

# Indice

<b>TESINA.....</b>	<b>4</b>
TESTO DELLA TESINA.....	4
INTRODUZIONE.....	4
STRUMENTI PER LO SVILUPPO.....	5
<b>STUDIO DI FATTIBILITÀ.....</b>	<b>6</b>
MOTIVAZIONI DEL PROGETTO.....	6
LIMITI DEL PROGETTO.....	6
OBIETTIVI DEL SISTEMA.....	6
ANALISI DEI SISTEMI ESISTENTI.....	7
CONSIDERAZIONI LEGALI.....	7
<i>Software</i> .....	7
<i>Hardware</i> .....	7
<b>ANALISI DEI REQUISITI.....</b>	<b>8</b>
<b>SPECIFICHE DI SISTEMA.....</b>	<b>12</b>
PIANIFICAZIONE TEMPORALE.....	12
<i>Tabella delle attività</i> .....	12
<i>Tabella risorse</i> .....	13
<i>Tabella “Chi fa che cosa”</i> .....	13
<i>Costi delle attività</i> .....	15
<i>Diagramma di GANTT</i> .....	16
<i>Carta di PERT</i> .....	17
DESCRIZIONE FUNZIONALE.....	18
<i>Lettore codice a barre</i> .....	18
<i>Process Specification (PSPEC)</i> .....	36
<i>Strutture dati</i> .....	43
DESCRIZIONE ARCHITETTURALE.....	45
<i>L’hardware</i> .....	46
GLOSSARIO.....	58

<b>PROGETTAZIONE .....</b>	<b>59</b>
STRATEGIA ADOTTATA.....	59
I MODULI DI BASE .....	59
I MODULI DA REALIZZARE .....	62
<i>Lettura bolletta</i> .....	62
<i>Contatore denaro</i> .....	63
<i>Gestione resto</i> .....	66
<i>Schedulatore</i> .....	67
<i>Statistica</i> .....	68
<i>Gestione messaggi</i> .....	69
<i>Varie</i> .....	70
GRAFICI ALLEGATI ALLA DESCRIZIONE .....	71
<i>Grafico relazione USA:</i> .....	71
<i>Class diagram: Gestione bolletta</i> .....	72
<i>Class diagram: gestione denaro</i> .....	73
<b>APPENDICE A: DATABASE STATISTICA .....</b>	<b>74</b>
<b>APPENDICE B: INTERFACCIA UTENTE .....</b>	<b>75</b>
<b>APPENDICE C: DESIGN E FUNZIONALITÀ.....</b>	<b>76</b>
<b>RIFERIMENTI.....</b>	<b>78</b>
BIBLIOGRAFIA .....	78
SITI UTILI .....	78

# Tesina

## ***Testo della tesina***

Corsi di Laurea in Ing. Informatica ed Ing. Elettronica  
 Corso di Ingegneria del Software  
 Prof. Umberto Lo Faso, a. a. 1998/99

Si deve progettare il software di gestione di una macchina per lo sportello di pagamento automatico (cioè senza operatore, di tipo Bancomat) delle fatture di utenza come ad es. telefono, acqua, gas, elettricità o altro.

Tre esemplari osservabili di macchine di questo genere si trovano presso la Telecom, a Palermo, in via Pacinotti, ma funzionano solo per fatture Telecom.

La macchina deve essere in grado di:

- 1) istruire il cliente sul suo stesso uso e guidarlo nelle operazioni,
- 2) essere in grado di raccogliere i dati della fattura (la soluzione più opportuna da scegliere è a cura del progettista)
- 3) accettare la moneta e dare il resto
- 4) fornire la ricevuta del pagamento effettuato

La qualità del sistema consiste nella robustezza ed affidabilità dell'interazione con i clienti, soprattutto nei momenti di particolare affollamento.

Il sistema deve essere in grado di conservare i dati e di calcolare le seguenti statistiche su base giornaliera, mensile, annuale:

- a) numero di clienti in totale presentatisi allo sportello automatico,
- b) estremi dell'operazione (nome, data, ora, cifra, ...),
- c) informazioni sui casi peggiori.

## ***Introduzione***

Il lavoro richiesto per la tesina di Ingegneria del Software è quello di analisi, pianificazione e progettazione del sistema descritto nel testo. Nella pianificazione dei tempi, delle attività e dei costi si deve simulare una situazione aziendale; ovvero si deve supporre di essere parte integrante di uno staff di sviluppatori (analista, progettista, programmatori, esperti di vari settori...) in modo da poter distinguere le attività e stabilire i costi e le durate. In questa tesina, ho voluto cambiare di poco questa situazione, rimanendo in un ambito a me più familiare. Ho ipotizzato, infatti, che a commissionarmi il lavoro siano stati tre studenti di Linguaggi e Traduttori<sup>1</sup>, i quali hanno ricevuto la tesina dal loro docente e, poiché sentivano la necessità di procedere in maniera sicura e moderna, mi hanno chiesto di svolgere per loro questo lavoro. L'attività di codifica, testing e convalida è quindi affidata a questo gruppo. La mia è quindi una tesina nella tesina.

---

<sup>1</sup> La scelta della materia Linguaggi e Traduttori non è stata proprio casuale. Si tratta infatti della prima materia del corso in cui si deve svolgere un lavoro piuttosto complesso di programmazione. Le tesine che sono realizzate sono spesso il frutto di mesi e mesi di lavoro di gruppo, in cui si incontrano non poche difficoltà di gestione e coordinazione tra più persone. In particolare io stesso sono stato occupato in una tesina del genere poco tempo fa, e molte delle difficoltà che abbiamo incontrato, adesso avremmo potuto evitarle mediante un lavoro di pianificazione, analisi e progettazione.

## ***Strumenti per lo sviluppo***

Per la realizzazione di questo documento sono stati usati un Pentium II 233Mhz, 32Mb RAM e 3.2Gb di HD, Modem 56K, Scanner e Stampante HP Laserjet.

Gli strumenti software usati, in ambiente Windows 98, sono:

- 1) *Turbo Case 2.02* per i grafici DFD, STD, decision table, EDFD, AFD, ACD, AICD, AID e il Dizionario Dati.
- 2) *Microsoft Word 2000* per la relazione, la progettazione in TDN e le specifiche PSPEC
- 3) *Microsoft Excel 2000* per il calcolo dei costi e le tabelle
- 4) *Microsoft Project* per la pianificazione delle attività, delle risorse, e le tavole di GANTT e di PERT
- 5) *Microsoft Draw* per la grafica di copertina
- 6) *Visual UML* per i grafici di classe
- 7) *Adobe Acrobat* per la stampa finale del documento.

## Studio di fattibilità

### ***Motivazioni del progetto***

Questo progetto ha come motivazione essenziale, quella di permettere il pagamento delle fatture di utenza anche al di fuori degli orari di ufficio. Questo porta alcune utili conseguenze, ad esempio la riduzione del carico di lavoro degli impiegati agli sportelli, il che equivale a meno code, e meno stress:

- 1) pagamento in qualunque orario
- 2) alleggerimento carichi lavoro
- 3) eliminazione lavoro ripetitivo
- 4) snellimento personale addetto

### ***Limiti del progetto***

Il progetto è sottoposto a due limiti temporali, cioè la scadenza della fase di analisi e di progettazione e la fine dell'intero sistema. La prima data è fissata a causa dell'esame (sessione di Settembre) di Ingegneria del Software, entro il quale la prima parte della tesina deve essere svolta. La seconda data è la data di esame (sessione di Ottobre/Novembre) di Linguaggi e Traduttori, per la quale deve essere presentato la versione funzionante<sup>2</sup>.

A partecipare al progetto sono:

- 1) analista/progettista
- 2) programmatore C++
- 3) programmatore C/C++, esperto in database
- 4) programmatore C++, disegnatore

La figura del cliente, in questo caso è il professore, il quale ha commissionato il lavoro sotto forma di tesina.

### ***Obiettivi del sistema***

La macchina che deve essere realizzata deve avere una notevole autonomia, anche in casi speciali (assenza di corrente elettrica) e deve essere in grado di facilitare il proprio uso. Inoltre deve rendere semplice ed immediato il pagamento e la ricezione del resto.

- 1) autonomia
- 2) auto-diagnosi
- 3) user-friendly
- 4) transazioni sicure
- 5) facilità nel pagamento
- 6) sveltezza dell'operazione

Il codice che sarà realizzato inoltre deve rispondere ai seguenti fattori di qualità:

- 1) modificabilità
- 2) riusabilità

---

<sup>2</sup> Per versione funzionante in realtà si intende una versione simulata della macchina, ovvero in cui le interfacce hardware vengono simulate sullo schermo del PC.

- 3) affidabilità
- 4) usabilità

Per poter imporre la macchina sul mercato (nazionale ed estero), devono essere perseguiti i seguenti obiettivi di mercato:

- 1) la macchina deve essere facilmente adattabile ad altre lingue
- 2) i tipi di moneta e banconote accettate devono poter essere cambiati facilmente (senza intervenire sul codice)
- 3) deve essere adattabile al maggior numero possibile di bollette di utenza già esistenti

### ***Analisi dei sistemi esistenti***

Al momento della realizzazione di tale sistema, a Palermo, l'unico esemplare di macchina simile già esistente si trova in via Pacinotti. La macchina è prodotta dalla SIGMA. Ad un'analisi si sono trovati i seguenti punti deboli:

- 1) alta riflessione dello schermo
- 2) non emette ricevuta (effettua solo la bollatura della fattura)
- 3) non accetta monete
- 4) schermo troppo alto per persone disabili

### ***Considerazioni legali***

Una prospettiva legale dell'analisi del sistema è quella di permettere a chiunque l'utilizzo. Per questo motivo sono state previste alcune funzionalità.

### ***Software***

- 1) è prevista l'attivazione/disattivazione del sonoro, per coloro che non riescono a distinguere i caratteri sullo schermo. I messaggi sonori sono descritti in maniera concisa le operazioni da effettuare.
- 2) accanto al testo devono apparire delle immagini (anche animate) di grandi dimensioni che migliorano la comprensione del messaggio.

### ***Hardware***

- 1) lo schermo sarà disposto ad altezza di carrozzina, ma inclinato in maniera da permettere la visione a chiunque.
- 2) lo schermo sarà coperto da una superficie metallica per parare il riflesso
- 3) lo schermo sarà illuminato (solo quando in funzione)

## Analisi dei requisiti

I requisiti raccolti per la macchina pagamento bollette alla prima riunione tra analista e programmatore leader, sono stati divisi in varie categorie:

- 1) requisito **funzionale**, ovvero che specifica caratteristiche funzionali che il sistema deve avere.
- 2) **legale**, cioè che specifica caratteristiche che il sistema deve avere per non andare contro la legge.
- 3) di **progettazione**, cioè relativo all'architettura logica o ad altre caratteristiche tecniche. Rientrano in questa categoria requisiti di tracciabilità, modularietà, espandibilità, portabilità, interoperabilità, riusabilità.
- 4) di **utilizzo**, cioè relativo alle modalità di utilizzo del sistema da parte degli utenti. Rientrano in questa categoria i requisiti di usabilità, documentazione, supporto, formazione.
- 5) **servizio**, cioè relativo alla capacità del sistema di fornire un servizio efficiente e continuativo. Rientrano in questa categoria i requisiti di disponibilità, efficienza, carichi elaborativi.

N requisito	1
Tipologia	Funzionale
Descrizione	<b>La macchina deve raccogliere da sola i dati della bolletta</b>
Importanza	1
Motivazione importanza	Questo requisito è l'essenza dell'automazione fornita dalla macchina. Se dovesse essere necessario un operatore che introduca il valore della bolletta, l'utilità cessa completamente
Legame con altri requisiti	

N requisito	2
Tipologia	Funzionale
Descrizione	<b>Deve mantenere delle statistiche sulle operazioni compiute</b>
Importanza	2
Motivazione importanza	I dati forniti dalla macchina, che devono rispettare la privacy, sono utili alla società che usa la macchina
Legame con altri requisiti	

N requisito	3
Tipologia	Funzionale
Descrizione	<b>Le operazioni devono essere temporizzate e deve essere previsto un tempo limite a persona.</b>
Importanza	2
Motivazione importanza	E' molto importante per sveltire l'operazione e per l'acquisizione delle statistiche
Legame con altri requisiti	statistiche

N requisito	4
Tipologia	Legale
Descrizione	<b>La macchina deve accettare sia banconote sia monete e deve dare il resto. Qualora questo non sia disponibile deve avvertire l'utente e chiedere conferma.</b>
Importanza	1
Legame con altri requisiti	

N requisito	5
Tipologia	Legale
Descrizione	<b>Deve fornire la ricevuta del pagamento, nella quale è riportato l'eventuale resto che non è stato possibile fornire</b>
Importanza	1
Legame con altri requisiti	

N requisito	6
Tipologia	Progettazione
Descrizione	<b>Deve essere possibile cambiare o aggiungere dei tipi di valuta</b>
Importanza	2
Motivazione importanza	E' importante che il sistema possa essere adattato ai cambiamenti sociali o per l'uso in diversi paesi
Legame con altri requisiti	cambio lingua

N requisito	7
Tipologia	Progettazione
Descrizione	<b>Si può cambiare la lingua d'utilizzo</b>
Importanza	3
Motivazione importanza	E' importante che il sistema possa essere adattato ai cambiamenti sociali o per l'uso in diversi paesi
Legame con altri requisiti	cambio valuta

N requisito	8
Tipologia	Progettazione
Descrizione	<b>Si richiede la massima generalità delle bollette da pagare</b>
Importanza	4
Motivazione importanza	Questo requisito rende la macchina generica, così da poter essere usata da più enti, per il pagamento delle bollette, cambiando solo la specifica della bolletta
Legame con altri requisiti	

N requisito	9
Tipologia	Progettazione
Descrizione	<b>Il sistema deve essere predisposto per la connessione remota con un sistema portatile per l'aggiornamento dei dati</b>
Importanza	5
Motivazione importanza	L'aggiunta di funzionalità future può portare alla connessione tra la macchina e altri computer remoti, o alla rete telematica
Legame con altri requisiti	

N requisito	10
Tipologia	Progettazione
Descrizione	<b>Il sistema deve prevedere informazioni di tipo testuale.</b>
Legame con altri requisiti	

N requisito	11
Tipologia	Progettazione
Descrizione	<b>Il sistema si può avvalere della visualizzazione di immagini e animazioni. Deve essere sempre visualizzato il tempo residuo e devono essere gestiti segnali sonori. Quest'ultima possibilità può essere disattivata.</b>
Importanza	Le immagini possono aiutare a comprendere quale operazione è richiesta, i suoni sono importanti per aiutare persone anziane o disabili.
Motivazione importanza	3
Legame con altri requisiti	Messaggi testuali

N requisito	12
Tipologia	Utilizzo
Descrizione	<b>I messaggi e le immagini dovranno essere i più chiari e comprensibili a tutti</b>
Importanza	1
Motivazione importanza	Tutti devono comprendere quale operazione deve essere svolta senza introdurre frustrazione.
Legame con altri requisiti	Messaggi testuali e immagini.

N requisito	13
Tipologia	Utilizzo
Descrizione	<b>La macchina deve istruire automaticamente l'utente sul suo uso</b>
Importanza	1
Motivazione importanza	La macchina deve invogliare l'uso della medesima, rispetto a soluzioni classiche come la coda allo sportello
Legame con altri requisiti	Funzionamento 24 ore su 24, pronta risposta ai comandi

N requisito	14
Tipologia	Utilizzo
Descrizione	<b>Il sistema deve essere abbastanza robusto da rimanere attivo 24 ore su 24</b>
Importanza	1
Motivazione importanza	Deve essere così perché la macchina possa avere successo
Legame con altri requisiti	istruire automaticamente

N requisito	15
Tipologia	Utilizzo
Descrizione	<b>Il sistema deve avere una pronta risposta ai comandi dell'utente in modo da consentire un veloce pagamento della bolletta</b>
Importanza	2
Motivazione importanza	Per accelerare l'operazione
Legame con altri requisiti	istruire automaticamente

N requisito	16
Tipologia	Utilizzo
Descrizione	<b>Deve essere possibile gestire automaticamente un turno. Questa funzionalità non deve limitare l'utilizzo, ma soltanto consentire alle persone in coda di non fare la fila attorno alla macchina.</b>
Importanza	5
Motivazione importanza	Per evitare confusione davanti la macchina, dovrebbe essere possibile prenotarsi e fare il turno con comodo.
Legame con altri requisiti	

N requisito	17
Tipologia	Servizio
Descrizione	<b>Deve essere garantita la correttezza e la sicurezza della transizione</b>
Importanza	1
Motivazione importanza	Non è ammissibile che la macchina sbagli, in quanto ciò corrisponderebbe a perdita di denaro o dell'utente o della società
Legame con altri requisiti	funzionamento 24 ore su 24

## Specifiche di sistema

### ***Pianificazione temporale***

Per la pianificazione temporale delle attività riportiamo alcuni grafici ottenuti con il programma MS Project, ed eventualmente rielaborati con l'Excel o il Word.

### ***Tabella delle attività***

<b>Task_Name</b>	<b>Duration</b>	<b>Start Date</b>	<b>Finish Date</b>	<b>Resource_Names</b>
Operazioni preliminari	5 days	02/08/99 8.00	06/08/99 17.00	
Uscita della tesina: discussione preliminare con il professore	1 day	02/08/99 8.00	02/08/99 17.00	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++
Riunione gruppo di lavoro: definizione responsabilità	1 day	03/08/99 8.00	03/08/99 17.00	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++
Scelta dell'obiettivo	2 days	04/08/99 8.00	05/08/99 17.00	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++
Incontro con l'analista	1 day	06/08/99 8.00	06/08/99 17.00	Programmatore C++ Leader;Analista
Analisi	14 days	09/08/99 8.00	26/08/99 17.00	
Analisi dei requisiti	2 days	09/08/99 8.00	10/08/99 17.00	Analista
Analisi costi/benefici	2 days	11/08/99 8.00	12/08/99 17.00	Analista
Analisi hardware/software	1 day	13/08/99 8.00	13/08/99 17.00	Analista;Ingegnere elettronico
Analisi del sistema	7 days	16/08/99 8.00	24/08/99 17.00	Analista
Validazione lavoro analisi	2 days	25/08/99 8.00	26/08/99 17.00	Analista;Programmatore C++ Leader
Fase di progetto	8,06 days	27/08/99 8.00	08/09/99 8.30	
Progettazione	7 days	27/08/99 8.00	07/09/99 8.30	Analista
Validazione del progetto	1 day	07/09/99 8.30	08/09/99 8.30	Analista;Programmatore C++ Leader
Definizione Interfaccia	18 days	09/08/99 8.00	01/09/99 17.00	
Scelta messaggi	15 days	09/08/99 8.00	01/09/99 17.00	Programmatore C++ Leader
Scelta immagini	15 days	09/08/99 8.00	27/08/99 17.00	Programmatore C++
Scelta file sonori	15 days	09/08/99 8.00	27/08/99 17.00	Programmatore C++ / Database
Programmazione	16 days	08/09/99 8.30	30/09/99 8.30	
Divisione dei compiti	1 day	08/09/99 8.30	09/09/99 8.30	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++
Codifica modulo gestione denaro	7 days	09/09/99 8.30	20/09/99 8.30	Programmatore C++ Leader
Codifica modulo gestione resto	7 days	20/09/99 8.30	29/09/99 8.30	Programmatore C++ Leader
Codifica moduli di base	4 days	09/09/99 8.30	15/09/99 8.30	Programmatore C++
Codifica modulo schedatore attività	4 days	15/09/99 8.30	21/09/99 8.30	Programmatore C++
Codifica modulo interfaccia	4 days	21/09/99 8.30	27/09/99 8.30	Programmatore C++
Codifica modulo statistica	2,5 wks	09/09/99 8.30	27/09/99 13.30	Programmatore C++ / Database
Incontro di verifica	1 day	29/09/99 8.30	30/09/99 8.30	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++
Verifica e test	5 days	30/09/99 8.30	07/10/99 8.30	
Eliminazione malfunzionamenti	4 days	30/09/99 8.30	06/10/99 8.30	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++
Unione moduli e test finale	1 day	06/10/99 8.30	07/10/99 8.30	Programmatore C++ Leader

Documentazione	9 days	07/10/99 8.30	20/10/99 8.30	
Scrittura documentazione: manuale utente	9 days	07/10/99 8.30	20/10/99 8.30	Programmatore C++ Leader
Scrittura documentazione: manuale tecnico	5 days	07/10/99 8.30	14/10/99 8.30	Programmatore C++ / Database
Scrittura messaggi	5 days	07/10/99 8.30	14/10/99 8.30	Programmatore C++
Fine del lavoro	2 days	20/10/99 8.30	22/10/99 8.30	
Assemblaggio finale	1 day	20/10/99 8.30	21/10/99 8.30	Programmatore C++ Leader
Installazione computer universitario	1 day	21/10/99 8.30	22/10/99 8.30	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++
Esame	1 day	22/10/99 8.30	25/10/99 8.30	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++

### ***Tabella risorse***

ID	Resource_Name	Initials	Standard Rate	Overtime Rate	Cost Per Use	Cost
1	Analista	A	L. 50.000/hr	L. 80.000/hr	L. 0	L. 9.200.000
2	Programmatore C++ Leader	P1	L. 30.000/hr	L. 35.000/hr	L. 0	L. 13.440.000
3	Programmatore C++ / Database	P2	L. 25.000/hr	L. 30.000/hr	L. 0	L. 8.900.000
4	Programmatore C++	P3	L. 25.000/hr	L. 30.000/hr	L. 0	L. 8.800.000
5	Ingegnere elettronico	I	L. 45.000/hr	L. 0/hr	L. 0	L. 45.000

ID	Resource_Name	Scheduled_Work	Baseline_Work	Actual_Work	Overtime_Work
1	Analista	184 hrs	192 hrs	0 hrs	0 hrs
2	Programmatore C++ Leader	448 hrs	272 hrs	0 hrs	0 hrs
3	Programmatore C++ / Database	356 hrs	224 hrs	0 hrs	0 hrs
4	Programmatore C++	352 hrs	200 hrs	0 hrs	0 hrs
5	Ingegnere elettronico	8 hrs	0 hrs	0 hrs	0 hrs

### ***Tabella "Chi fa che cosa"***

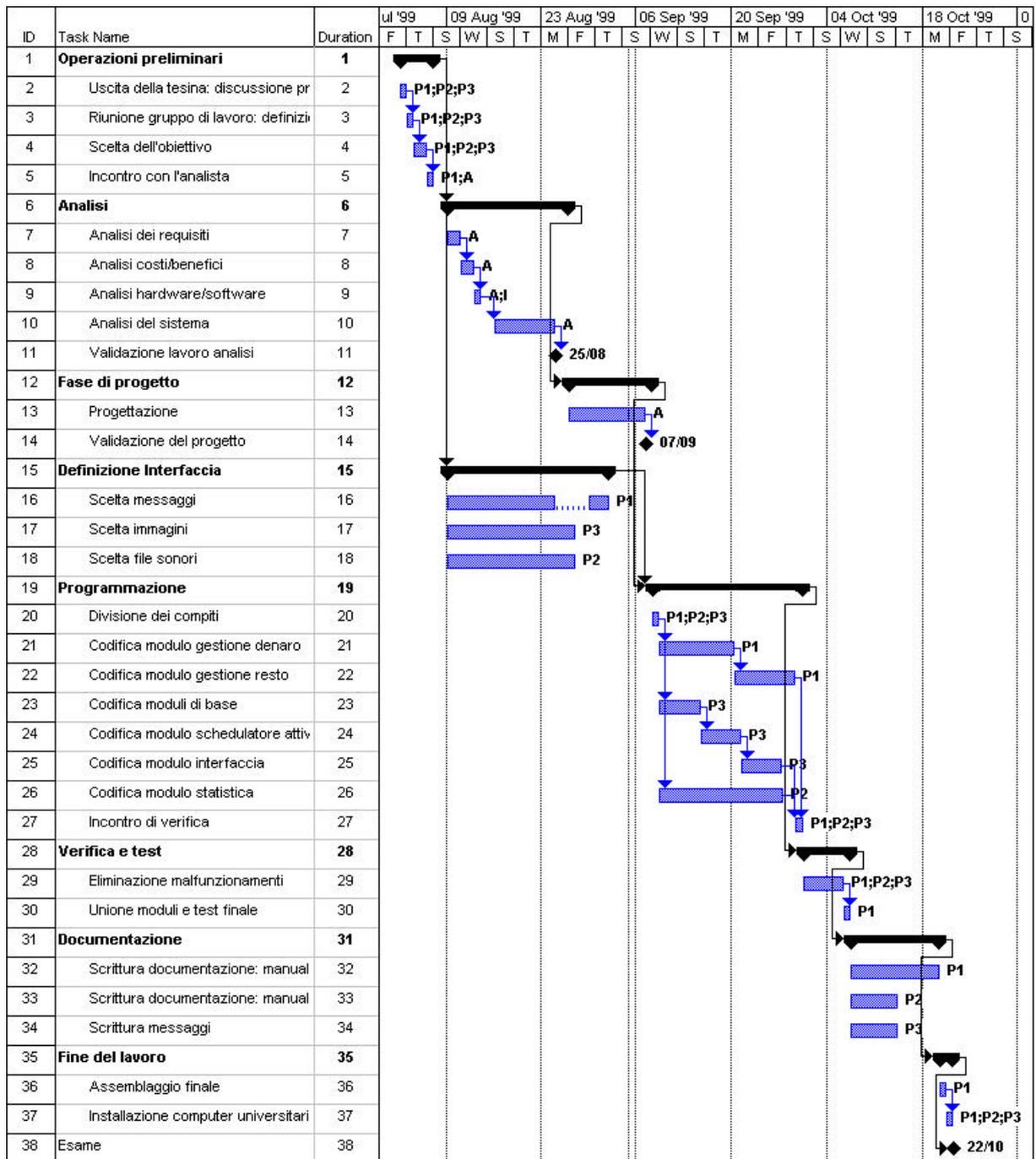
Resources and Assignments	Start	Finish	Work
Analista	06/08/99 8.00	08/09/99 8.30	184 hrs
Incontro con l'analista	06/08/99 8.00	06/08/99 17.00	8 hrs
Analisi dei requisiti	09/08/99 8.00	10/08/99 17.00	16 hrs
Analisi costi/benefici	11/08/99 8.00	12/08/99 17.00	16 hrs
Analisi hardware/software	13/08/99 8.00	13/08/99 17.00	8 hrs
Analisi del sistema	16/08/99 8.00	24/08/99 17.00	56 hrs
Validazione lavoro analisi	25/08/99 8.00	26/08/99 17.00	16 hrs
Progettazione	27/08/99 8.00	07/09/99 8.30	56 hrs
Validazione del progetto	07/09/99 8.30	08/09/99 8.30	8 hrs
Programmatore C++ Leader	02/08/99 8.00	25/10/99 8.30	448 hrs
Uscita della tesina: discussione preliminare con il professore	02/08/99 8.00	02/08/99 17.00	8 hrs
Riunione gruppo di lavoro: definizione responsabilità	03/08/99 8.00	03/08/99 17.00	8 hrs
Scelta dell'obiettivo	04/08/99 8.00	05/08/99 17.00	16 hrs
Incontro con l'analista	06/08/99 8.00	06/08/99 17.00	8 hrs
Validazione lavoro analisi	25/08/99 8.00	26/08/99 17.00	16 hrs
Validazione del progetto	07/09/99 8.30	08/09/99 8.30	8 hrs

Divisione dei compiti	08/09/99 8.30	09/09/99 8.30	8 hrs
Codifica modulo gestione denaro	09/09/99 8.30	20/09/99 8.30	56 hrs
Incontro di verifica	29/09/99 8.30	30/09/99 8.30	8 hrs
Eliminazione malfunzionamenti	30/09/99 8.30	06/10/99 8.30	32 hrs
Unione moduli e test finale	06/10/99 8.30	07/10/99 8.30	8 hrs
Scrittura documentazione: manuale utente	07/10/99 8.30	20/10/99 8.30	72 hrs
Assemblaggio finale	20/10/99 8.30	21/10/99 8.30	8 hrs
Installazione computer universitario	21/10/99 8.30	22/10/99 8.30	8 hrs
Esame	22/10/99 8.30	25/10/99 8.30	8 hrs
Scelta messaggi	09/08/99 8.00	01/09/99 17.00	120 hrs
Codifica modulo gestione resto	20/09/99 8.30	29/09/99 8.30	56 hrs
<b>Programmatore C++ / Database</b>	<b>02/08/99 8.00</b>	<b>25/10/99 8.30</b>	<b>356 hrs</b>
Uscita della tesina: discussione preliminare con il professore	02/08/99 8.00	02/08/99 17.00	8 hrs
Riunione gruppo di lavoro: definizione responsabilità	03/08/99 8.00	03/08/99 17.00	8 hrs
Scelta dell'obiettivo	04/08/99 8.00	05/08/99 17.00	16 hrs
Divisione dei compiti	08/09/99 8.30	09/09/99 8.30	8 hrs
Incontro di verifica	29/09/99 8.30	30/09/99 8.30	8 hrs
Eliminazione malfunzionamenti	30/09/99 8.30	06/10/99 8.30	32 hrs
Scrittura documentazione: manuale tecnico	07/10/99 8.30	14/10/99 8.30	40 hrs
Installazione computer universitario	21/10/99 8.30	22/10/99 8.30	8 hrs
Esame	22/10/99 8.30	25/10/99 8.30	8 hrs
Scelta file sonori	09/08/99 8.00	27/08/99 17.00	120 hrs
Codifica modulo statistica	09/09/99 8.30	27/09/99 13.30	100 hrs
<b>Programmatore C++</b>	<b>02/08/99 8.00</b>	<b>25/10/99 8.30</b>	<b>352 hrs</b>
Uscita della tesina: discussione preliminare con il professore	02/08/99 8.00	02/08/99 17.00	8 hrs
Riunione gruppo di lavoro: definizione responsabilità	03/08/99 8.00	03/08/99 17.00	8 hrs
Scelta dell'obiettivo	04/08/99 8.00	05/08/99 17.00	16 hrs
Divisione dei compiti	08/09/99 8.30	09/09/99 8.30	8 hrs
Codifica modulo schedatore attività	15/09/99 8.30	21/09/99 8.30	32 hrs
Codifica modulo interfaccia	21/09/99 8.30	27/09/99 8.30	32 hrs
Incontro di verifica	29/09/99 8.30	30/09/99 8.30	8 hrs
Eliminazione malfunzionamenti	30/09/99 8.30	06/10/99 8.30	32 hrs
Scrittura messaggi	07/10/99 8.30	14/10/99 8.30	40 hrs
Installazione computer universitario	21/10/99 8.30	22/10/99 8.30	8 hrs
Esame	22/10/99 8.30	25/10/99 8.30	8 hrs
Scelta immagini	09/08/99 8.00	27/08/99 17.00	120 hrs
Codifica moduli di base	09/09/99 8.30	15/09/99 8.30	32 hrs
<b>Ingegnere elettronico</b>	<b>13/08/99 8.00</b>	<b>13/08/99 17.00</b>	<b>8 hrs</b>
Analisi hardware/software	13/08/99 8.00	13/08/99 17.00	8 hrs

**Costi delle attività**

Attività	Durata	Risorse	Costo
Operazioni preliminari	5 days		L. 3.200.000
Uscita della tesina: discussione preliminare con il professore	1 day	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++	L. 640.000
Riunione gruppo di lavoro: definizione responsabilità	1 day	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++	L. 640.000
Scelta dell'obiettivo	2 days	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++	L. 1.280.000
Incontro con l'analista	1 day	Programmatore C++ Leader;Analista	L. 640.000
Analisi	14 days		L. 6.125.000
Analisi dei requisiti	2 days	Analista	L. 800.000
Analisi costi/benefici	2 days	Analista	L. 800.000
Analisi hardware/software	1 day	Analista;Ingegnere elettronico	L. 445.000
Analisi del sistema	7 days	Analista	L. 2.800.000
Validazione lavoro analisi	2 days	Analista;Programmatore C++ Leader	L. 1.280.000
Fase di progetto	8,06 days		L. 3.440.000
Progettazione	7 days	Analista	L. 2.800.000
Validazione del progetto	1 day		L. 640.000
Definizione Interfaccia	18 days	Analista;Programmatore C++ Leader	L. 9.600.000
Scelta messaggi	15 days	Programmatore C++ Leader	L. 3.600.000
Scelta immagini	15 days	Programmatore C++	L. 3.000.000
Scelta file sonori	15 days	Programmatore C++ / Database	L. 3.000.000
Programmazione	16 days		L. 9.540.000
Divisione dei compiti	1 day	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++	L. 640.000
Codifica modulo gestione denaro	7 days	Programmatore C++ Leader	L. 1.680.000
Codifica modulo gestione resto	7 days	Programmatore C++ Leader	L. 1.680.000
Codifica moduli di base	4 days	Programmatore C++	L. 800.000
Codifica modulo schedulatore attività	4 days	Programmatore C++	L. 800.000
Codifica modulo interfaccia	4 days	Programmatore C++	L. 800.000
Codifica modulo statistica	2,5 wks	Programmatore C++ / Database	L. 2.500.000
Incontro di verifica	1 day	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++	L. 640.000
Verifica e test	5 days		L. 2.800.000
Eliminazione malfunzionamenti	4 days	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++	L. 2.560.000
Unione moduli e test finale	1 day	Programmatore C++ Leader	L. 240.000
Documentazione	9 days		L. 4.160.000
Scrittura documentazione: manuale utente	9 days	Programmatore C++ Leader	L. 2.160.000
Scrittura documentazione: manuale tecnico	5 days	Programmatore C++ / Database	L. 1.000.000
Scrittura messaggi	5 days	Programmatore C++	L. 1.000.000
Fine del lavoro	2 days		L. 880.000
Assemblaggio finale	1 day	Programmatore C++ Leader	L. 240.000
Installazione computer universitario	1 day	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++	L. 640.000
Esame	1 day	Programmatore C++ Leader;Programmatore C++ / Database;Programmatore C++	L. 640.000
<b>Totale</b>			<b>L. 39.745.000</b>

## Diagramma di GANTT



## **Descrizione funzionale**

La strategia adottata, per la definizione dei diagrammi di flusso dei dati, è di tipo top-down. A partire dal context diagram, le funzioni sono state scomposte in sottobolle via via più semplici in maniera ricorsiva, fino ad arrivare ai processi primitivi.

Poiché la notazione DeMarco ha introdotto i flussi di controllo, e il case usato li mette a disposizione, la notazione è stata espansa di questo strumento espressivo.

Un altro simbolo aggiuntivo che è stato utilizzato è la control bar, che sintetizza la politica di controllo del processo presente nel DFD in cui si trova. Le control bar sono state epanse mediante diadrammi STD o tavole di decisione.

I moduli primitivi sono stati descritti mediante PSPEC in un linguaggio non formale di alto livello.

## **Lettore codice a barre**

Nella definizione del problema si è presentata una scelta di carattere tecnico. L'introduzione dei dati dalla fattura nel sistema avviene mediante il sistema di codice a barre, già in uso nelle fatture di utenza più importanti.

Per supportare tale scelta si può procedere in due modi:

- 1) utilizzare un lettore di codici a barre già pronto tra quelli esistenti sul mercato,
- 2) supportare il sistema di un lettore scanner generico e acquistare il modulo per la decodifica del codice a barre.

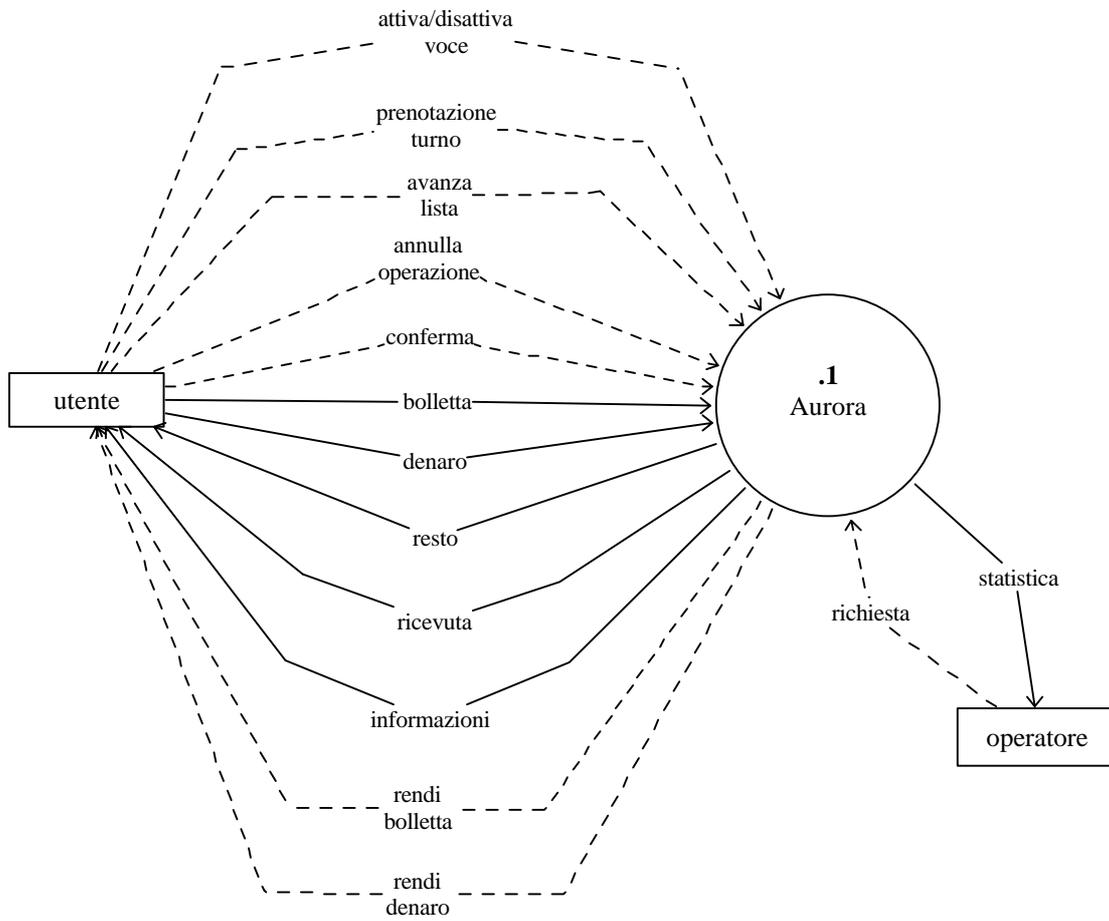
Nel codice a barre della fattura ci sono le informazioni riguardanti all'importo, ma nessuna che riguarda i dati utente. Poiché la rilevazione e la memorizzazione di tali dati è stato espressamente richiesto, si è pensato di utilizzare la seconda scelta, la quale permette di estrarre ulteriori dati anche in forma differente dal codice a barre.

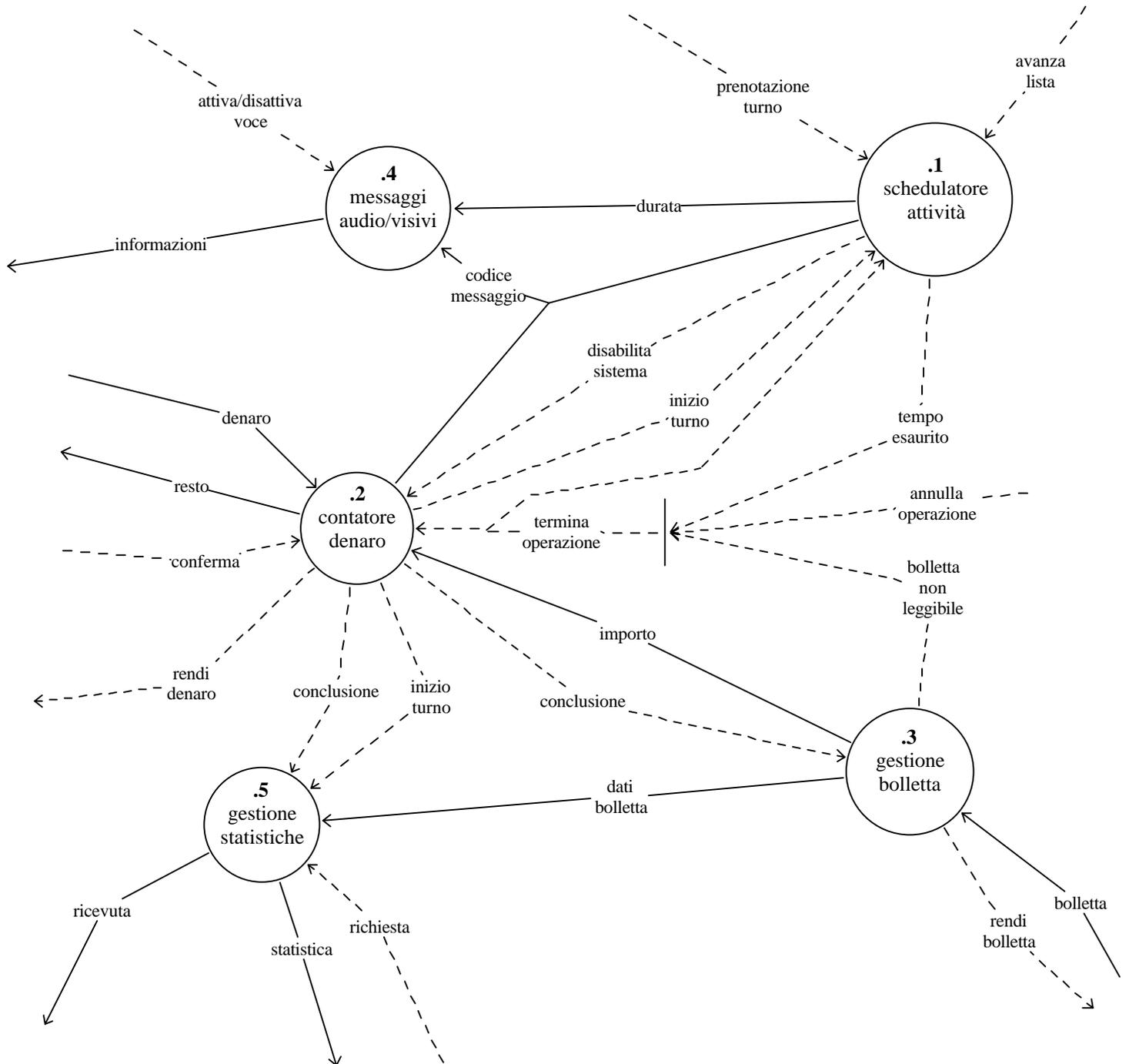
Tali moduli esistono sul mercato e svolgono la funzionalità di

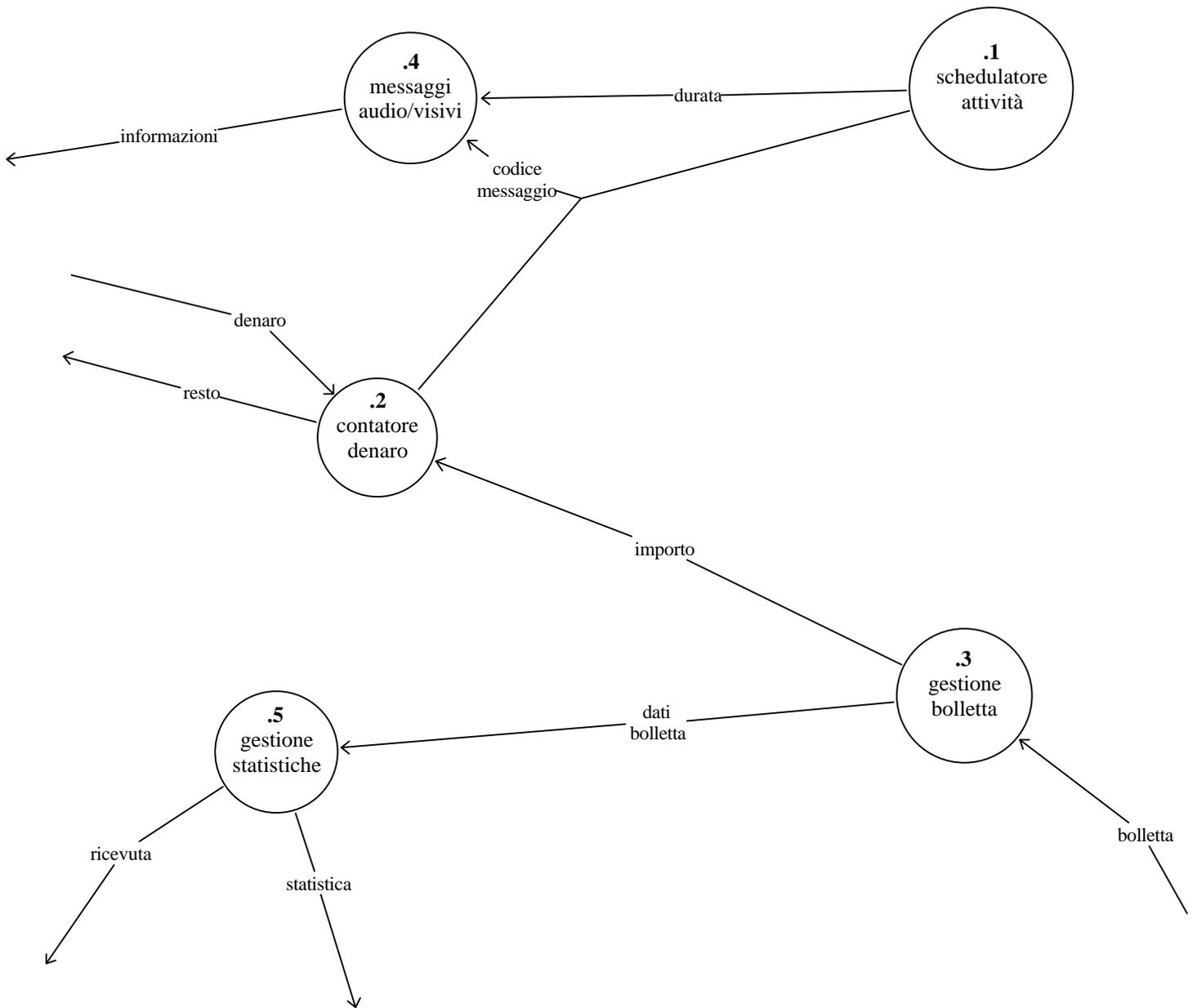
- 1) interprete codice a barre
- 2) OCR / ICR

Una delle società che fornisce tali prodotti, sotto forma di librerie è la DATASIS GROUP SRL (vedere sez. riferimenti) che fornisce:

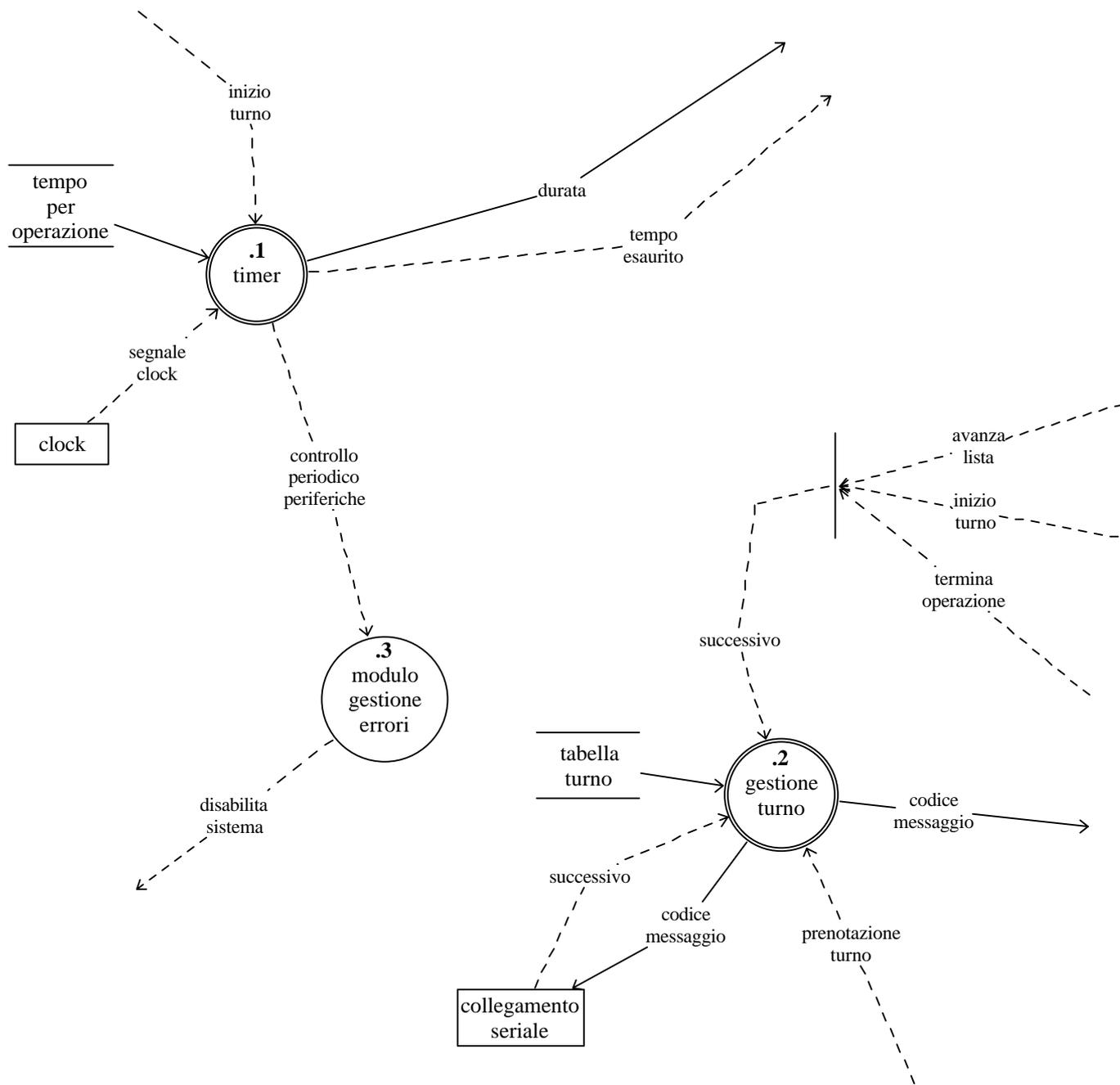
- 1) moduli per l'individuazione del barcode all'interno di una pagina,
- 2) moduli per l'interpretazione dei barcode anche con orientamento non esattamente a 180° o 90° ,
- 3) reti neurali per la lettura dei caratteri anche nelle fatture più rovinate.

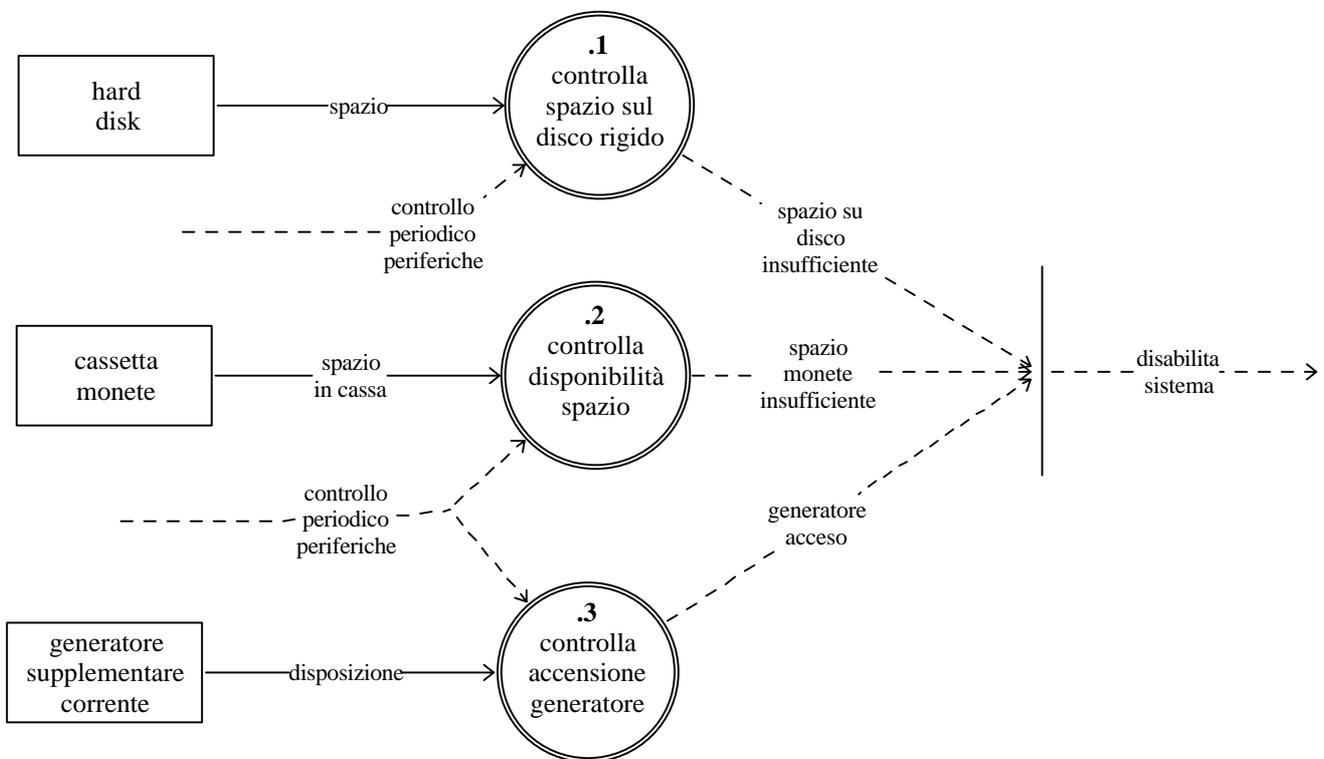


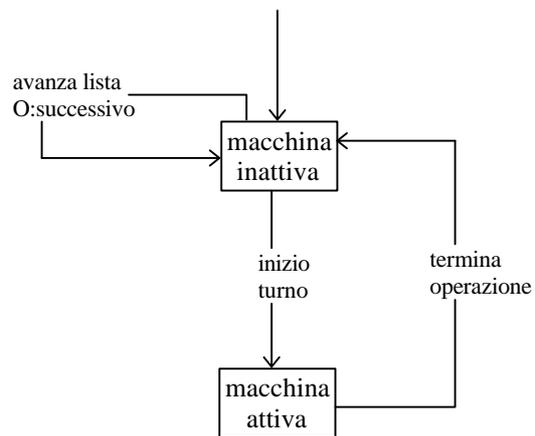


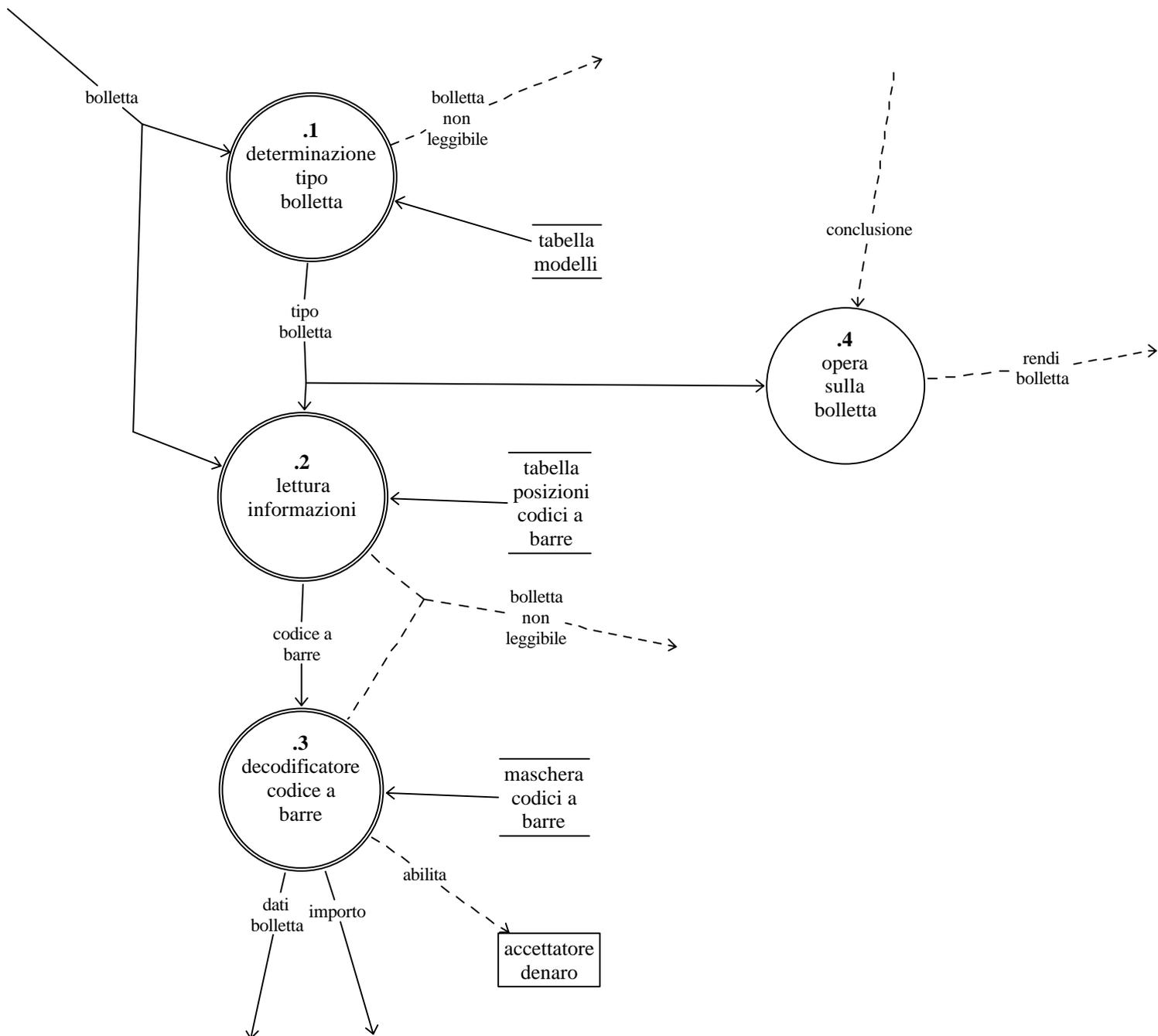


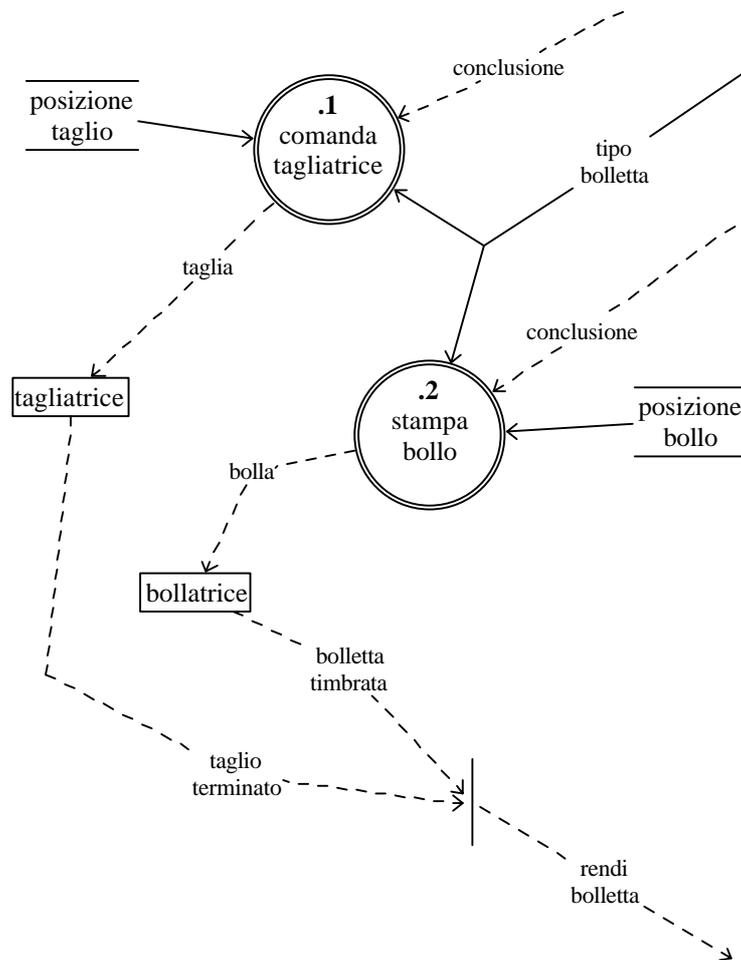


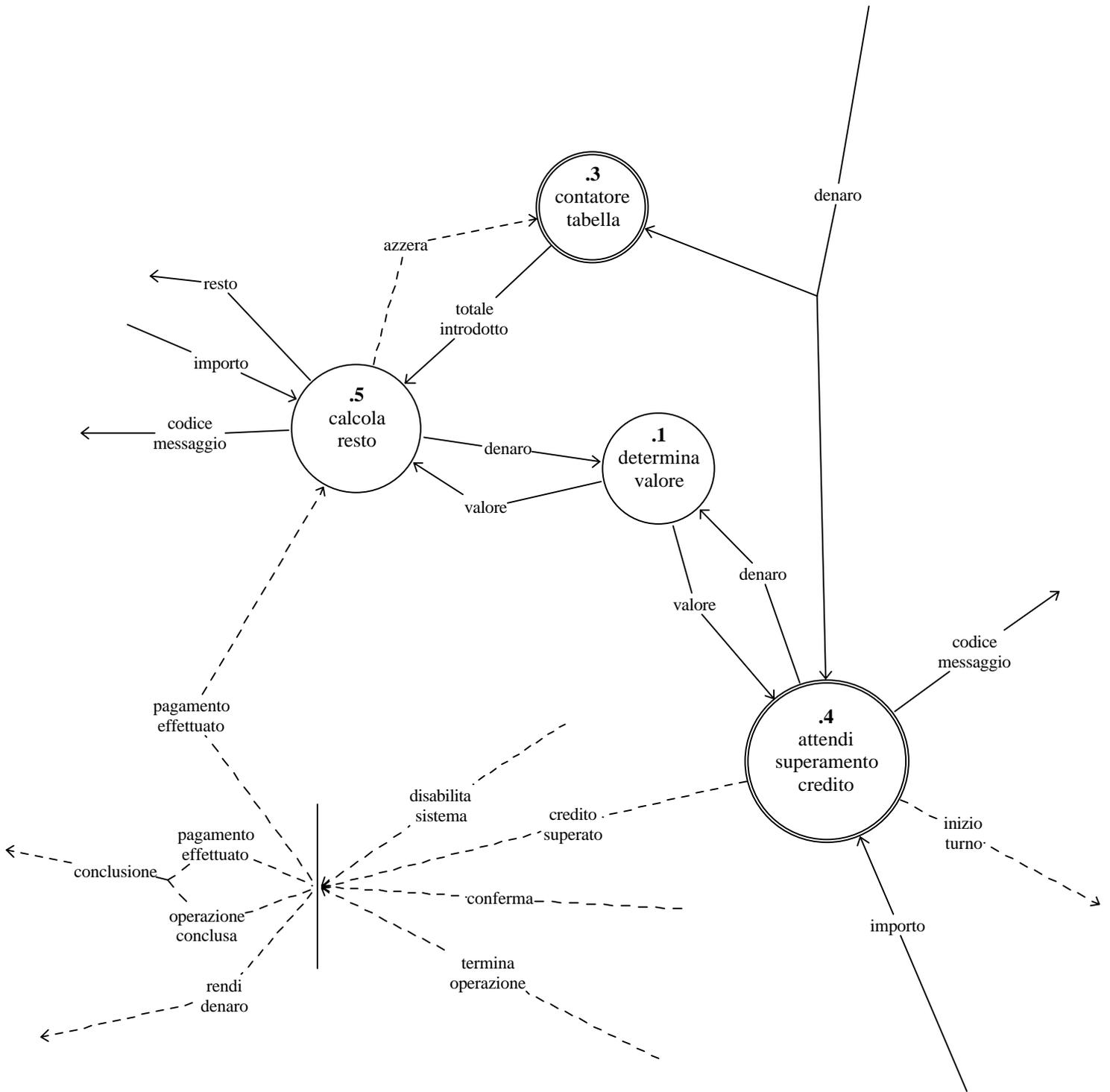


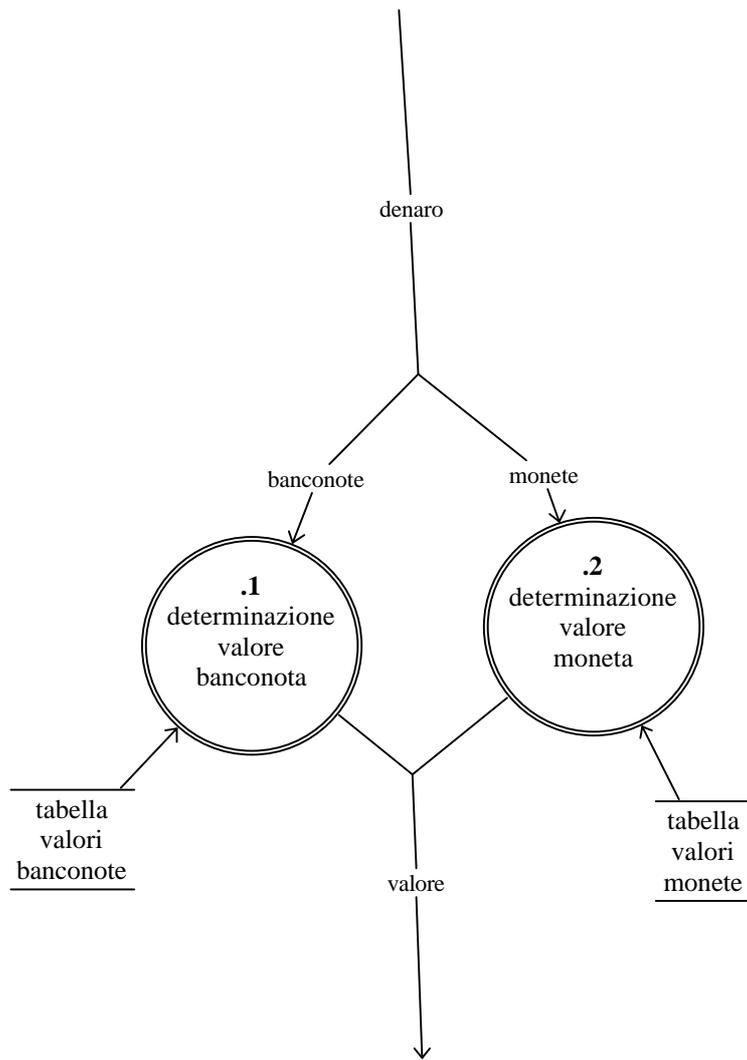




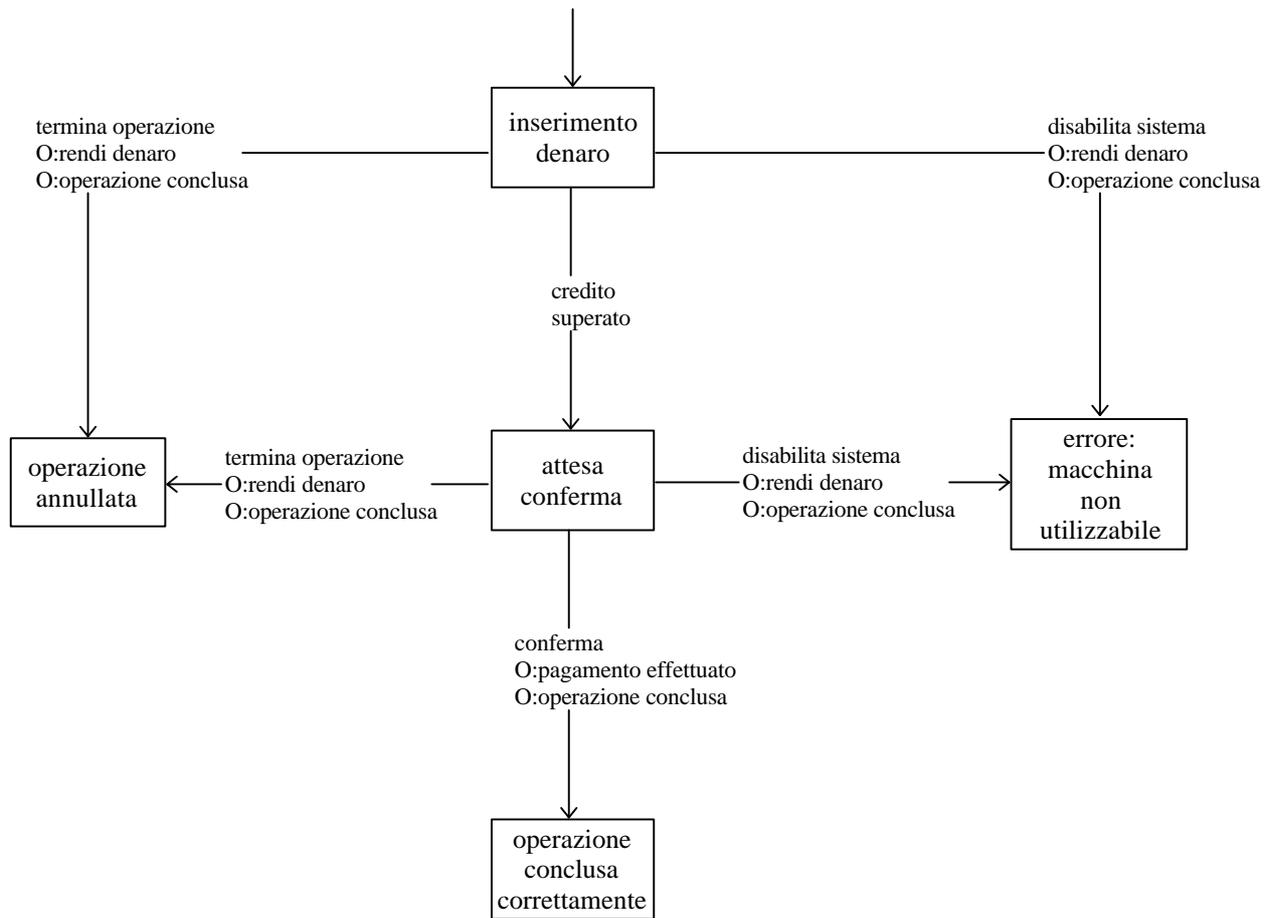


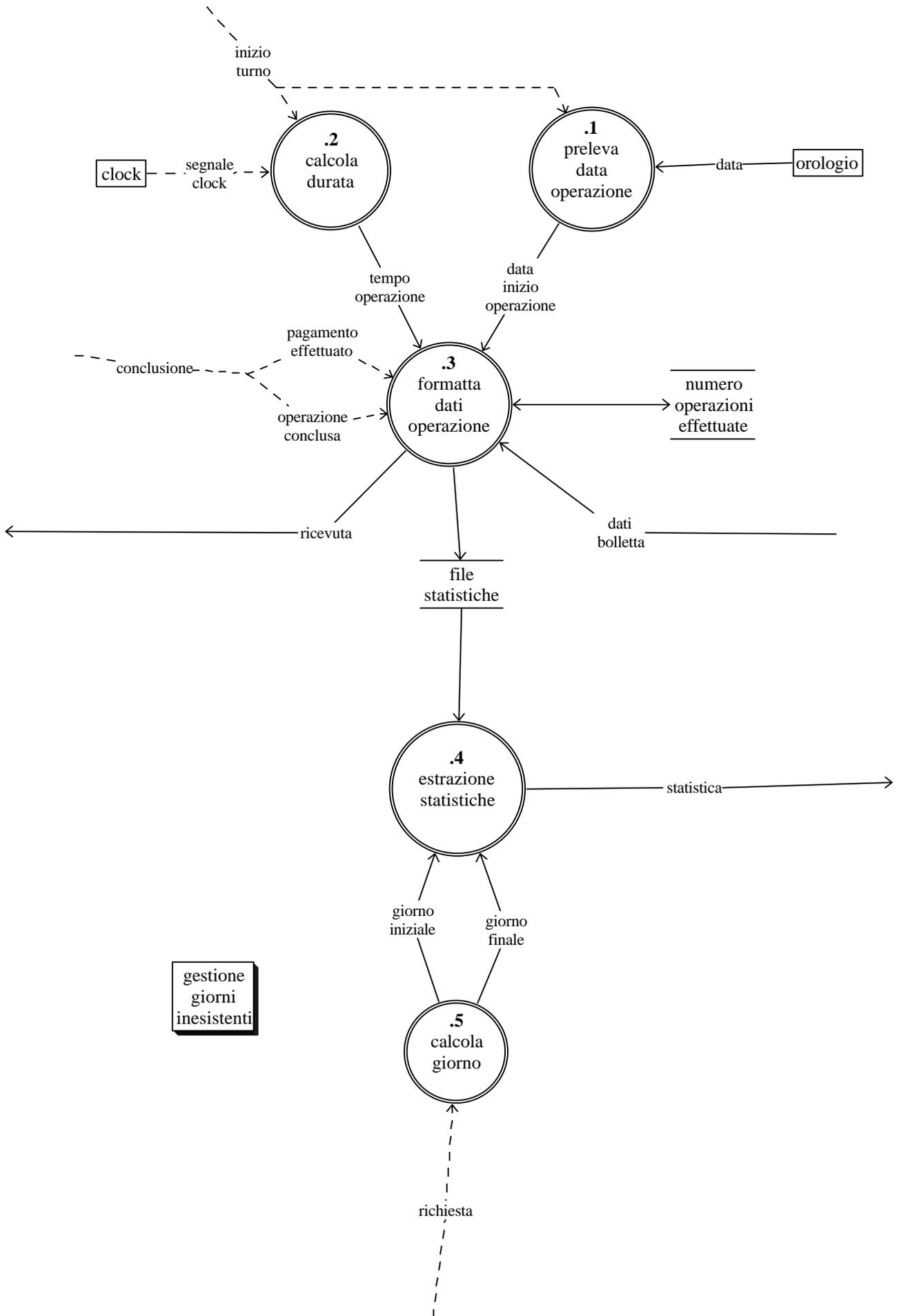


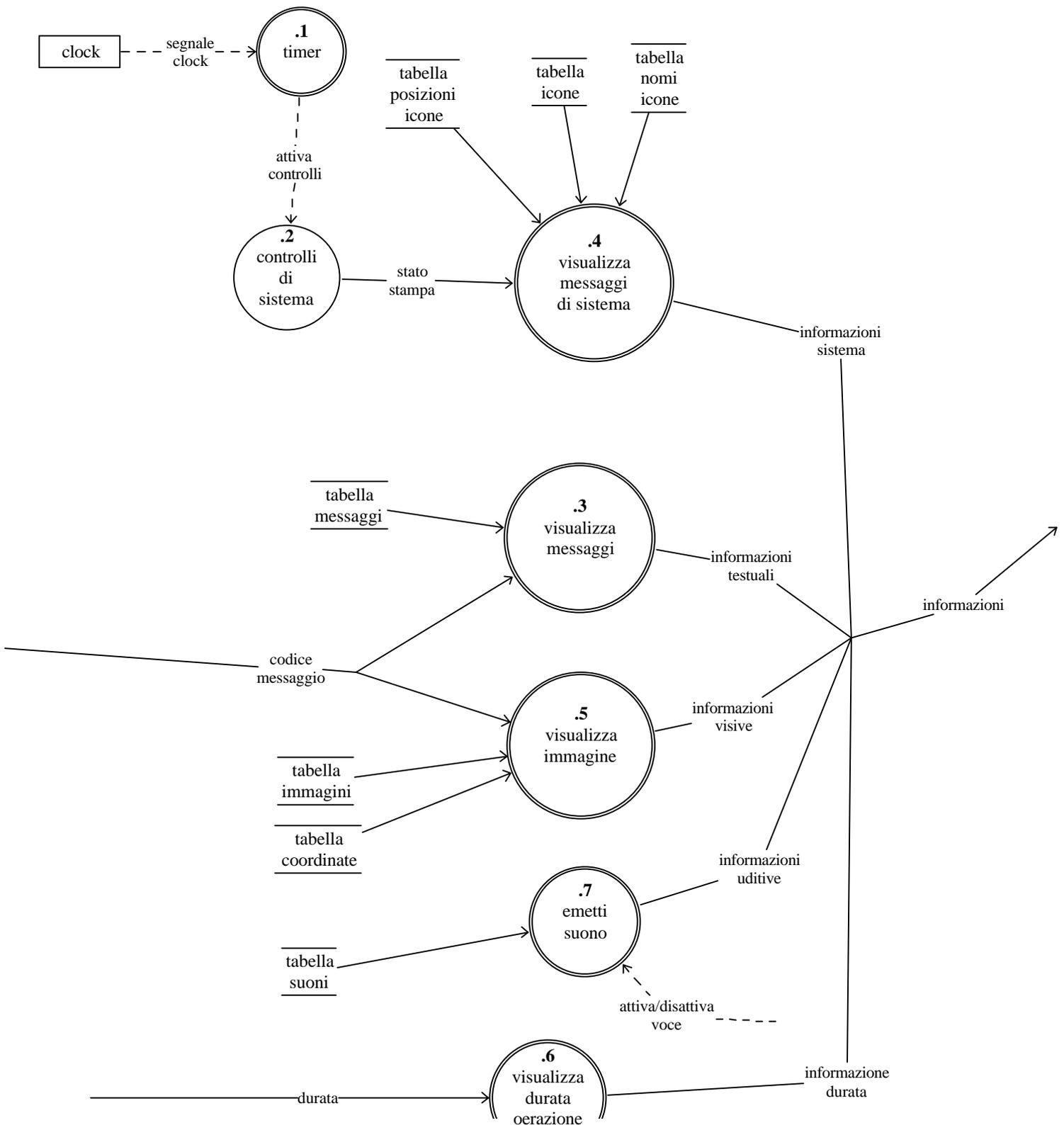


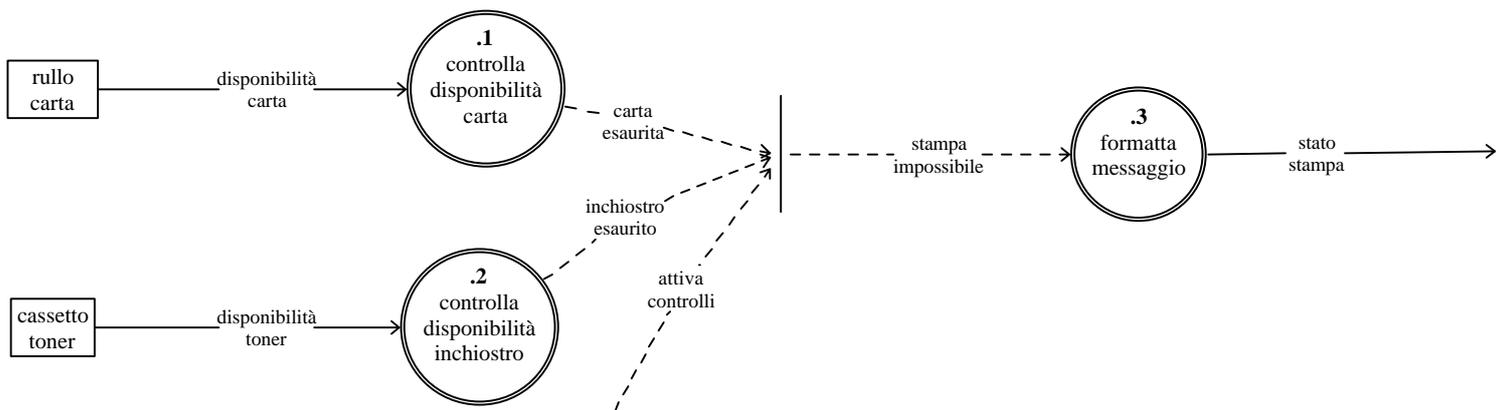












Input			Output
tempo esaurito	annulla operazione	bolletta non leggibile	termina operazione
non_attivo	non_attivo	non_attivo	non_attivo
OTHERS			attivo

## **Process Specification (PSPEC)**

### **PSPEC: determinazione\_tipo\_bolletta**

INPUT:bolletta

READ:tabella\_modelli

ISSUE:bolletta\_non\_leggibile

OUTPUT:tipo\_bolletta

Questo modulo tenta di riconoscere la bolletta tra i modelli in ingresso.

Un modello descrive i tratti caratteristici di una bolletta: posizione codice a barre, posizione e tipo di simbolo.

SE la bolletta non è riconosciuta o non è leggibile ALLORA  
viene sollevata una eccezione

### **ALTRIMENTI**

si determinano i punti chiave

si confrontano i punti chiave con la tabella dei modelli

in uscita si fornisce il codice corrispondente al tipo di bolletta.

### **PSPEC: determinazione\_valore\_moneta**

READ:tabella\_valori\_monete

INPUT:monete

OUTPUT:valore

determina il valore della moneta confrontando il tipo nella tabella ed estraendo il valore corrispondente.

### **DS: controlla\_disponibilità.d**

INPUT:differenza

RECOGNIZE:pagamento\_effettuato

INPUT:totale\_introdotta

UPDATE:archivio\_denaro\_in\_cassa

OUTPUT:codice\_messaggio

OUTPUT:resto

GENERATE:azzera

UPDATE:archivio\_denaro\_in\_cassa

Nel calcolare la disponibilità tiene conto delle eventuali monete ricevute dal pagamento, se questo dovesse avvenire. In questa maniera si evita il controllo supplementare di espellere le monete in più superflue.

**PSPEC: controlla\_disponibilità**

INPUT:differenza  
 RECOGNIZE:pagamento\_effettuato  
 INPUT:totale\_introdotta  
 UPDATE:archivio\_denaro\_in\_cassa  
 OUTPUT:codice\_messaggio  
 GENERATE:azzera  
 UPDATE:archivio\_denaro\_in\_cassa  
 OUTPUT:resto

1) somma temporaneamente ad archivio\_denaro\_in\_cassa il totale\_introdotta da questa tabella temporanea

2) tenta di trovare la combinazione di monete dal valore massimo ma uguale o minore all'importo.

- 3) SE valore è UGUALE ALLORA  
     visualizza messaggio 'RESTO DISPONIBILE'  
 ALTRIMENTI  
     visualizza messaggio 'RESTO NON DISPONIBILE'
- 4) SE pagamento\_effettuato ALLORA  
     aggiorna denaro\_in\_cassa  
     fornisce resto

**PSPEC: gestione\_turno**

RECOGNIZE:prenotazione\_turno  
 RECOGNIZE:termina\_operazione  
 INPUT:tabella\_turno  
 OUTPUT:codice\_messaggio

Questo modulo si occupa di gestire una lista di nominativi, mostrandoli a schermo quando riceve il segnale di operazione terminata.

**PSPEC: calcola\_totale\_introdotta**

INPUT:valore  
 INPUT:totale\_introdotta  
 OUTPUT:denaro  
 OUTPUT:totale  
 ricava il valore dell'intero archivio 'totale\_introdotta' sommando ad uno ad uno i valori ricavati dal modulo ricava\_valore

**PSPEC: determina\_resto**

INPUT:importo  
 INPUT:totale  
 OUTPUT:differenza  
 Effettua la differenza tra l'importo e il valore\_introdotta in soldi, ovvero calcola il resto

**PSPEC: attendi\_superamento\_credito**

INPUT:importo

GET:denaro

GET:valore

GENERATE:credito\_superato

GENERATE:inizio\_turno

PRODUCE:codice\_messaggio

PUT:denaro

- 1) genera inizio\_turno
- 2) usa determina\_valore
- 3) confronta valore con importo; FINCHE' valore è minore  
visualizza messaggio 'INSERIRE DENARO'
- 4) genera credito\_superato

**PSPEC: contatore\_tabella**

RECOGNIZE:azzera

INPUT:denaro

OUTPUT:totale\_introdotta

RIPETE

somma denaro in tabella totale\_introdotta

FINCHE' not azzera

azzera la tabella

**PSPEC: formatta\_dati\_operazione**

RECOGNIZE:operazione\_conclusa

RECOGNIZE:pagamento\_effettuato

INPUT:tempo\_operazione

INPUT:dati\_bolletta

INPUT:data\_inizio\_operazione

UPDATE:numero\_operazioni\_effettuate

OUTPUT:ricevuta

OUTPUT:file\_statistiche

UPDATE:numero\_operazioni\_effettuate

Questo modulo si occupa di tre attività:

- 1) aggiornare il database statistico, raccogliendo dati dai moduli complementari sugli estremi e sull'esito dell'operazione.
- 2) aggiornare il numero di operazioni svolte (sia riuscite che non).
- 3) formattare i dati che devono essere stampati nella ricevuta.

**PSPEC: calcola\_durata**

RECOGNIZE:inizio\_turno

RECOGNIZE:segnale\_clock

OUTPUT:tempo\_operazione

modulo che si occupa di calcolare la durata dell'intera operazione indipendentemente dal modulo schedulatore. Non è altro che un timer.

**PSPEC: preleva\_data\_operazione**

RECOGNIZE:inizio\_turno

INPUT:data

OUTPUT:data\_inizio\_operazione

questo modulo preleva dall'orologio le informazioni sulla data dell'operazione e le formatta rendendole disponibili agli altri moduli.

**PSPEC: controlla\_spazio\_sul\_disco\_rigido**

RECOGNIZE:controllo\_periodico\_periferiche

INPUT:spazio

GENERATE:spazio\_su\_disco\_insufficiente

**PSPEC: controlla\_disponibilità\_spazio**

RECOGNIZE:controllo\_periodico\_periferiche

INPUT:spazio\_in\_cassa

GENERATE:spazio\_monete\_insufficiente

**PSPEC: controlla\_accensione\_generatore**

INPUT:disposizione

RECOGNIZE:controllo\_periodico\_periferiche

GENERATE:generatore\_acceso

**PSPEC: comanda\_tagliatrice**

RECOGNIZE:conclusione

INPUT:posizione\_taglio

INPUT:tipo\_bolletta

GENERATE:taglia

**PSPEC: stampa\_bollo**

INPUT:tipo\_bolletta

INPUT:posizione\_bollo

RECOGNIZE:conclusione

GENERATE:bolla

**PSPEC: lettura\_informazioni**

READ:tabella\_posizioni\_codici\_a\_barre

INPUT:tipo\_bolletta

INPUT:bolletta

ISSUE:bolletta\_non\_leggibile

OUTPUT:codice\_a\_barre

Questo modulo preleva il codice a barre dalla bolletta e lo consegna in uscita.

**PSPEC: decodificatore\_codice\_a\_barre**

READ:maschera\_codici\_a\_barre

INPUT:codice\_a\_barre

OUTPUT:dati\_bolletta

OUTPUT:importo

GENERATE:abilita

ISSUE:bolletta\_non\_leggibile

**PSPEC: determinazione\_valore\_banconota**

INPUT:banconote

READ:tabella\_valori\_banconote

OUTPUT:valore

determina il valore della banconota confrontando il tipo nella tabella ed estraendo il valore corrispondente.

**PSPEC: estrazione\_statistiche**

INPUT:giorno\_finale

INPUT:giorno\_iniziale

READ:file\_statistiche

OUTPUT:statistica

questo modulo si occupa di

- 1) interrogare la base di dati per l'intervallo di tempo richiesto
- 2) effettuare una media algebrica dei dati ottenuti
- 3) fornire i dati in uscita

**PSPEC: calcola\_giorno**

RECOGNIZE:richiesta

OUTPUT:giorno\_iniziale

OUTPUT:giorno\_finale

In base alla richiesta in ingresso decifra l'intervallo di tempo e fornisce il giorno iniziale e il giorno finale dell'intervallo di tempo in cui deve essere calcolata la statistica

**DS: gestione\_statistiche.d**

RECOGNIZE:richiesta

RECOGNIZE:inizio\_turno

INPUT:dati\_bolletta

RECOGNIZE:conclusione

OUTPUT:ricevuta

OUTPUT:statistica

La gestione delle statistiche è intesa non soltanto come modulo atto ad immagazzinare i dati relativi all'operazione ( generalità - importo - data - orario - durata - fine operazione ) ma gestisce anche la gestione delle statistiche da fornire in uscita all'utente sotto forma di ricevuta.

**PSPEC: controlla\_disponibilità\_carta**

INPUT:disponibilità\_carta

GENERATE:carta\_esaurita

verifica che vi sia ancora carta per stampare un'altra ricevuta

**PSPEC: controlla\_disponibilità\_inchiostro**

INPUT:disponibilità\_toner

GENERATE:inchiostro\_esaurito

verifica che vi sia ancora inchiostro per stampare una ricevuta

**PSPEC: formatta\_messaggio**

RECOGNIZE:stampa\_impossibile

OUTPUT:stato\_stampa

**PSPEC: timer**

RECOGNIZE:segnale\_clock

INPUT:tempo\_per\_operazione

RECOGNIZE:inizio\_turno

GENERATE:attiva\_controlli

OUTPUT:durata

GENERATE:tempo\_esaurito

GENERATE:controllo\_periodico\_periferiche

modulo indipendente che si occupa di far svolgere a intervalli di tempo costante dei controlli sul sistema.

**PSPEC: visualizza\_messaggi\_di\_sistema**

INPUT:tabella\_icone

INPUT:tabella\_posizioni\_icone

INPUT:tabella\_nomi\_icone

INPUT:stato\_stampa

OUTPUT:informazioni\_sistema

questo modulo si occupa di mostrare su schermo alcune icone inerenti delle operazioni disabilitate del sistema. Attualmente l'unico controllo viene svolta sulla possibilità di stampa, ma ulteriori controlli possono essere facilmente aggiunti.

**PSPEC: visualizza\_messaggi**

INPUT:tabella\_messaggi

INPUT:codice\_messaggio

OUTPUT:informazioni\_testuali

**PSPEC: visualizza\_immagine**

INPUT:tabella\_coordinate

INPUT:tabella\_immagini

INPUT:codice\_messaggio

OUTPUT:informazioni\_visive

**PSPEC: emetti\_suono**

INPUT:tabella\_suoni

RECOGNIZE:attiva/disattiva\_voce

OUTPUT:informazioni\_uditive

**PSPEC: visualizza\_durata\_oerazione**

INPUT:durata

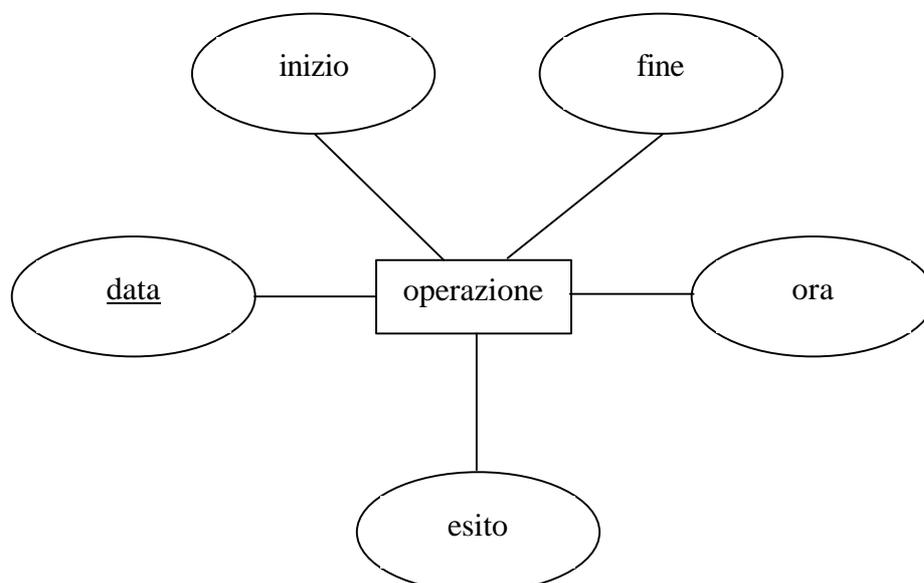
OUTPUT:informazione\_durata

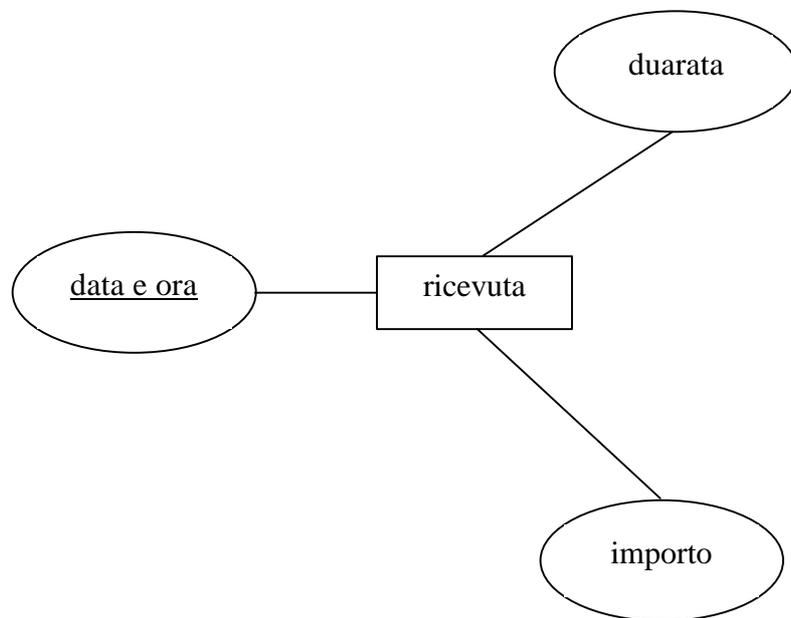
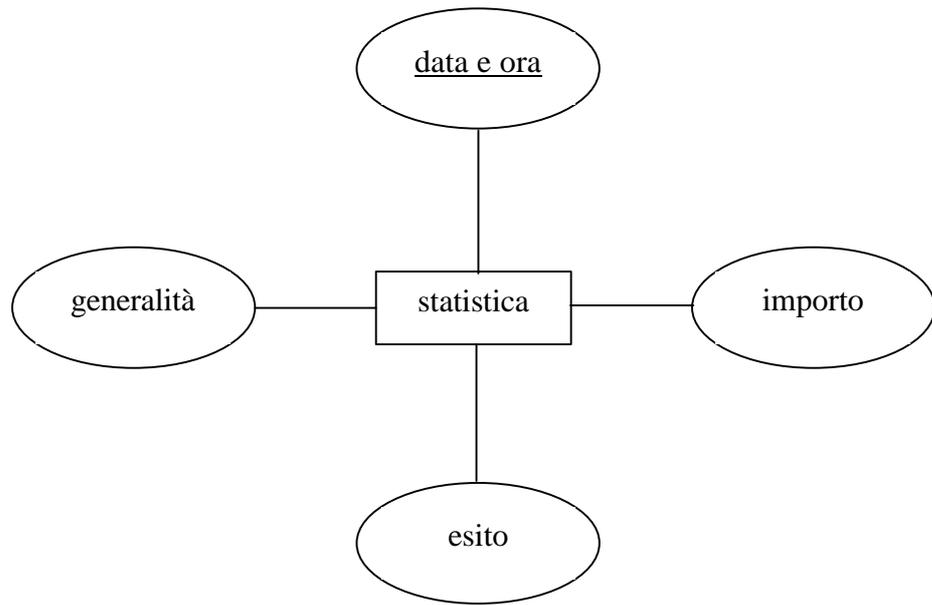
Sul video c'è un campo riservato alla visualizzazione del tempo trascorso dall'inizio dell'operazione fino al momento attuale, e accanto il tempo totale disponibile.

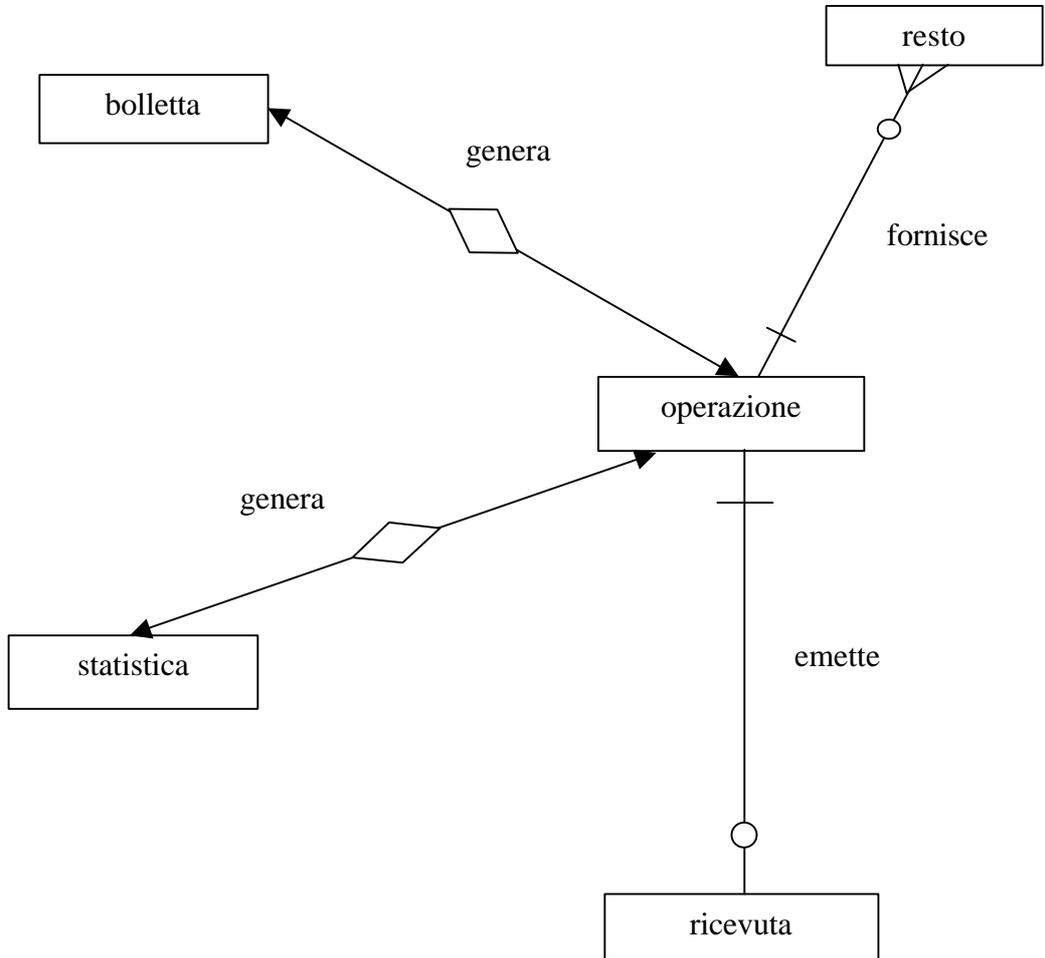
## Strutture dati

Gli archivi presenti nel sistema sono molteplici e prevedono la possibilità di cambiare certi parametri di funzionamento anche dopo la codifica, semplicemente intervenendo sui file opportuni.

- 1) *archivio statistiche*, vedere la sezione database
- 2) *archivio monete*, insieme di immagini e parametri che permettono l'individuazione delle monete autentiche e lo scarto dei falsi.
- 3) *archivio banconote*, insieme di immagini e parametri che permettono l'individuazione delle banconote autentiche e lo scarto dei falsi.
- 4) *archivio valore denaro*, collega ad ogni codice di moneta o banconota introdotta, il suo valore corrente.
- 5) *archivio messaggi testuali*, raccolta di stringhe da visualizzare sullo schermo.
- 6) *archivio messaggi sonori*, raccolta di file sonori.
- 7) *archivio immagini ed icone*, raccolta di immagini di diverse dimensioni.







## **Descrizione architetture**

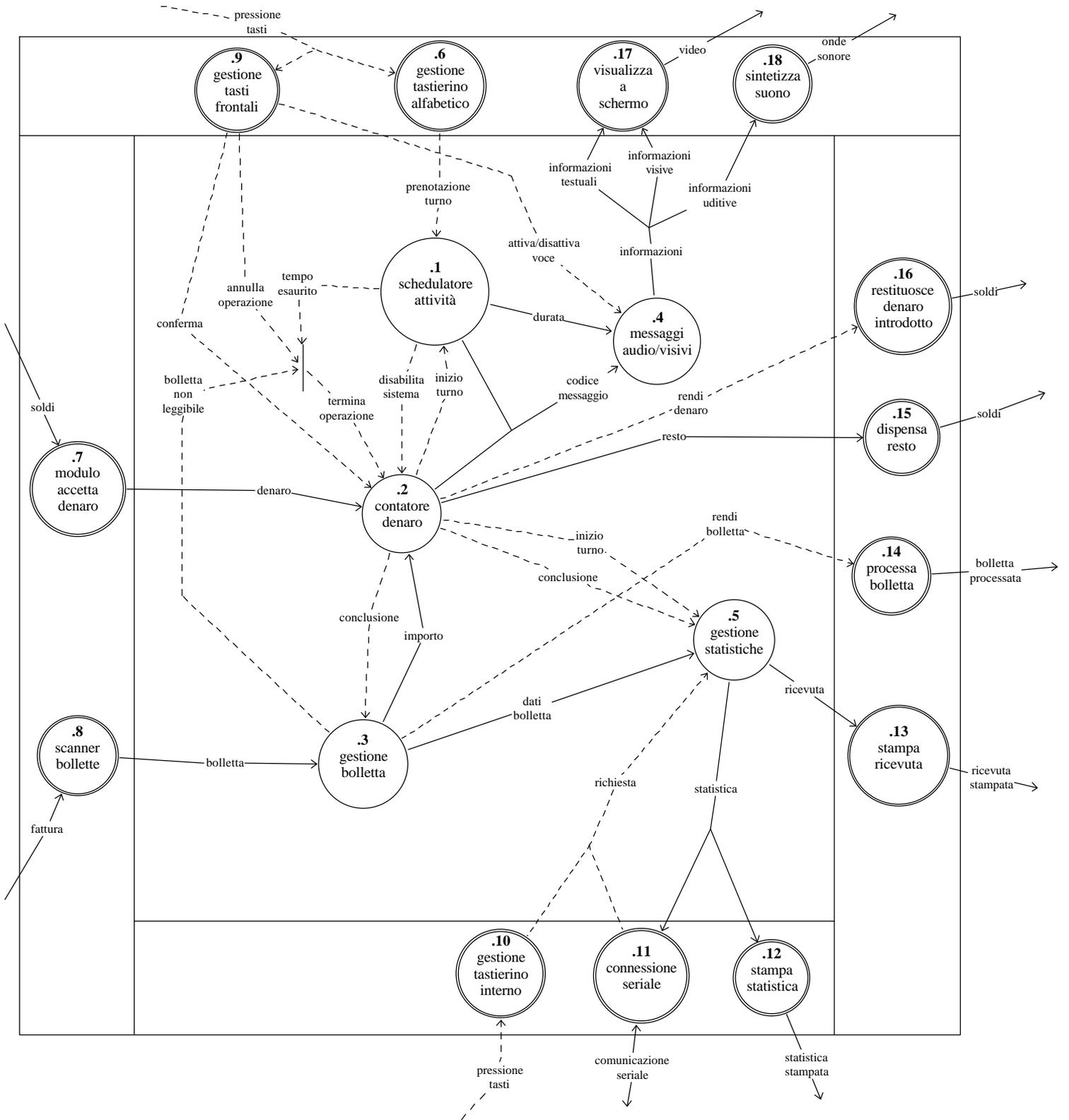
Il Turbo Case mette a disposizione una serie di grafici in cui è possibile definire l'architettura del sistema. Per arrivare a questo genere di grafici si è partito dall'EDFD, in cui sono state racchiusi i moduli in delle superbolle. Ogni superbolla si occupa di un particolare aspetto del problema. A partire da questo grafico si arriva all'ACD (Architecture Context Diagram) che va letto effettuando un parallelo con il Context Diagram. Da questo grafico si può evincere l'interazione del sistema con l'hardware. Ad un livello diverso si pone l'AICD (Architecture Interconnect Context Diagram) in cui viene mostrato il tipo di connessione tra sistema e hardware. I tipi utilizzati sono soltanto due:

- 1) linea sottile: connessione elettrica
- 2) linea spessa: connessione meccanica

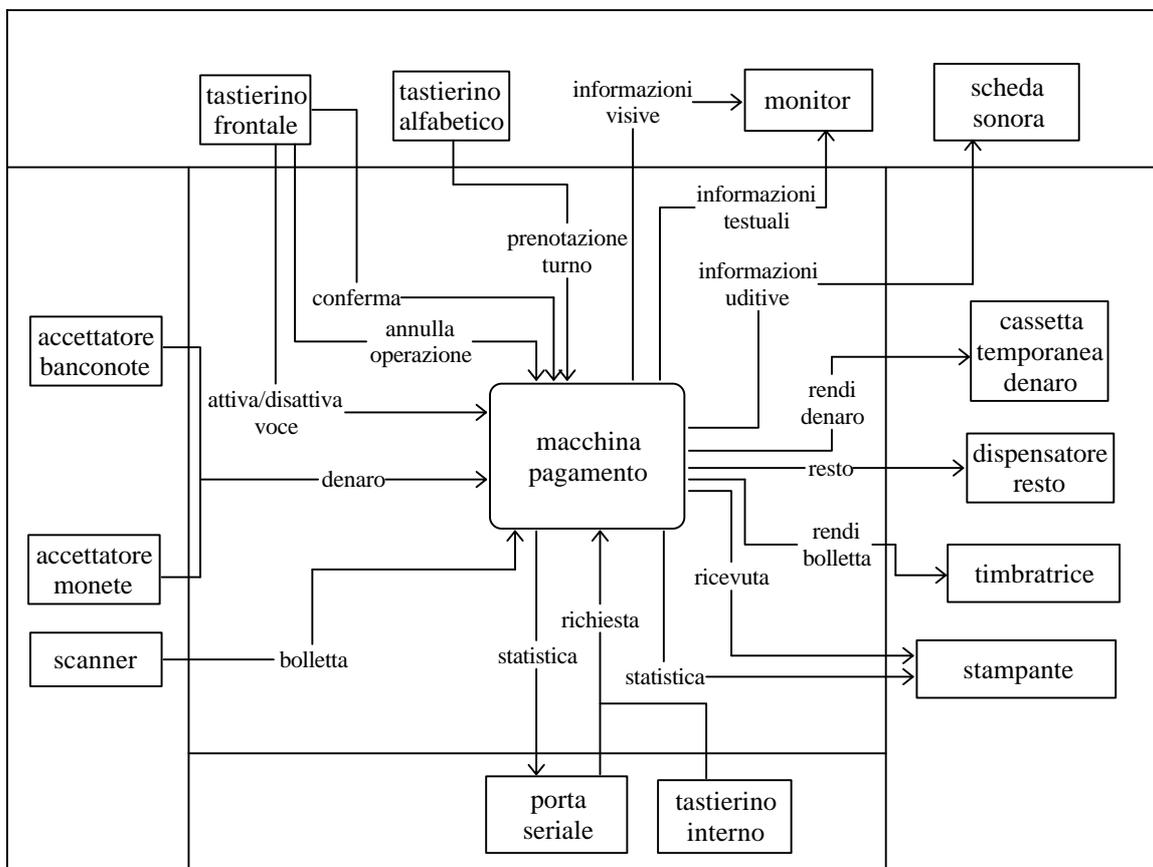
Dall'ACD deriva un'ulteriore grafico, l'AFD (Architecture Flow Diagram) che utilizza le superbolle dell'EDFD e ne visualizza le connessioni. Lo stesso grafico, ma con i tipi di connessione è l'AID (Architecture Interconnect Diagram).

## **L'hardware**

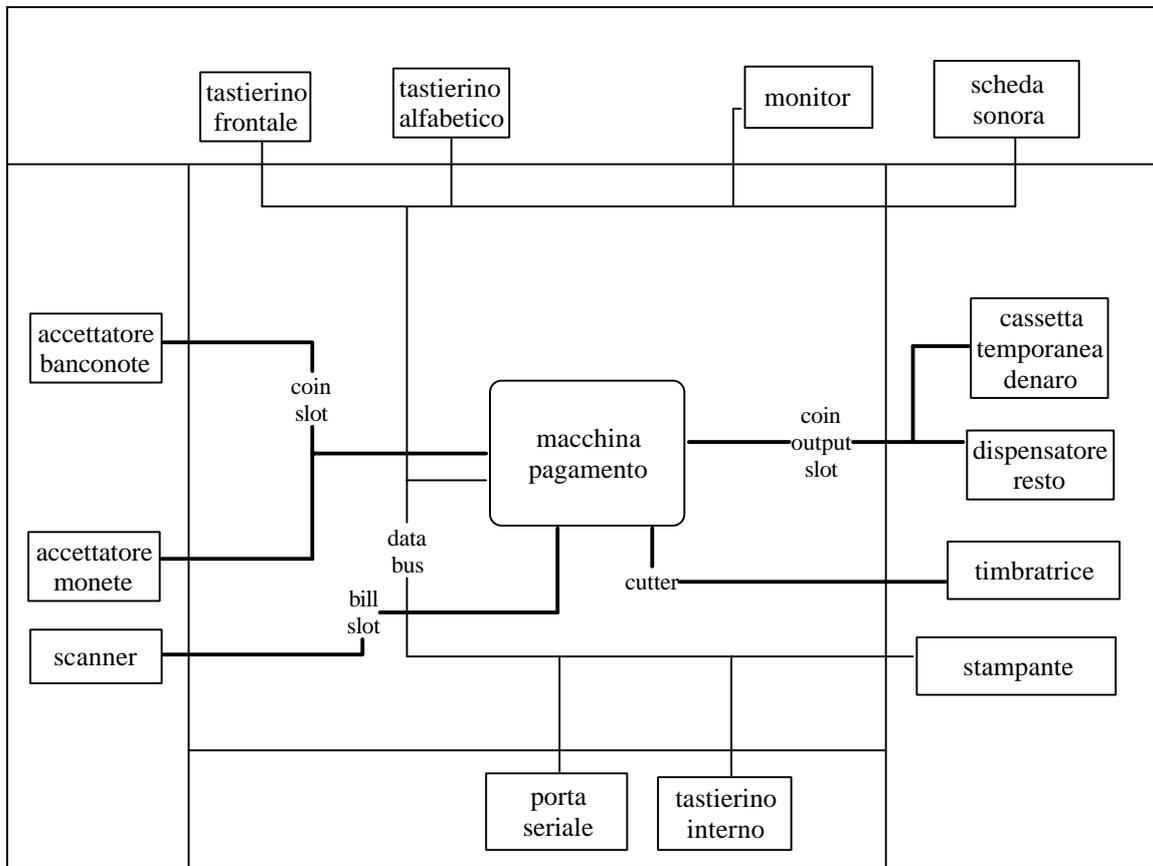
- 1) Personal computer basato su processore Pentium, equipaggiato con 16 Mb RAM e 3 Gb di Hard Disk.
- 2) Monitor 14" a fosfori verdi con schermo antiriflesso.
- 3) Modulo per il trattamento delle bollette, deve caricare la bolletta, bollarla ed eventualmente tagliarla.
- 4) Scanner a media risoluzione (300 DPI a scala di grigi) per prelevare il codice a barre ed eventuali altri dati dalla bolletta
- 5) Cassa fatture, per conservare la parti della fattura trattenute.
- 6) Lettore di banconote de tipo TP95 o EUREKA della Coges, già aggiornati per l'euro e facilmente riprogrammabili per futuri cambiamenti.
- 7) Riconoscitore di lega del tipo EUR/10 o EUR/10 Executive, con le stesse caratteristiche di sopra.
- 8) Cassa temporanea monete
- 9) Cassa automatica monete rendiresto del tipo EURO 2000
- 10) Cassa automatica banconote rendiresto
- 11) Stampante ricevuta su carta termica a rullo
- 12) Generatore supplementare energia UPS
- 13) Pulsantiera
- 14) Tastierino alfabetico
- 15) Display per caratteri alfabetici

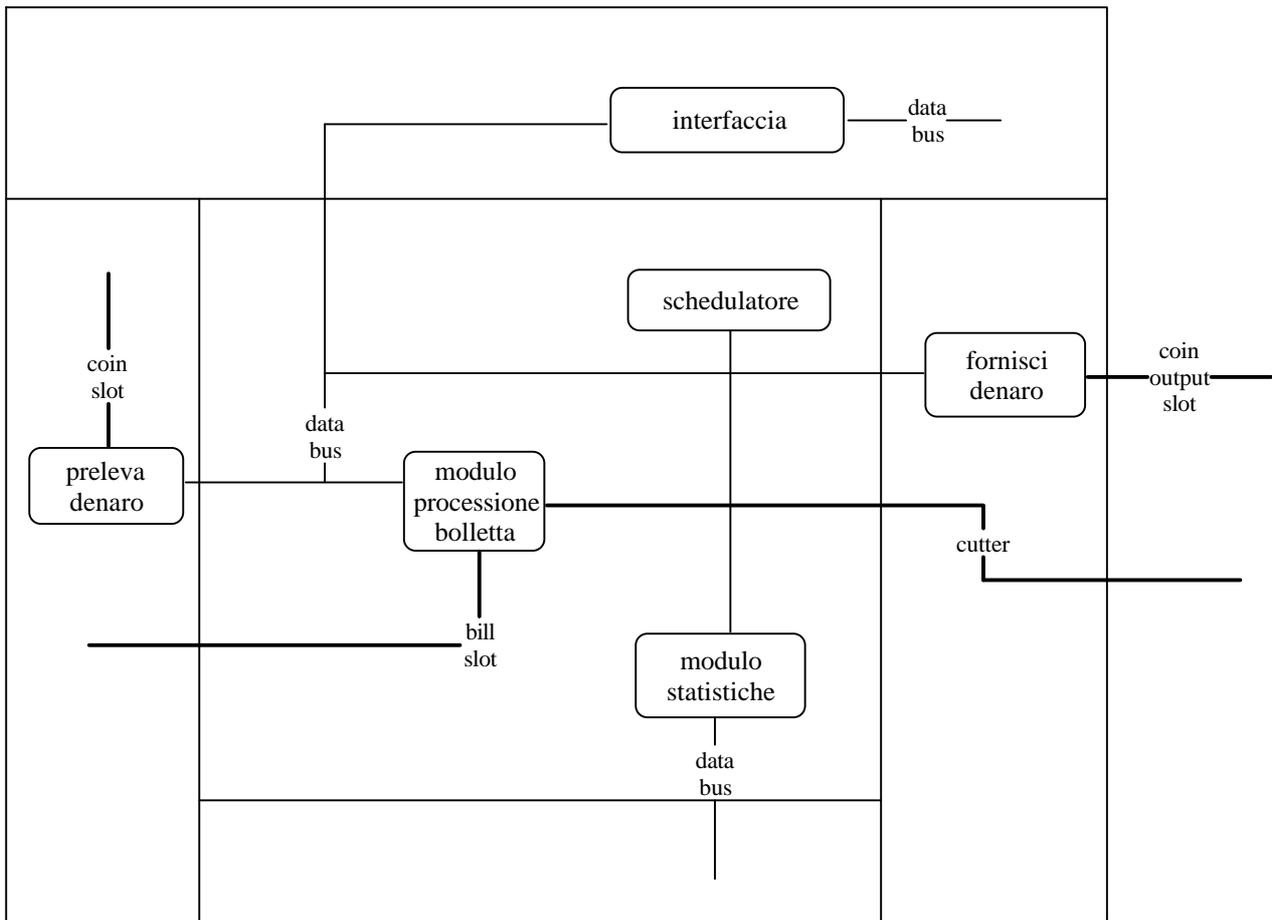












Req. Component	Architecture Component					
*****	interfaccia	preleva denaro	fornisci denaro	modulo statistiche	modulo processione bolletta	scheduler
gestione_tastierino_alfabetico	X					X
gestione_tasti_frontali	X					
visualizza_a_schermo	X					
sintetizza_suono	X					
contatore_denaro		X				
modulo_accetta_denaro		X				
dispensa_resto			X			
restituisce_denaro_introdotta			X			
gestione_statistiche				X	X	
gestione_tastierino_interno				X		
stampa_statistica				X		
stampa_ricevuta				X		
connessione_seriale				X		
gestione_bolletta					X	
scanner_bollette					X	
processa_bolletta					X	
scheduler_attività						X

Name	Definition	Type
abilita	segnale_booleano	Control
altezza	misura	
anno	[ 'min'..'max' ] *Un intero dovrebbe essere abbastanza grande da gestire la data per parecchio tempo*	
annuale	anno	
annulla operazione	DOMAIN: [ attivo   non_attivo ]	Data/ Control
archivio denaro in cassa	tabella_denaro	Data
attiva/disattiva voce	segnale_booleano	Data/ Control
attiva controlli	segnale_booleano	Control
avanza lista	segnale_booleano  DOMAIN: [ attivo   non attivo ]	Control
azzerata	segnale_booleano	Control
banconote	[ 1..32 ]	Data
bolla	segnale_booleano	Control
bolletta	immagine_bitmap	Data
bolletta non leggibile	DOMAIN: [ attivo   non_attivo ]	Control
bolletta processata	*la bolletta inserita può essere tagliata e timbrata, oppure restituita integra, a seconda delle esigenze.*	Data
bolletta timbrata	segnale_booleano	Control
byte	DOMAIN: integer	
carta esaurita	segnale_booleano	Control
codice a barre	immagine_bitmap	Data
codice messaggio	*La gestione dei messaggi è gestita mediante un codice, in quanto le situazioni che si possono presentare sono in numero limitato. La costante del numero massimo di messaggi è MAX_MESSAGGI* DOMAIN: integer	Data
cognome	DOMAIN: string	
comunicazione seriale	*Questo è un canale attraverso il quale si adotta un protocollo di comunicazione remota.*	Data
conclusione	[ pagamento_effettuato   operazione_conclusa ]	Control
conferma	segnale_booleano	Data/ Control
controllo periodico periferiche	segnale_booleano	Control
coordinata X	misura	
coordinata Y	misura	
coordinate	coordinata_X + coordinata_Y	

Name	Definition	Type
credito	DOMAIN: real $x \geq 0$	
credito superato	segnale_booleano	Control
data	anno + mese + giorno	Data
data inizio operazione	data	Data
dati bolletta	generalità + importo	Data
dati operazioni	generalità + data + ora + importo + risultato	
denaro	[ banconote   monete ] *Il denaro introdotto è di due tipi: monete o banconote.*	Data
differenza	DOMAIN: Real	Data
dimensioni bolletta	larghezza + altezza	
disabilita sistema	segnale_booleano	Control
disponibilità carta	DOMAIN: [ true   false ]	Data
disponibilità toner	DOMAIN: [ true   false ]	Data
disposizione	DOMAIN: [attivo   non attivo ]	Data
durata	tempo	Data
durata media	durata	
durata operazione	durata	
fattura	*E' la bolletta nel quale ci sono le informazioni sull'importo da pagare*	Data
file statistiche	dati_operazioni	Data
generalità	cognome + nome	
generatore acceso	segnale_booleano	Control
giornaliera	data	
giorno	[ 1..31 ]	
giorno finale	data	Data
giorno iniziale	data	Data
gradazione colore	[ 0 .. 23 ] *** vedere Teoria e tecnica di elaborazione dell'immagine per stabilire la profondità migliore per una immagine a gradazione di grigi ***	
immagine	immagine_bitmap	
immagine bitmap	1 { riga_di_pixel }	
importo	credito	Data
inchiostro esaurito	segnale_booleano	Control
informazione durata	DOMAIN: string	Data
informazioni	[ informazioni_testuali + informazioni_visive + informazioni_sistema + informazioni_uditive + informazione_durata ]	Data
informazioni sistema	[ nomeicona + posizione_immagine + immagine ]	Data
informazioni testuali	DOMAIN: string	Data

Name	Definition	Type
informazioni uditive	suono	Data
informazioni visive	[ posizione_immagine + immagine_bitmap ]	Data
inizio turno	segnale_booleano DOMAIN: [ attivo   non attivo ]	Control
intervallo	giorno_iniziale + giorno_finale	
larghezza	misura	
maschera codici a barre	{ larghezza }	Data
mensile	anno + mese	
mese	[ 1..12 ]	
minuti	[ 0 .. 59 ]	
misura	[ misura_cm   misura_pollici ]	
misura cm	DOMAIN: real	
misura pollici	DOMAIN: real	
monete	[ 1..4 ] *Il dispensatore di resto accetta solo 4 tipi di monete*	Data
nome	DOMAIN: string	
nome icona	DOMAIN: string	
nominativo	DOMAIN: string	
numero clienti	DOMAIN: integer	
numero operazioni effettuate	DOMAIN: integer	Data
onde sonore	*E' l'uscita degli altoparlanti, ovvero i suoni e le voci che si ascoltano*	Data
operazione conclusa	segnale_booleano	Control
ora	ore + minuti + secondi	
ore	[ 0..23 ]	
pagamento effettuato	segnale_booleano	Control
pixel	gradazione_colore	
posizione bollo	coordinate	Data
posizione immagine	coordinate	
posizione logo	coordinate	
posizione taglio	coordinate	Data
prenotazione turno	nominativo	Data/ Control
pressione tasti	segnale_booleano	Control
quantità	DOMAIN: integer	
rendi bolletta	segnale_booleano	Data/ Control

Name	Definition	Type
rendi denaro	segnale_booleano	Data/ Control
resto	tabella_denaro	Data
ricevuta	generalità + importo + data + ora + durata_operazione	Data
ricevuta stampata	*Dalla stampante esce un rullo di carta sul quale vengono stampate le informazioni relative all'operazione effettuata*	Data
richiesta	[ annuale   mensile   giornaliera   intervallo ] + tipo_statistica	Data/ Control
riga di pixel	1 { pixel }	
risultato	DOMAIN: [ operazione_riuscita   operazione annullata ]	
secondi	[ 0..99 ]	
segnale booleano	*Un segnale booleano non porta alcuna informazione, e serve esclusivamente per attivare o disattivare un processo* DOMAIN: [ BASSO   ALTO ]	
segnale clock	segnale_booleano	Control
soldi	*Sono i crediti fisici (monete o banconote) che vengono introdotti nella macchina, o che si prelevano da essa*	Data
spazio	DOMAIN: integer	Data
spazio in cassa	DOMAIN: integer	Data
spazio monete insufficient	e segnale_booleano	Control
spazio su disco insufficient	e segnale_booleano	Control
stampa impossibile	segnale_booleano	Control
statistica	numero_clienti + durata_media + risultato	Data
statistica stampata	*Dalla stampante esce un rullo di carta sul quale vengono stampate le statistiche richieste*	Data
stato stampa	DOMAIN: [ possibile   impossibile ]	Data
successivo	DOMAIN: [ attivo   non attivo ]	Control
suono	{ byte }	
tabella coordinate	coordinate	Data
tabella denaro	8 { quantità } 8 *Una tabella denaro indica il quantitavo di monete introdotte per ogni tipo. In questa maniera è possibile calcolare la moneta da espellere ed è possibile aggiornare l'archivio delle monete possedute*	
tabella icone	immagine	Data
tabella immagini	immagine_bitmap	Data
tabella messaggi	DOMAIN: string	Data
tabella modelli	dimensioni_bolletta + posizione_logo	Data
tabella nomi icone	nomeicona	Data
tabella posizioni codici a barre	coordinate	Data

Name	Definition	Type
tabella posizioni icone	coordinate	Data
tabella suoni	suono	Data
tabella turno	nominativo	Data
tabella valori banconote	credito	Data
tabella valori monete	credito	Data
taglia	segnale_booleano	Control
taglio terminato	segnale_booleano	Control
tempo	ore + minuti + secondi	
tempo esaurito	DOMAIN: [ attivo   non_attivo ]	Control
tempo operazione	durata	Data
tempo per operazione	tempo	Data
termina operazione	DOMAIN: [ attivo   non_attivo ]	Control
tipo bolletta	*teoricamente non esiste alcun limite prefissato al numero di tipi di bollette memorizzati, tuttavia tale limite è fissato dalla costante MAX_TIPI_BOLLETTE* DOMAIN: integer	Data
tipo statistica	DOMAIN: [ casi_riusciti   casi_peggiori ]	
totale	DOMAIN: real	Data
totale introdotto	tabella_denaro	Data
valore	DOMAIN: real	Data
video	*Ciò che appare sul video*	Data

## Progettazione

Il linguaggio scelto per questa applicazione è un linguaggio orientato agli oggetti, ad alto livello. Tra le possibili scelte sono disponibili: Pascal ad oggetti, Java , C++ o Visual Basic. Una caratteristica del sistema operativo su cui deve lavorare il programma è che deve essere 'ad eventi'. Ovvero le situazioni di attesa si risolvono lasciando il controllo al sistema operativo, che poi si occupa di mandare gli eventi opportuni. Un gestore degli eventi si occuperà di richiamare le procedure opportune. Gli eventi significativi del nostro esempio sono:

- tasto annulla
- tasto conferma
- tempo esaurito
- introduzione denaro
- introduzione bolletta

Un esempio di sistema operativo che si adatta a questo tipo di programmazione è Windows (dalla versione 3.1 alla più aggiornata 98 o 2000).

### **Strategia adottata**

A differenza della fase di analisi, nella quale la strategia adottata è stata essenzialmente di tipo top-down, qui si è avanzato in maniera incrementale, ovvero *bottom-up*. Si è infatti definito prima le classi base che rappresentano le fondamenta del programma, e poi si è salito verso i moduli più superficiali. A volte, tuttavia si è tornato su alcune decisioni, scomponendo un modulo in più moduli, per aumentare la coesione interna dei moduli.

### **I moduli di base**

I moduli qui elencati non sarà necessario realizzarli, in quanto si suppone che siano già pronti in una libreria di moduli:

- data
- ora
- coda
- timer
- immagine
- suono

```
module data;
```

```
uses
```

```
exports
```

```
    type data = ?
```

```
    function data_valida(in giorno : data) : Boolean;
```

```
    function somma(in giorno : data, in giorni : Integer ) :
```

```
data;
```

```
    function dopo(in giorno1,giorno2 : data) : Boolean;
```

```
    function prima(in giorno1,giorno2 : data) : Boolean;
```

```
    function uguale(in giorno1,giorno2 : data) : Boolean;
```

```
    procedure imposta_secondi(in giorno : data, in secondi :
```

```
Integer);
```

```

    function imposta_secondi(in giorno : data, in secondi :
Integer ) : data;
    function stringa(in giorno : data) : string;
implementation
    -- Il modulo rappresenta uno specifico istante di tempo --
    -- con una precisione fino al millesimo di secondo --
end data;

module ora;
uses
exports
    type ora = ?
    function preleva_ora() : ora;
    function secondi(in orario: ora) : Integer;
    function stringa(in orario : data) : string;
implementation
    -- Il modulo rappresenta uno specifico istante del giorno
    --
    -- con una precisione fino al millesimo di secondo --
end data;

module timer;
uses
exports
    procedure azzera();
    procedure avvia();
    procedure ferma();
    function preleva_secondi() : Integer;
    procedure aggiungiLanciaEvento(evento : Evento, secondi :
integer);
    procedure abilita();
    procedure disabilita();
end data;

generic module coda(tipo);
uses
exports
    procedure aggiungi_in_cosa(in oggetto : tipo);
    function preleva_in_testa() : tipo;
end data;

module immagine;
uses
exports
    procedure carica(in nome_file : string);
    procedure visualizza(in coordinate : punto);
    function preleva_Altezza() : Integer;
    function preleva_Larghezza() : Integer;
end data;

module suono;

```

```
uses
exports
    procedure carica(in nome_file : string);
    procedure riproduci();
    procedure ferma();
end data;

module thread;
uses
exports
    procedure run();
    procedure stop();
    procedure start();
end data;
```

## ***I moduli da realizzare***

La lista di moduli qui presente è quella che effettivamente va codificata. I moduli sono raggruppati secondo la logica descritta nei DFD. Le bolle del DFD rappresentano una sorta di contenitore logico dei moduli fisici, che sono quelli di seguito elencati.

### ***Lettura bolletta***

```

module analizza_immagine inherits immagine;
uses
exports
  const MAX_TIPO_BOLLETTA;
  type tipo_bolletta = [1..MAX_TIPO_BOLLETTA];

  function riconosci_tipo_bolletta(in bolletta : immagine) :
tipo_bolletta;
    raise bolletta_nonleggibile;
    -- questa funzione effettua dei controlli sulla dimensione --
    -- e sulla posizione dei simboli nella bolletta per --
    -- determinare il tipo di bolletta tra quelli memorizzati --
    -- se la bolletta non è leggibile questo viene comunicato --
    -- mediante una eccezione --

    function preleva_codice_barre(in bolletta : immagine) :
immagine;
    -- questa funzione localizza il codice a barre all'interno --
    -- della bolletta e lo riporta in uscita --

    function interpreta_codice_barre(in bolletta : immagine) :
real;
    -- a seconda della scelta implementativa questa funzione --
    -- può essere COMPRATA o realizzata --

    function ricava_dati_bolletta(in bolletta : immagine) :
dati_bolletta;
    -- a seconda della scelta implementativa questa funzione --
    -- può essere COMPRATA o realizzata: si tratta di un modulo --
    -- per estrarre dati supplementari dalla bolletta --
    -- come ad esempio i dati anagrafici --

implementation
  -- questo modulo contiene tutti i controlli necessari --
  -- per estrarre i dati utili dalla bolletta --
composed of
end analizza_immagine;

```

```

module gestione_bolletta;
uses accunulo_credito;
      gestione_messaggi;
      analizza_immagine;
      gestione_turno
exports
  procedure processa_bolletta();
  -- questa procedura viene attivata dall'evento provocato --
  -- dall'inserimento della bolletta --

  procedure inserita_bolletta();
  -- questa procedura viene attivata dall'evento provocato --
  -- dall'inserimento della bolletta --

implementation
  -- questa modulo si occupa di processare i dati provenienti --
  -- dalla bolletta, di comunicare l'importo al modulo conta --
  -- denaro e di espellere la bolletta. Si occupa anche del --
  -- trattamento del caso speciale in cui la bolletta non sia --
  -- leggibile --

composed of
end gestione_bolletta;

```

## ***Contatore denaro***

```

module denaro;
uses
exports
  const MAX_TIPI_DENARO;
  -- numero di tubi di denaro presenti --

  function valore(in tipo_denaro: [1..MAX_ TIPI_DENARO]) :
real;
  -- fornisce il valore del tipo di denaro fornito --

implementation
  -- lo scopo di questo modulo è di fare il mapping --
  -- tra l'identificativo del denaro (numero intero) --
  -- e il valore della moneta o banconota --
  -- corrispondente (numero in virgola mobile) --
composed of
end denaro;

```

```

module tabella_denaro;
uses denaro;
exports
  const MAX_QUANTITA;
  -- massima quantità di monete per tubo di denaro --

  procedure azzera();
  -- pone a zero tutte le quantità --
  -- corrispondente al 'tipo denaro' --

  procedure aggiungi(in tipo_denaro : [1..
MAX_TIPI_DENARO]);
  -- incrementa di uno la quantità di denaro --
  -- corrispondente al 'tipo denaro' --

  procedure sottrai(in tipo_denaro : [1.. MAX_TIPI_DENARO]);
  -- decrementa di uno la quantità di denaro --
  -- corrispondente al 'tipo denaro' --
  -- Il valore non può scendere sotto zero --

  function quantita(in tipo_danaro : [1.. MAX_TIPI_DENARO])
: [0.. MAX_QUANTITA];
  -- fornisce la quantità di danaro del tipo sezionato --
  -- presente nella tabella --

  function valore() : real;
  -- fornisce la quantità di danaro del tipo sezionato --
  -- presente nella tabella --

implementation
  -- Questo modulo tiene conto di un quantitativo di denaro --
  -- suddividendolo in degli ideali 'mucchi' dello stesso --
  -- tipo --
composed of
end tabella_denaro;

```

```

module accumulocredito;
uses tabella_denaro;
    denaro;
    gestione_messaggi;
    gestione_turno
    resto
    statistica
exports
    enum risultato = {successo, fallimento};

    procedure annulla_operazione();
    -- questa procedura è richiamata dagli eventi di --
    -- tempo scaduto e annulla_operazione e si occupa di --
    -- terminare il processo, restituendo la bolletta --
    -- e i soldi introdotti --

    procedure conferma();
    -- questa procedura è richiamata dall'evento di conferma --
    -- ed è ascoltata solo se il denaro introdotto --
    -- ha superato quello richiesto --
    -- e si occupa di fornire il resto e processare la --
    -- bolletta e di stampare la ricevuta --

    procedure inserimento_denaro(tipo : [1..MAX_TIPO_DENARO]);
    -- incrementa la tabella_denaro che rappresenta il --
    -- denaro inserito e poi controlla se l'importo --
    -- è stato raggiunto o superato, e calcola la --
    -- disponibilità dell'eventuale resto. --

    procedure avvia_processo(dati_clienti : dati_bolletta);
    -- avvia il timer e dispone il gestore eventi in modo che --
    -- siano richiamate le procedure opportune (vedi sopra) --
    -- e si mette in attesa del denaro o si un altro evento --

implementation
    -- questo modulo è quello che si occupa dello svolgimento --
    -- dell'intera operazione di pagamento. --
    -- viene richiamato dal modulo che legge la bolletta --
    -- e cede il controllo al sistema operativo mettendosi --
    -- in attesa degli eventi che si possono verificare --
    -- se l'importo è stato raggiunto o superato calcola la --
    -- disponibilità dell'eventuale resto e aspetta una --
    -- conferma o un annullamento --
    -- se l'operazione va a buon fine, si emana il resto, si --
    -- processa la bolletta e si stampa la statistica --
    -- se l'operazione non va a buon fine, si rende il denaro --
    -- e si espelle la bolletta --

composed of
end accumulocredito;

```

```

module gestione_cassa_monete inherits tabella_denaro;
uses
exports
    function verifica_spazio_sufficiente() : Boolean;
    -- verifica che la cassetta monete possa accettare --
    -- le monete introdotte dall'utente --

    function disponibilita() : tabella_denaro;
    -- fornisce la quantità di monete presenti in cassa --

implementation
    -- Questo modulo simula il meccanismo di accumulo di monete --
    -- degli accettatori di lega, i quali possiedono dei tubi --
    -- nei quali le monete vengono depositate --
    -- e deve essere aggiornato ogni volta entra o --
    -- esce del denaro dalla cassa --
composed of
end gestione_cassa_monete;

```

## ***Gestione resto***

```

module resto;
uses tabella_denaro;
    gestione_cassa_monete;
    gestione_messaggi;
exports
    function calcola_resto(in totale : tabella_denaro; in
importo : real) : tabella_denaro;
        raise resto_non_disponibile;
    -- questa funzione calcola una combinazione possibile --
    -- delle monete disponibili in cassa, oppure solleva --
    -- una eccezione, nel caso in cui il resto non è --
    -- disponibile. --

implementation
    -- questo modulo si occupa di trattare il resto da --
    -- fornire all'utente: a partire dall'importo e --
    -- dal denaro introdotto calcola la differenza e --
    -- quindi tenta una combinazione delle monete che --
    -- ci sono nella cassa monete per ottenere il resto --
composed of
end resto;

```

## Schedulatore

```

module gestione_turno inherits coda(string),thread;
uses gestione_messaggi imports tempo_operazione;
    timer;
exports
    procedure prenotazione(in generalita : string);
    -- questa procedura mette in coda una persona --

    function nominativo_corrente() : string;
    -- questa funzione fa accedere al nominativo della --
    -- persona a cui tocca --

    procedure successivo();
    -- permette di accedere al nominativo successivo --
implementation
    -- il modulo è l'implementazione di una coda di stringhe --
composed of
end gestione_turno;

module gestione_errori inherits thead;
uses gestione_periferiche;
    timer;
exports
    function errore_di_sistema() : Boolean;
    -- questa funzione indica se si è verificato un errore --
    -- nel sistema --
implementation
    -- verifica ogni possibile causa di errore --
composed of
end gestione_errori;

```

## Statistica

```

module statistica;
uses timer;
exports
    procedure inizio_operazione();
    -- notifica che l'operazione è iniziata, così da --
    -- prelevare l'orario di inizio --

    procedure fine_operazione(fine : risultato);
    -- notifica che l'operazione è finita e specifica --
    -- se ha avuto successo e meno. La statistica --
    -- verrà memorizzata e viene stampata la ricevuta --

    procedure preleva_statistica(in richiesta :
base_statistica);
    -- mostra la statistica sul display e la invia attraverso --
    -- il cavo seriale --

    procedure stampa_statistica(in richiesta :
base_statistica);
    -- stampa la statistica su carta e la invia attraverso --
    -- il cavo seriale --

implementation
    -- questo modulo rappresenta l'interfaccia con il database --
    -- interno, in cui sono memorizzate le informazioni --
    -- che riguardano le operazioni effettuate (vedere la --
    -- sezione Database) --
composed of
end statistica;

```

## ***Gestione messaggi***

```

module gestione_messaggi;
uses immagine,suono;
      gestione_periferiche;
exports
  procedure tempo_operazione(in tempo : ora);
  -- questa procedura (aggiornata costantemente dal timer) --
  -- visualizza la durata corrente dell'operazione --

  procedure visualizza_messaggio(in codice : integer);
  -- ad ogni stato della macchina è associato un codice --
  -- al quale corrispondono una stringa, una immagine --
  -- bitmap e un suono (vedere Interfaccia per i dettagli) --

  procedure controlli_sistema();
  -- questa procedura (aggiornata costantemente dal timer) --
  -- effettua dei controlli sulla carta e sull'inchiostro --
  -- ed eventualmente si occupa di visualizzare la --
  -- mancanza di uno dei due. --
implementation
  -- Questo modulo viene richiamato da tutti gli altri --
  -- e si occupa di interagire con la memoria video --
  -- per visualizzare testo e immagini nelle posizioni --
  -- previste dalle varie tabelle --
composed of
end gestione_messaggi;

```

## Varie

```
module gestione_periferiche;
uses
exports
    enum stato_generatore = { attivo, disattivo };
    enum stato_carta = { disponibile, non_disponibile };
    enum stato_inchiostro = { pieno, vuoto };

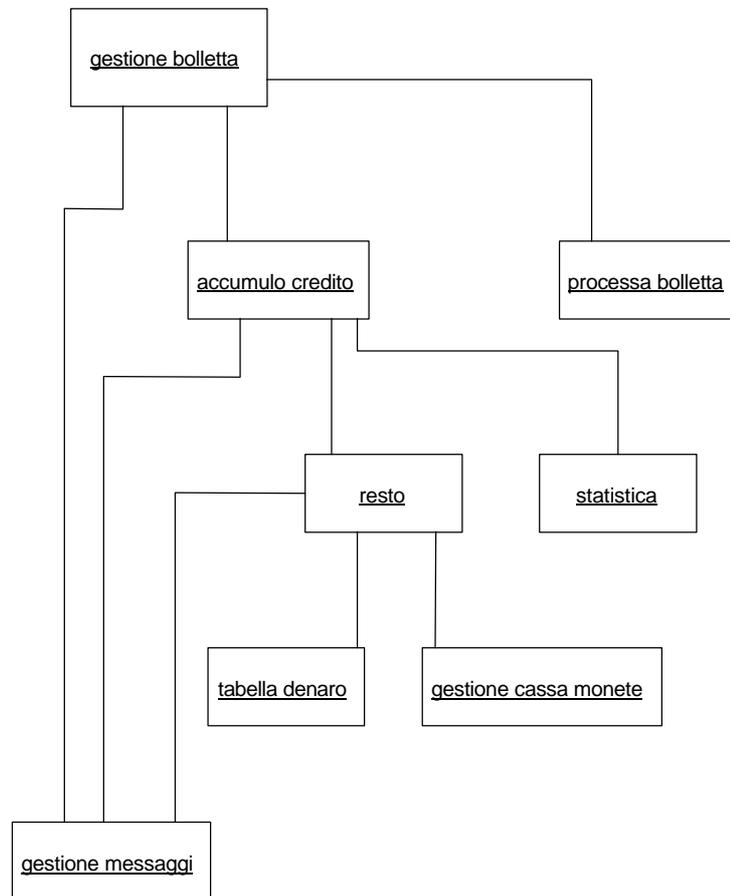
    function verifica_stato_generatore() : stato_generatore;
    -- controlla che il generatore di energia supplementare --
    -- non sia attivo --

    function verifica_stato_carta() : stato_carta;
    -- controlla che la carta non sia finita --

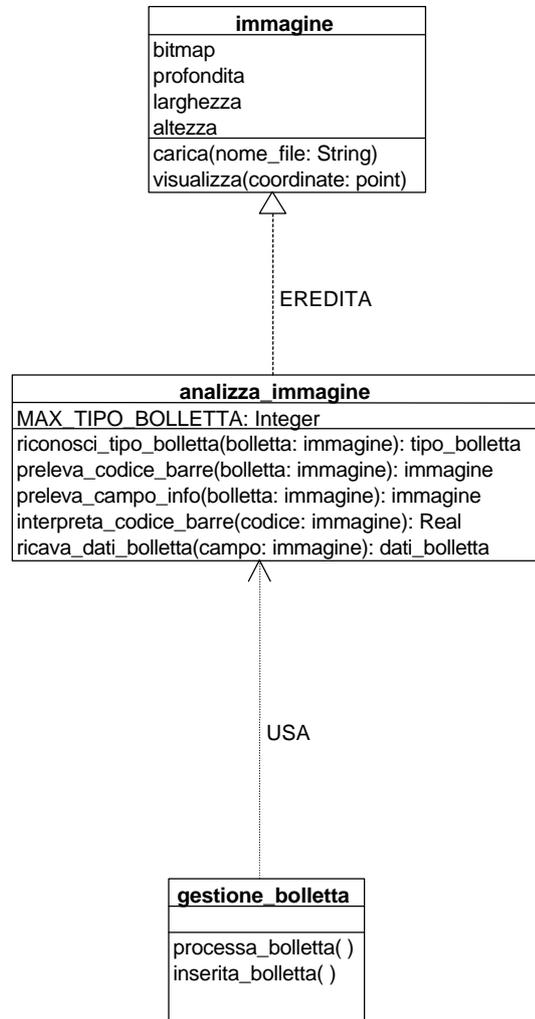
    function verifica_stato_inchiostro() : stato_inchiostro;
    -- controlla che l'inchiostro sia ancora disponibile --

    function verifica_spazioHD_sufficiente() : Boolean;
    -- lo spazio su HD deve essere garantito affinché --
    -- l'operazione di scrittura delle statistiche --
    -- sia una operazione valida --

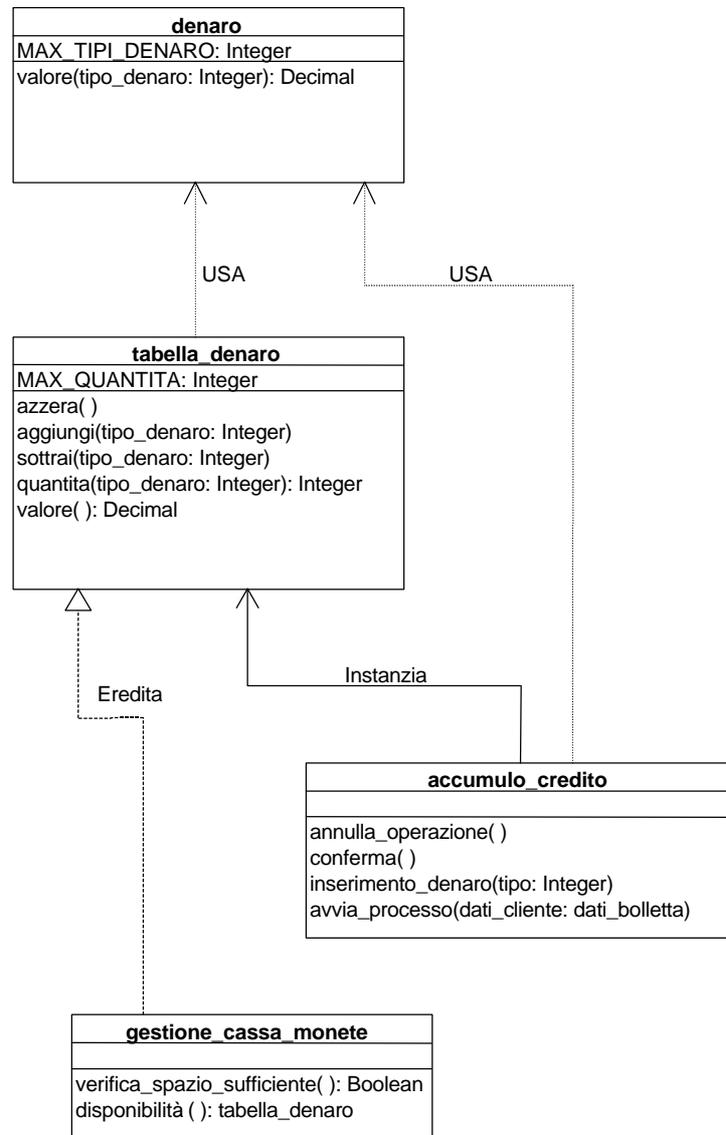
implementation
    -- per eseguire tali controlli, il modulo si deve --
    -- interfacciare direttamente all'hardware o al S.O. --
composed of
end gestione_periferiche;
```

**Grafici allegati alla descrizione****Grafico relazione USA:**

## Class diagram: Gestione bolletta



## Class diagram: gestione denaro



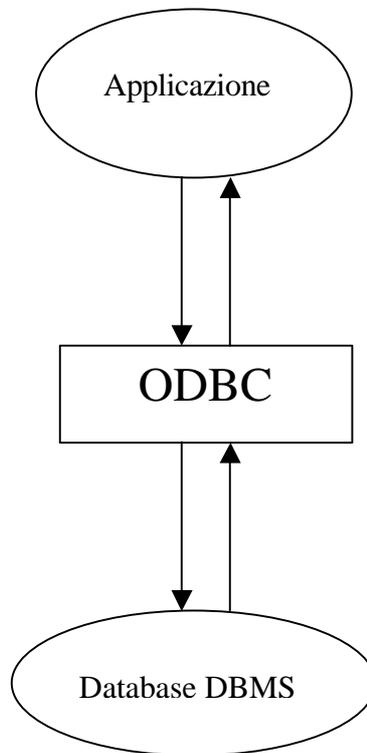
## Appendice A: database statistica

Il modello di database usato in questo progetto sfrutta la tecnologia DBMS (Database Management System). Il tipo preciso di database può, grazie a questa scelta, essere deciso a seconda del sistema operativo usato, delle esperienze personali dei programmatori, e della disponibilità.

Il database (di qualunque tipo sia) viene infatti gestito mediante interfaccia ODBC (Open Database Connectivity) in modo che il programma risulterà indipendente dal sistema usato. Il sistema di database può anche essere cambiato in seguito senza intervenire più nel codice.

Le interrogazioni al database – sia quelle interne che quelle effettuate mediante connessione remota - avvengono (secondo lo standard ODBC) in linguaggio SQL (Structured Query Language) ovvero il linguaggio strutturato usato per effettuare richieste ad una base dati.

Nota: nel caso in cui l'implementazione del sistema dovesse essere svolta in linguaggio Java, si può usare in modo perfettamente equivalente la tecnologia JDBC nativa del linguaggio, che offre una semplice interfaccia al motore ODBC.



## Appendice B: Interfaccia utente

Sul video della macchina saranno visibili sia del testo sia dei simboli in formato grafico. Ogni zona dello schermo è dedicata ad una funzionalità ben precisa. Le informazioni sono suddivise in classi e ogni classe ha a disposizione la sua zona nello schermo. In questa maniera più informazioni complementari possono essere visualizzate nello stesso tempo.

<b>ICONE</b>	
DESCRIZIONE TESTUALE DELLE ICONE	
<b>MESSAGGIO TESTUALE</b>	<b>IMMAGINE O ANIMAZIONE</b>
<b>DATA E ORA CORRENTE</b>	<b>DURATA OPERAZIONE</b>

- icone: nella parte superiore dello schermo vengono visualizzate delle immagini che devono rappresentare la disponibilità o meno di alcune funzionalità del sistema (ad esempio, macchina funzionante, ricevuta non disponibile, cassa vuota, cassa piena ...),
- descrizione testuale delle icone: sotto ad ogni icona, viene riportato un messaggio testuale che spiega in modo sintetico la funzionalità, in modo che non ci possano essere dubbi o ambiguità,
- messaggio testuale: rappresenta un testo a più righe, con i caratteri molto grandi, che comunica all'utente qual'è l'operazione richiesta,
- immagine o animazione: accanto al testo appare una grande immagine che deve avere un impatto visivo notevole sull'utente, per comunicargli in modo rapido l'operazione richiesta,
- data, ora e durata operazione: sono dei valori costantemente aggiornati, e rappresentano delle informazioni temporali.

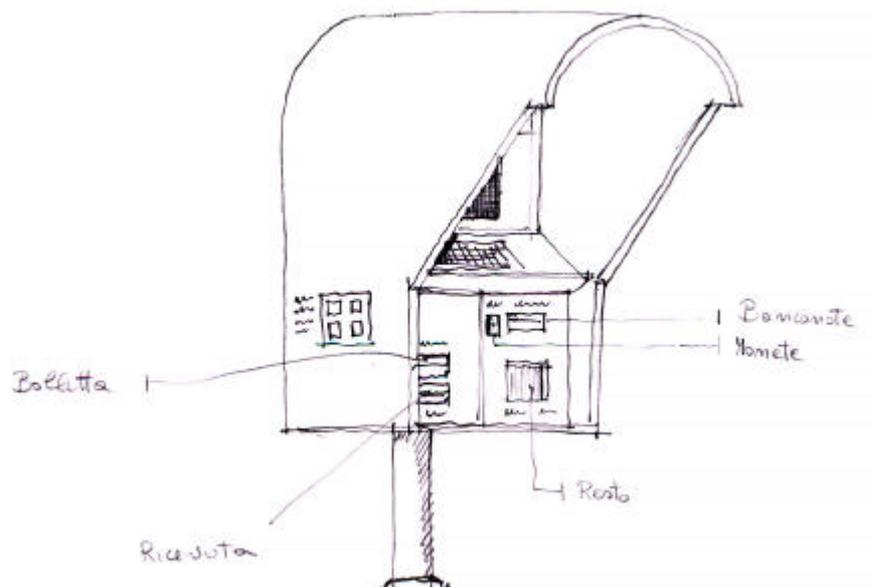
## Appendice C: design e funzionalità

Lo sportello per il pagamento delle bollette, Aurora, è finalizzato ad una utilizzazione, da parte degli utenti, comoda e funzionale. L'altezza dei vari componenti, l'inclinazione della testiera, sono progettati in modo da essere utilizzati anche dai disabili seduti su carrozzella.

Sono state individuate diverse zone in funzione delle operazioni che devono essere compiute. La parete frontale si suddivide in tre sotto-zone disposte su piani differenziati per facilitarne l'individuazione delle diverse operazioni:

- 1) quella superiore, con un'inclinazione di 15° verso l'osservatore per evitare i riflessi sul video che vi è collocato. E' stata scelta questa posizione perché in tal modo il video si trova all'altezza degli occhi dell'osservatore. Inoltre non è difficoltoso da vedere per un utente costretto su carrozzella;
- 2) quella centrale, inclinata, con una pulsantiera per la conferma o l'annullamento dell'operazione;
- 3) quella inferiore verticale, operativa, suddivisa nettamente con una linea scura ad incasso, in due parti:
  - a) a destra sono raggruppate le fessure per l'immissione di monete e banconote, e la cassettona per la restituzione del resto o delle monete introdotte.
  - b) a sinistra sono disposte la fessura per l'immissione della bolletta per la lettura delle informazioni e quella per la fuoriuscita della ricevuta.

Tutti i punti operativi sono contrassegnati con delle didascalie testuali e grafiche.



Lo sportello è dotato di una copertura a mensola, che riduce l'effetto di riflessione sul video.

Lateralmmente è collocata una testiera alfabetica, in cui è possibile prenotare il turno.



## Riferimenti

### ***Bibliografia***

A.Fuggetta, C. Ghezzi, S. Morasca, A.Morzenti, M. Pezzè, “Ingegneria del Software”, Mondadori Informatica.

Roger S. Pressman, “Principi di ingegneria del software”, Informatica professionale.

### ***Siti utili***

DATASIS GROUP SRL: <http://space.tin.it/economia/csavolde/DATASIS.html> sito in cui si possono trovare librerie per il riconoscimento dei codici a barre e del testo stampato e scritto a mano.

BAR CODE SITE : [http://www.barcode.it/glossary/bar\\_code\\_site.htm](http://www.barcode.it/glossary/bar_code_site.htm) sito in cui trovare informazioni tecniche sui codici a barre e prodotti hardware per la lettura e l’interpretazione.

MANUALE DEL CODICE A BARRE:  
<http://www.acse.com/italian/manuale/classificazione.htm>, in cui trovare informazioni su come vengono interpretati i vari tipi di codice a barre esistenti.

EAN: <http://www.ean.be/index.html>, sito in cui si trova uno degli standard più diffusi per il codice a barre dei prodotti.

Altri siti sui codici a barre: [www.barcodesystems.net](http://www.barcodesystems.net), [www.adams1.com](http://www.adams1.com), [www.coder.it](http://www.coder.it);

SISTEMI PAGAMENTO: [www.coges.it](http://www.coges.it), sito in cui trovare una ampia scelta di prodotti come riconoscitori di lega, gettoniere rendiresto, lettori banconote e altro ancora.

ANALISI-DISEGNO: [www.analisi-disegno.com](http://www.analisi-disegno.com), sito da cui prelevare la modulistica per l’analisi dei requisiti.