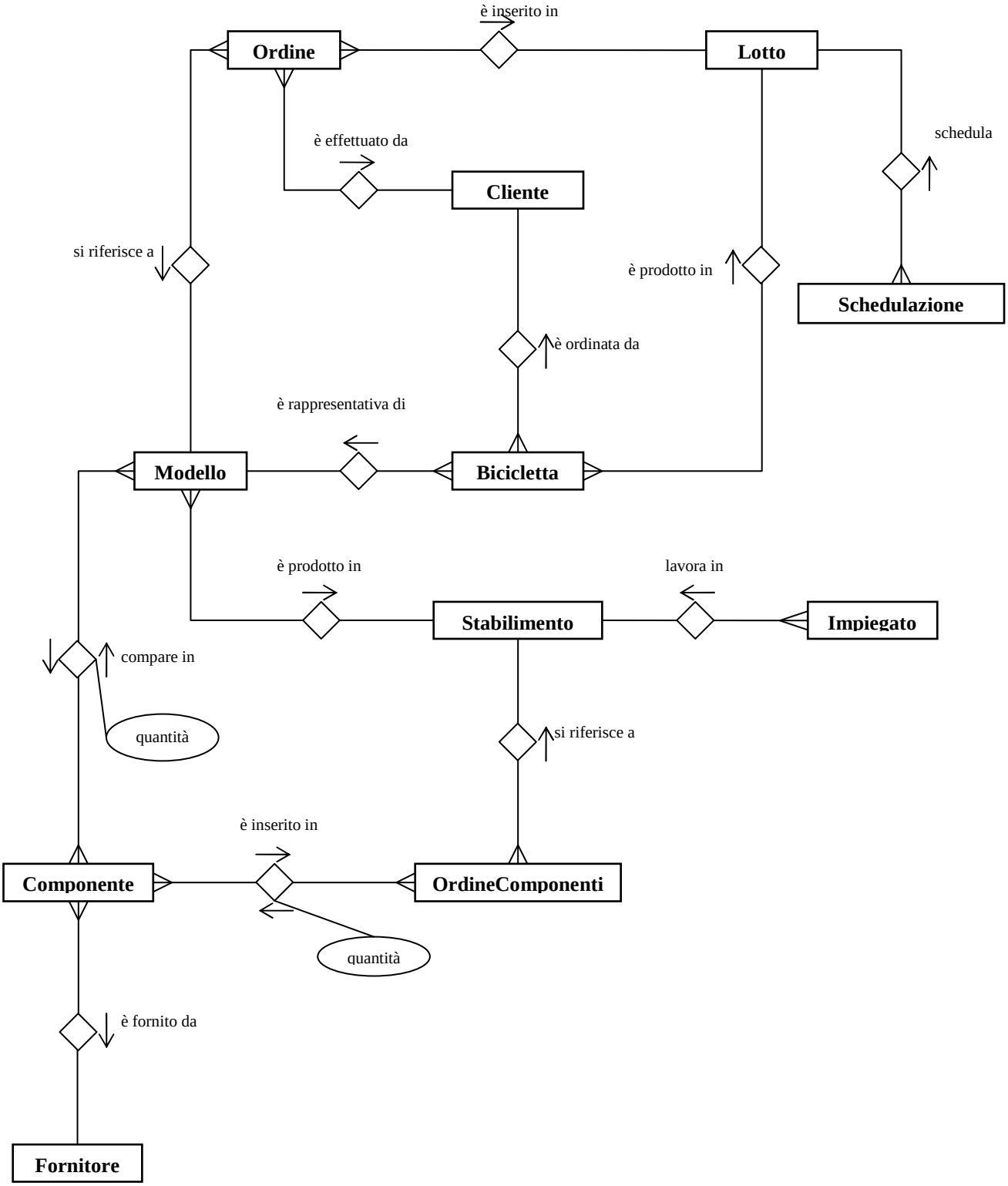
Entità fornitore:



Entità ordine componenti:

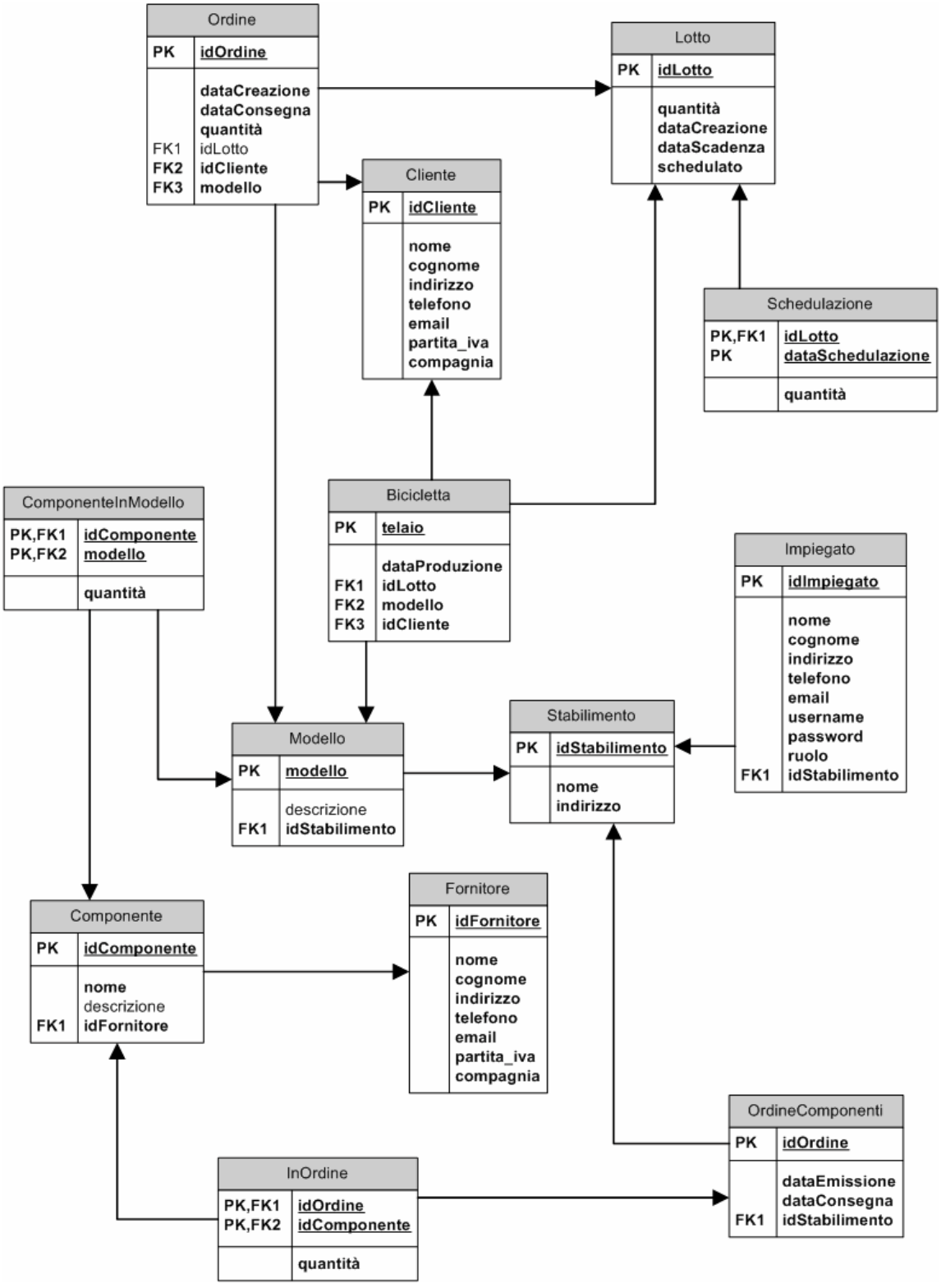


Dalla descrizione prima vista si può trarre il seguente ERD:



4.2 Progetto logico

Dall'ERD prima visto, si possono trarre le seguenti tabelle nelle quali sono state aggiunte le chiavi esterne necessarie alle relazioni:



4.2.1 Tabelle

Ordine

CAMPO	TIPO	DESCRIZIONE
idOrdine	Contatore	Identifica l'ordine
dataCreazione	Data (richiesto: si, indice: no)	Indica la data di creazione dell'ordine
dataConsegna	Data (richiesto: si, indice: no)	Indica la data di consegna dell'ordine
quantità	Intero (richiesto: si, indice: no)	Indica la quantità delle biciclette ordinate
idLotto	Intero lungo	Identifica il lotto in cui l'ordine è inserito
idCliente	Intero lungo	Identifica il cliente che ha effettuato l'ordine
modello	Intero lungo	Identifica il modello a cui l'ordine si riferisce

Entità cliente:

CAMPO	TIPO	DESCRIZIONE
idCliente	Contatore	Identifica il cliente
nome	Testo(50) (richiesto: si, indice: no)	Indica il nome del cliente
cognome	Testo(50) (richiesto: si, indice: no)	Indica il cognome del cliente
indirizzo	Testo(50) (richiesto: si, indice: no)	Indica l'indirizzo del cliente
telefono	Testo(20) (richiesto: si, indice: no)	Indica il numero di telefono del cliente
email	Testo(20) (richiesto: si, indice: no)	Indica l'indirizzo e-mail del cliente
partita_iva	Testo(20) (richiesto: si, indice: no)	Indica la partita iva del cliente
compagnia	Testo(50) (richiesto: si, indice: si)	Indica il nome della compagnia del cliente

Entità bicicletta:

CAMPO	TIPO	DESCRIZIONE
telaio	Contatore	Identifica la bicicletta
dataProduzione	Data (richiesto: si, indice: no)	Indica la in cui la bicicletta è stata prodotta
idLotto	Intero lungo	Identifica il lotto in cui la bicicletta è stata prodotta
modello	Carattere(1)	Identifica il modello di bicicletta prodotto
idCliente	Intero lungo	Identifica il cliente che ha ordinato la bicicletta

Entità modello:

CAMPO	TIPO	DESCRIZIONE
modello	Carattere(1)	Identifica il modello
descrizione	Testo(50) (richiesto: no, indice: no)	Descrizione verbale delle caratteristiche del modello.
idStabilimento	Intero lungo	Identifica lo stabilimento che produce il modello

Entità stabilimento:

CAMPO	TIPO	DESCRIZIONE
idStabilimento	Contatore	Identifica lo stabilimento
nome	Testo(50) (richiesto: si, indice: si)	Indica il nome dello stabilimento
indirizzo	Testo(50) (richiesto: si, indice: si)	Indica l'indirizzo dello stabilimento

Entità impiegato:

CAMPO	TIPO	DESCRIZIONE
idImpiegato	Contatore	Identifica l'impiegato
nome	Testo(50) (richiesto: si, indice: no)	Indica il nome dell'impiegato
cognome	Testo(50) (richiesto: si, indice: no)	Indica il cognome dell'impiegato
indirizzo	Testo(50) (richiesto: si, indice: no)	Indica l'indirizzo dell'impiegato
telefono	Testo(20) (richiesto: si, indice: no)	Indica il numero di telefono dell'impiegato
email	Testo(20) (richiesto: si, indice: no)	Indica l'indirizzo e-mail dell'impiegato
username	Testo(8) (richiesto: si, indice: no)	Indica l'username dell'impiegato
password	Testo(8) (richiesto: si, indice: no)	Indica la parrword dell'impiegato
ruolo	Testo(50) (richiesto: si, indice: no)	Indica il ruolo del'impiegato
idStabilimento	Intero lungo	Identifica lo stabilimento in cui l'impiegato lavora

Entità lotto:

CAMPO	TIPO	DESCRIZIONE
idLotto	Contatore	Identifica il lotto
quantità	Intero(richiesto: si, indice: si)	Indica la quantità di biciclette di cui il lotto è composto
dataCreazione	Data (richiesto: si, indice: no)	Indica la data di creazione del lotto
dataScadenza	Data (richiesto: si, indice: no)	Indica la data massima di produzione del lotto
schedulato	Si/No (richiesto: si)	Indica se il lotto è stato schedulato

Entità schedulazione:

CAMPO	TIPO	DESCRIZIONE
idLotto	Intero lungo	Identifica il lotto da schedulare nel giorno
quantità	Intero (richiesto: si, indice: si)	Identifica la quantità di lotto schedulata nel giorno
dataSchedulazione	Data (richiesto: si, indice: no)	Indica la data di schedulazione

Entità componente:

CAMPO	TIPO	DESCRIZIONE
idComponente	Contatore	Identifica il componente
nome	Testo(50) (richiesto: si, indice: no)	Indica il nome del componente
descrizione	Testo(50) (richiesto: si, indice: no)	Descrizione verbale delle caratteristiche del componente
idFornitore	Intero lungo	Identifica il fornitore che fornisce il componente

Entità fornitore:

CAMPO	TIPO	DESCRIZIONE
idFornitore	Contatore	Identifica il fornitore
nome	Testo(50) (richiesto: si, indice: no)	Indica il nome del fornitore
cognome	Testo(50) (richiesto: si, indice: no)	Indica il cognome del fornitore
indirizzo	Testo(50) (richiesto: si, indice: no)	Indica l'indirizzo del fornitore
telefono	Testo(50) (richiesto: si, indice: no)	Indica il numero di telefono del fornitore
email	Testo(50) (richiesto: si, indice: no)	Indica l'indirizzo e-mail del fornitore
partita_iva	Testo(50) (richiesto: si, indice: no)	Indica la partita iva del fornitore
compagnia	Testo(50) (richiesto: si, indice: si)	Indica il nome della compagnia del fornitore

Entità ordineComponenti:

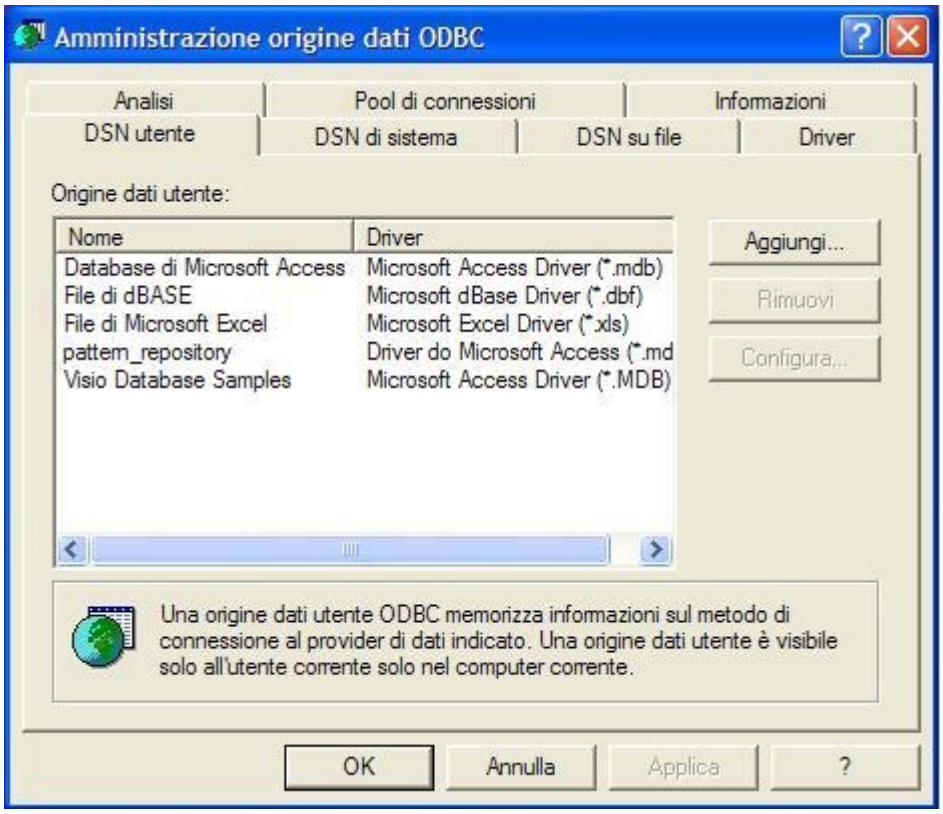
CAMPO	TIPO	DESCRIZIONE
idOrdine	Contatore	Identifica l'ordine
dataEmissione	Data (richiesto: si, indice: no)	Indica la data di emissione dell'ordine
dataConsegna	Data (richiesto: si, indice: no)	Indica la data di consegna dell'ordine
idStabilimento	Intero lungo	Identifica lo stabilimento che ha emesso l'ordine

APPENDICE

Procedura di installazione

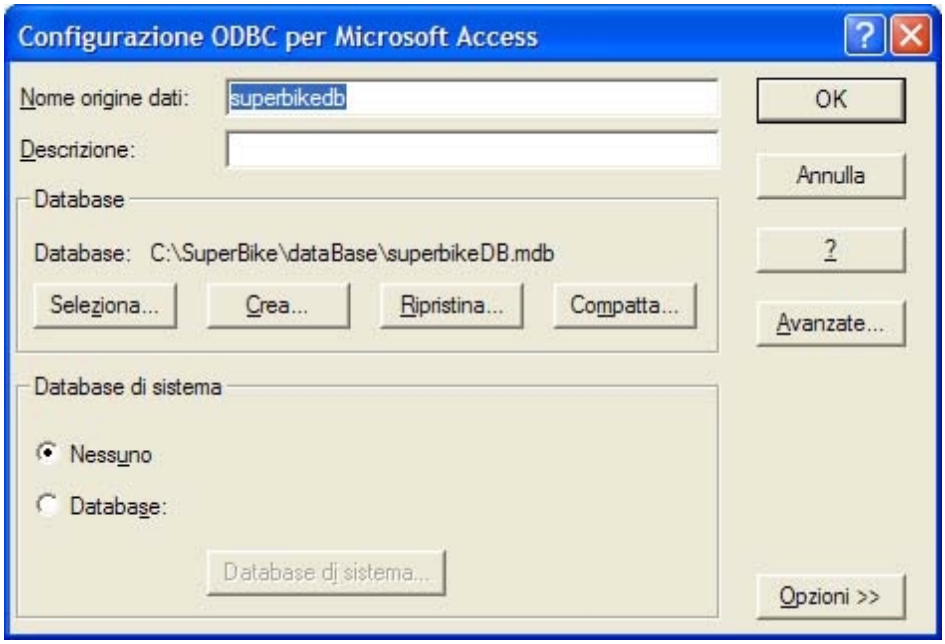
Per l'installazione del software eseguire le seguenti operazioni:

1. copiare la cartella `SuperBike` dal CD su ogni terminale di destinazione; essa contiene i files sorgenti e compilati del software e il file `superbikeDB.mdb` che contiene il Database;
2. impostare nel Server il Database `superbikeDB.mdb` come origine dei dati ODBC eseguendo queste operazioni (su Windows Xp):
   - accedere al Pannello di Controllo, scegliere la voce "Strumenti di amministrazione" e qui "Origine dati (ODBC)";
   - si aprirà la seguente finestra "Amministrazione origine dati ODBC" nella quale selezionare l'etichetta **DSN utente**, quindi fare click su **Aggiungi**;



   - appare la finestra "Crea nuova origine dati" nella quale selezionarele voce **Microsoft Access Driver (*.mdb)** e poi fare click su **Fine**;

- a questo punto appare la finestra “Configurazione ODBC per Microsoft Access”; nel campo nome origine dati scrivere “`superbikedb`” poi cliccare su **Seleziona** e immettere il percorso completo della cartella in cui è contenuto il file `superbikeDB.mdb` come mostrato in figura, qiundi cliccare su **OK**;



3. modificare il file `run.bat` contenuto nella cartella `<path>/Superbike/bin`:
   - modificare la seguente riga:

     ```
     set JADE_HOME=<percorso_jade>
     ```

     dove `<percorso_jade>` è la cartella in cui è installato jade
   - modificare la seguente riga:

     ```
     set SERVER="nome_server"
     ```

     dove `"nome_server"` è il nome del server.

4. avviare il sistema nella seguente modalità dopo essere entrati nella cartella `<path>/Superbike/bin`:
   - sul Server lanciare il file `system.bat`;
   - dal terminale dell'addetto ufficio clienti lanciare il file `agOrdini.bat`;
   - dal terminale dell'addetto produzione lanciare il file `agLotti_addetto.bat`;
   - dal terminale del responsabile settore produzione lanciare il file `agLotti_responsabile.bat`;
   - dal terminale del responsabile dello stabilimento Nord lanciare il file `agSchedulazione_a.bat`;
   - dal terminale del responsabile dello stabilimento Sud lanciare il file `agSchedulazione_b.bat`;
   - dal terminale del responsabile magazzino dello stabilimento Nord lanciare il file `agConsumi_a.bat`;
   - dal terminale del responsabile magazzino dello stabilimento Sud lanciare il file `agConsumi_b.bat`;
   - dal terminale dell'operaio addetto alla produzione dello stabilimento Nord lanciare il file `agEtichette_a.bat`;
   - dal terminale dell'operaio addetto alla produzione dello stabilimento Sud lanciare il file `agEtichette_b.bat`.