



**CONSIGLIO NAZIONALE
DELLE RICERCHE**

Antonio Camurri °, Goffredo Haus ^,
Alberto Stiglitz ^, Renato Zaccaria °

**LRC C4 MUSIC: ARCHITETTURA,
SPECIFICHE FUNZIONALI DI
ALTO LIVELLO E STANDARD DI
DOCUMENTAZIONE DELLA
STAZIONE DI LAVORO
MUSICALE INTELLIGENTE**

N. 7/09

11/89

**PROGETTO FINALIZZATO
SISTEMI INFORMATICI E CALCOLO PARALLELO**

SOTTOPROGETTO 7

SISTEMI DI SUPPORTO AL LAVORO INTELLETTUALE

**PROGETTO FINALIZZATO
SISTEMI INFORMATICI E CALCOLO PARALLELO**

SOTTOPROGETTO 7
Sistemi di Supporto al Lavoro Intellettuale
Coordinatore Cesare Maioli

Antonio Camurri [°], Goffredo Haus [^],
Alberto Stiglitz [^], Renato Zaccaria [°]

**LRC C4 MUSIC: ARCHITETTURA,
SPECIFICHE FUNZIONALI DI
ALTO LIVELLO E STANDARD DI
DOCUMENTAZIONE DELLA
STAZIONE DI LAVORO
MUSICALE INTELLIGENTE**

N. 7/09

11/89

Rapporto Tecnico

[°] DIST - Dipartimento di Informatica Sistemistica e Telematica
Università di Genova.

[^] LIM - Laboratorio di Informatica Musicale
Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano.

"STAZIONE DI LAVORO MUSICALE INTELLIGENTE"

Rapporto R2:

"LRC C4 MUSIC: architettura, specifiche funzionali di alto livello e standard di documentazione della STAZIONE DI LAVORO MUSICALE INTELLIGENTE."

Parole chiave

computer music, audio digital signal processing, artificial intelligence, functional specifications, documentation standards, programming & microprogramming languages .

INDICE:

Premessa generale.

Parte I.

- P0. Introduzione.**
- P1. Architettura della SLMI.**
- P2. Discussione schematica delle funzionalità della SLMI.**
- P3. Librerie disponibili nella SLMI.**
- P4. Descrizione sintetica dei moduli della SLMI.**
- P5. Standard utilizzati nell'implementazione.**
- P6. Configurazione hardware.**

Parte II.

- P7. Introduzione.**
 - P7.1 Generalità.**
 - P7.2 Struttura della Parte II.**
 - P7.3 Nota preliminare.**
- P8. Rapporto tecnico R3.**
 - P8.1 Strumenti formali per la descrizione delle specifiche funzionali dei moduli dei livelli 1 e 2.**
 - P8.2 Strumenti formali per la descrizione delle specifiche funzionali dei moduli del livello 3.**
- P9. Documentazione operativa.**
 - P9.1 "Manuale d'uso".**
 - P9.2 "Note tecniche".**
- P10. Note di stile.**
- P11. Riferimenti bibliografici.**

Appendici.

- A1. Indice schematico del "Manuale d'uso".**
- A2. Indice schematico di "Note tecniche".**
- A3. Esempio di indice per un "Manuale di riferimento".**

Premessa generale.

La **Parte I** di questo rapporto fornisce una descrizione ad alto livello dell'architettura complessiva della **SLMI** (Stazione di Lavoro Musicale Intelligente) e dei singoli moduli software di cui è costituita.

La **Parte II** è una guida schematica ed esauriente per la stesura dei rapporti tecnici e dei documenti operativi relativi ai moduli della **SLMI**.

Parte I.

P0. Introduzione.

Nei successivi paragrafi viene fornita una descrizione ad alto livello dell'architettura complessiva della **SLMI** e dei singoli moduli software di cui è costituita.

In particolare, sono descritte:

- * l'architettura software complessiva della **SLMI**;
- * l'architettura hardware complessiva della **SLMI**;
- * le funzionalità di alto livello che possono configurarsi come ambiti operativi specifici per l'utente finale, sia in relazione ai moduli propri della **SLMI** che in relazione a moduli applicativi reperibili in commercio;
- * le funzionalità di alto livello dei singoli moduli software della **SLMI**;
- * le librerie di modelli, dati e programmi sviluppate e di corredo per la **SLMI**;
- * gli standard di codifica e di ambiente di sviluppo software adottati per il progetto **SLMI**.

P1. Architettura della SLMI.

La **SLMI** si configura come un insieme di moduli hardware e software organizzati in modo gerarchico.

In prima approssimazione, il sistema è strutturato a cinque livelli:

- 0. **Livello hardware:**
 - Sistema di elaborazione general purpose.
 - Moduli per la elaborazione numerica di suoni.
 - Dispositivi per la registrazione e la riproduzione del suono.
 - Dispositivi di controllo e distribuzione di segnali (patchbay MIDI e di segnali audio, mixer).

- 1. **Livello S/W 1:**
 - Moduli a basso livello per le attività di analisi-sintesi del suono, la costruzione e l'elaborazione di campioni e modelli timbrici, la produzione di master audio numerici e il restauro di registrazioni audio.

- 2. **Livello S/W 2:**
 - Moduli ad alto livello per le attività di composizione, analisi/sintesi di testi musicali e di strutture di testi musicali, performance musicale e audiovisuale, orchestrazione, DTP musicale.

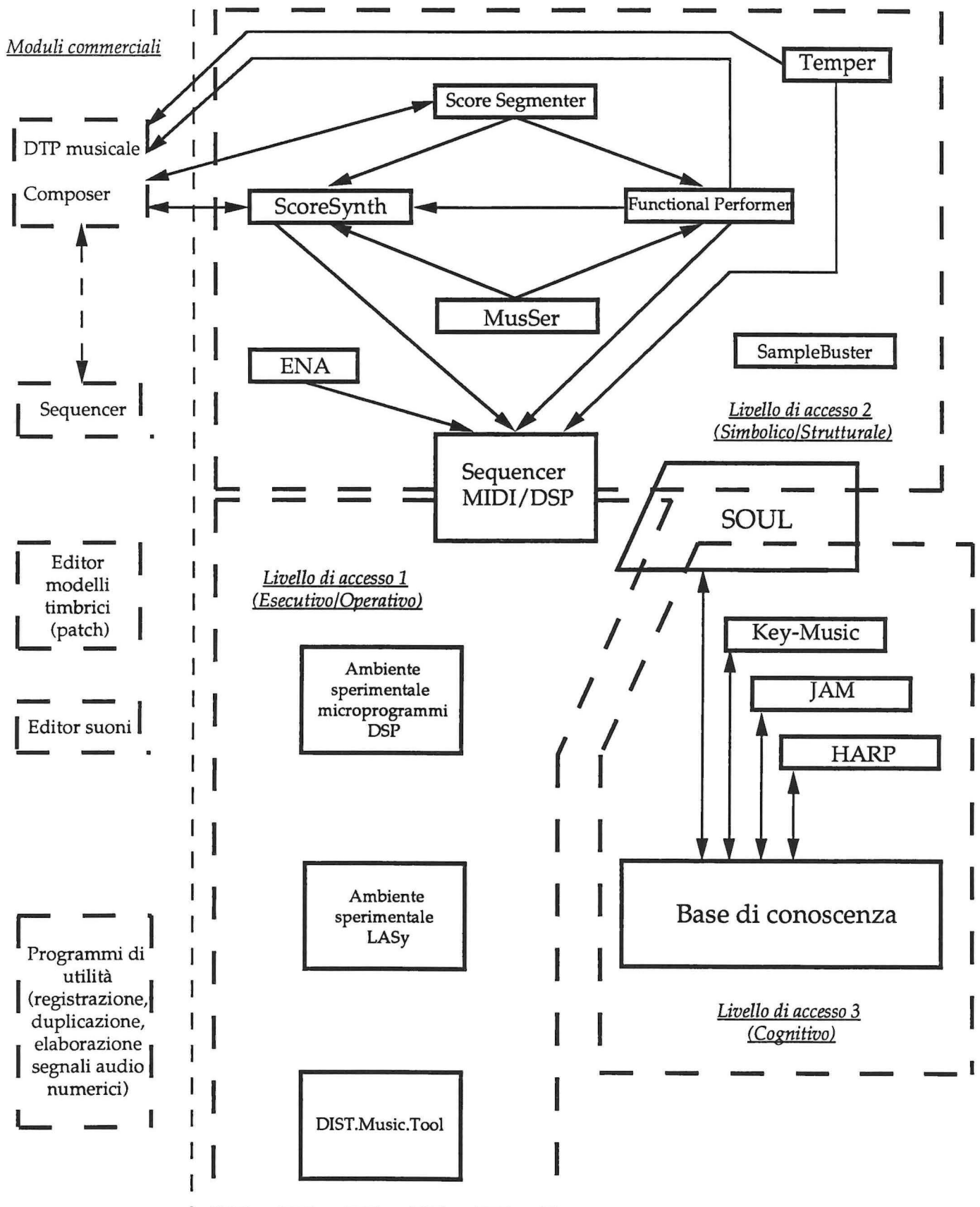
- 3. **Livello S/W 3:**
 - Moduli ad alto livello per la rappresentazione di conoscenza musicale.

- 4. **Interfaccia utente:**
 - Interfaccia utente ad alto livello di tipo ipertestuale.

Nel diagramma funzionale della pagina seguente sono rappresentati i livelli S/W 1, 2 e 3, mentre nel paragrafo P6 verrà descritto il livello hardware.

Il livello "Interfaccia utente" sarà costituito da un insieme di moduli interscambiabili che opereranno su un insieme di stack attivi di HyperCard, per le seguenti attività: gestione delle richieste utente; presentazione di risultati/diagnostiche in formati standard; presentazione di oggetti/entità ad un livello di astrazione adatto ad un utente che intende operare in un particolare sottoambiente.

PFI - 2
Architettura
Stazione di Lavoro Musicale Intelligente
Diagramma funzionale



P2. Discussione schematica delle funzionalità della SLMI.

In questo paragrafo sono brevemente discusse le funzionalità di alto livello della SLMI che possono configurarsi come ambiti operativi specifici per l'utente finale, sia in relazione ai moduli propri della SLMI che in relazione a moduli applicativi reperibili in commercio.

Rappresentazione di conoscenza musicale

Uno degli obiettivi primari della SLMI è quello di assistere il compositore nelle fasi altamente creative del processo compositivo. Ciò include la definizione di piani e architetture per una composizione e la codifica del materiale musicale in una partitura operativa. Allo scopo di facilitare l'interazione col sistema, dovrà essere supportata una definizione più ampia possibile di "materiale musicale", mediante diverse rappresentazioni della conoscenza musicale. I sistemi *Key-Music* ed *HARP* (Hybrid Action Representation and Planning) hanno l'obiettivo di fornire la soluzione per alcuni di questi problemi. In particolare, *Key-Music* è un sistema per la rappresentazione di conoscenza musicale basato su reti di Petri e reti semantiche di frame ad eredità multipla; mentre *HARP* consiste in un sistema ibrido, ovvero integra aspetti di tipo simbolico ed analogico. Ad essi verrà affiancato il modulo *JAM*, un sistema prototipale per la rappresentazione di conoscenza musicale.

Un altro obiettivo primario della SLMI è quello di fornire ai ricercatori strumenti di indagine conoscitiva su vari aspetti dell'ambito musicale. Tra questi ultimi, uno dei più rilevanti riguarda le problematiche dell'ascolto. A tale proposito verrà realizzato il modulo *SOUL* (Self-Organizing Universal Listener), implementazione di un modello per l'*ascolto intelligente*, ovvero per il riconoscimento di una sorgente sonora (tipicamente uno strumento musicale) a partire da un segnale complesso (costituito ad esempio da più strumenti). Quest'ultimo modello è basato su reti neurali, in particolare sul modello di Kohonen.

Composizione

L'attività di composizione musicale può anche essere realizzata combinando la scrittura tradizionale resa possibile dai programmi *composer* commerciali insieme con la scrittura non tradizionale basata su reti di Petri, oggetti musicali e algoritmi con il modulo *ScoreSynth*. Per la composizione di

opere musicali di natura seriale verrà realizzato *MusSer*, un modulo specifico per la generazione di serie musicali equilibrate con un qualsivoglia temperamento equabile a n gradi (con n pari). Gli oggetti musicali argomento degli algoritmi definiti nei modelli *ScoreSynth* possono essere derivati tanto da moduli applicativi commerciali via Standard MIDI File che dal modulo *MusSer* o anche dal modulo *ScoreSegmenter* che decompone in oggetti musicali intere partiture precedentemente codificate.

Analisi/sintesi testi musicali

La trasformazione di testi musicali preesistenti è una peculiare funzionalità della SLMi resa possibile dall'uso combinato dei moduli *ScoreSegmenter* (per l'analisi della partitura e la sua decomposizione in oggetti musicali), *ScoreSynth* (per la sintesi di trasformazioni della partitura mediante modelli di reti di Petri) e *Functional Performer* (per la sintesi di trasformazioni della partitura mediante operatori in tempo reale).

Analisi strutture testi musicali

Il modulo *ScoreSegmenter*, come già accennato qui sopra, consente di identificare gli oggetti musicali di cui è composta una partitura con una particolare affidabilità per i brani musicali in forma di fuga o di sonata. E' quindi uno strumento essenziale e fortemente innovativo di supporto al musicologo e in generale allo studioso dei testi musicali.

Performance musicale

Il modulo *Functional Performer* permette di eseguire brani musicali attivando da tastiera ASCII funzioni musicali che vengono applicate ad oggetti musicali predisposti precedentemente o tramite moduli SLMi o tramite composer o sequencer commerciali in formato Standard MIDI File. Il modulo *ENA* (Experimental Neural Accompanist), basato su reti neurali di tipo back-propagation, genera in tempo reale un accompagnamento musicale automatico durante l'esecuzione di una parte solistica.

Performance audiovisuale

Il modulo *TEMPER* permette di eseguire automaticamente processi audiovisivi costituiti da animazioni grafiche nello stile del pittore M. C. Escher e

processi MIDI ottenuti per traslitterazione delle animazioni grafiche (intepretate come piani melodici).

Analisi-sintesi suono

E' un ambiente sperimentale in cui è disponibile un modulo (microprogramma per il DSP Motorola 56001) per l'analisi-sintesi con la trasformata wavelet. In combinazione con programmi come Digidesign SoftSynth e Digidesign SoundDesigner permette di confrontare e potenziare le caratteristiche della analisi-sintesi additiva per armoniche con quelle della analisi-sintesi mediante trasformata wavelet.

Editing di campioni sonori

Il modulo DIST.Music.Tool della SLMI e il programma Digidesign SoundDesigner consentono di effettuare l'editing di campioni sonori direttamente all'elaboratore e di effettuare operazioni di import/export dalle periferiche MIDI a campionamento.

Preparazione modelli timbrici

E' un ambiente sperimentale in cui è disponibile un insieme di moduli (microprogrammi per il DSP Motorola 56001) per la sintesi del suono secondo vari modelli di sintesi: funzioni di due variabili, waveshaping, automi cellulari.

Orchestrazione

L'attività di orchestrazione è una delle più complesse per il musicista; richiede un ciclo ipotesi-di-orchestrazione/sperimentazione-sonora/editing-dell'assetto-orchestrare che si ripete tante volte quante sono necessarie per il raggiungimento dell'obiettivo di tessitura timbrica ricercata; per questo è stato progettato un ambiente in cui siano operativi concorrentemente due tipi di moduli software: uno di supporto alla ricerca dei suoni nella banca di timbri disponibili (il modulo *SampleBuster* o gli ambienti per la sperimentazione timbrica nella SLMI) e l'altro per l'esecuzione dei brani nelle varie ipotesi di tessitura timbrica (un qualunque sequencer MIDI commerciale o il modulo *Sequencer MIDI-DSP* nella SLMI, che consente di controllare

contemporaneamente unità di sintesi MIDI e DSP microprogrammati in tempo reale).

DTP musicale

Integrando i moduli della SLMI capaci di generare partiture (*ScoreSynth*, *Functional Performer*) con gli strumenti per il Desk Top Publishing musicale commerciali (ad es. il programma Finale della Coda Music Software), è possibile ottenere le partiture generate in notazione tradizionale e ai massimi livelli di qualità (fotocompositrici che accettano codici PostScript; es. Linotronics 100 P).

Produzione master audio numerici

I programmi sequencer commerciali, il modulo *Sequencer MIDI-DSP* e alcuni programmi e dispositivi di utilità (Digidesign DAT I/O, Digidesign SoundTools) consentono di registrare, prima a livello MIDI e poi come audio digitale su disco rigido SCSI, su nastro DAT e come audio analogico su nastri open reel, il master delle produzioni musicali.

Restauro registrazioni audio

Il modulo DIST.Music.Tool della SLMI e il programma Digidesign SoundDesigner consentono di effettuare l'editing di registrazioni audio per il restauro di vecchi supporti; anche in questo caso, alcuni programmi e dispositivi di utilità (Digidesign DAT I/O, Digidesign SoundTools) consentono di spostare da un supporto all'altro e di convertire nei formati richiesti l'informazione audio digitale tra nastri DAT, dischi rigidi SCSI, campionatori MIDI, campionatori SCSI, nastri analogici open reel.

P3. Librerie disponibili nella SLMI.

I moduli della SLMI hanno accesso in lettura e scrittura a un insieme di librerie che costituiscono uno degli elementi costitutivi essenziali della SLMI stessa; in questo paragrafo vengono succintamente descritte tali librerie; nel Rapporto tecnico R3 ("Specifiche funzionali dei moduli della STAZIONE DI LAVORO MUSICALE INTELLIGENTE") saranno dettagliati i vincoli relativi agli accessi per ogni singolo modulo.

Libreria di reti di Petri "macro"

Il modulo *ScoreSynth* ha accesso ad una libreria di reti di Petri "macro" ovvero un insieme di costrutti di reti di Petri che sono descritti in forma parametrica e che possono essere invocati mediante parametri formali per valore. Questa libreria è stata progettata per rendere disponibile quell'insieme di costrutti che ricorrono sovente nello sviluppo di modelli di partiture con reti di Petri e rendere perciò più efficiente la fase di editing dei modelli formali.

Libreria di routine di lettura/scrittura di Standard MIDI File v 1.0

Questa libreria (costituita da una cinquantina di routine) costituisce un vero e proprio modulo della SLMi e come tale sarà descritta in R3 e più succintamente nel prossimo paragrafo. E' stata progettata per rendere disponibile in forma standard a tutti i ricercatori del presente progetto il principale strumento di comunicazione con i moduli applicativi reperibili in commercio e tra i moduli della SLMi stessi. Tanto le applicazioni commerciali quanto i moduli della SLMi hanno funzionalità di import/export di file a livello di partitura MIDI come Standard MIDI File v 1.0.

Libreria di patch timbriche per microprogrammi del DSP Motorola 56001

Il modulo *Sequencer MIDI-DSP* consente di variare parametricamente il modello timbrico corrente mediante patch timbriche; ogni microprogramma sviluppato per tale modulo, come anche i microprogrammi per l'*Ambiente sperimentale per microprogrammi DSP*, avranno a disposizione un corredo di patch timbriche (analogamente a quanto accade per i preset dei sintetizzatori MIDI) organizzati in una libreria accessibile dai due moduli citati. L'utente di tali moduli potrà ovviamente sviluppare le sue specifiche patch timbriche e aggiungerle alla libreria sviluppata nell'ambito del presente progetto.

Banca timbrica di campioni sonori

Il modulo *SampleBuster* è basato su una banca timbrica di campioni sonori costituita da circa 7.000 elementi delle seguenti famiglie timbriche originali:

- f1) MUMS (McGill University Master Samples)
- f2) Sound Examples (Current Directions in Computer Music Research, M.I.T.)
- f3) AKAI S900 Samples Library
- f4) AKAI S950 Samples Library
- f5) AKAI S1000 Samples Library
- f6) libreria di suoni campionati del L.I.M.

le famiglie timbriche f1 ed f2 sono memorizzate su CD-DA (Compact Disc Digital Audio); le famiglie f3, f4, f5 ed f6 sono memorizzate su dischi magnetici.

P4. Descrizione sintetica dei moduli della SLMI.

In questo paragrafo vengono descritti sinteticamente tutti i moduli sperimentali che fanno parte dell'architettura della SLMI. Per una descrizione più dettagliata e formale delle funzionalità dei singoli moduli, si rimanda al Rapporto tecnico R3 ("Specifiche funzionali dei moduli della STAZIONE DI LAVORO MUSICALE INTELLIGENTE").

ScoreSynth

Il modulo **ScoreSynth** consisterà in un'applicazione, sviluppata in ambiente MPW su computer Macintosh, che permetterà di sintetizzare partiture musicali mediante l'esecuzione di modelli formali sviluppati con reti di Petri.

I modelli che verranno utilizzati consistono in un particolare adattamento delle reti di Petri al contesto musicale, e in un'algebra musicale. Gli *oggetti musicali* sono associati a *posti* e saranno codificati in uno dei seguenti tre formati: un codice musicale alfanumerico, Standard MIDI File v.1.0, una codifica numerica interna dei dati MIDI. Le funzioni musicali sono descritte per mezzo di espressioni associate a *transizioni* e saranno codificate mediante operatori dell'algebra musicale appositamente definita. Gli operatori potranno essere applicati ai parametri musicali *altezza, durata, dinamica* (intensità); inoltre potranno modificare l'*ordinamento* degli oggetti musicali. L'approccio seguito permette di descrivere gli oggetti musicali e le loro trasformazioni all'interno delle composizioni in modo svincolato dal livello simbolico di rappresentazione che è tipico della comune notazione musicale. In questo modo la musica appare come un modello multistratificato nel quale il musicista sceglie il numero e le caratteristiche dei diversi livelli di rappresentazione.

MusSer

Il modulo **MusSer** consisterà in un'applicazione, sviluppata in ambiente MPW su computer Macintosh, che permetterà di generare pattern seriali equilibrati in ambito di scale temperate di cardinalità pari.

Le serie equilibrate sono una delle strutture base utilizzate nella composizione di musica seriale. Utilizzando MusSer, un compositore sarà in grado di valersi del computer nella fase di sviluppo e ricerca dei pattern basilari per una composizione seriale. Infatti avrà a disposizione archivi permanenti, residenti su memoria di massa, contenenti tutte le sequenze numeriche corrispondenti alle serie dodecafoniche di base, e gli strumenti per la ricerca selettiva delle serie con le caratteristiche desiderate, e per la loro elaborazione. In questo modo potrà creare archivi contenenti pattern seriali "personalizzati" da poter utilizzare con altri moduli di ausilio alla composizione musicale appartenenti alla SLMI.

Sequencer MIDI/DSP

Il modulo **Sequencer MIDI/DSP** consisterà in un'applicazione, sviluppata in ambiente MPW su computer Macintosh, di tipo *sequencer* (registratore e processore di sequenze di dati musicali) per il controllo simultaneo e sincrono di unità MIDI e di schede DSP e per il mixing a livello MIDI di sequenze musicali.

L'applicazione sarà composta da due sezioni: un *sequencer MIDI* che unirà alle consuete funzioni di registrazione, riproduzione ed elaborazione di sequenze MIDI anche la possibilità di controllare, tramite un apposito driver, la scheda Digidesign Sound Accelerator (basata sul DSP Motorola 56001) collocata internamente al Macintosh, vedendola come una normale unità di sintesi e/o campionamento esterna controllabile via MIDI; un *mixer* per il bilanciamento dei livelli di intensità di differenti e contemporanee sequenze di dati musicali (numero massimo: 32) che opererà direttamente sulle sequenze di dati MIDI, agendo sul parametro MIDI standard *key-velocity*, ed eliminando quindi la necessità di utilizzare per l'attività di mixing un apposito dispositivo hardware esterno di tipo convenzionale.

Microprogrammi per Sequencer MIDI-DSP e Sound Accelerator Card

Il modulo **Microprogrammi per Sequencer MIDI-DSP e Sound Accelerator Card** consisterà in una libreria di microprogrammi, sviluppata in ambiente MPW

su computer Macintosh con scheda Digidesign Sound Accelerator (basata su DSP Motorola 56001), per la sintesi del suono mediante DSP dedicato.

I microprogrammi implementeranno le seguenti tecniche di sintesi: TVF (funzioni a due variabili), Waveshaping, Wavelet e Automi Cellulari; e saranno utilizzabili in ambiente MPW (ad esempio verranno usati, nell'ambito della SLMI, per lo sviluppo dell'*Ambiente sperimentale per microprogrammi DSP per Sound Accelerator Card*).

Functional Performer

Il modulo **Functional Performer** consisterà in un'applicazione, sviluppata in ambiente MPW su computer Macintosh, che permetterà di realizzare performance funzionali in tempo reale di pattern musicali e loro trasformazioni.

L'applicazione accetterà in input frammenti musicali (melodici polifonici) codificati in due formati: Standard MIDI File v.1.0 generati da altre applicazioni della SLMI o commerciali; formato di rappresentazione interna. I frammenti potranno essere organizzati in appositi archivi caricabili all'inizio di una performance. L'utente-esecutore avrà quindi a disposizione un certo numero di operatori, attivabili agendo col mouse o con la tastiera ASCII del Macintosh, per la trasformazione e l'esecuzione in tempo reale dei frammenti musicali originari e/o derivati. Gli operatori implementeranno una serie di trasformazioni tipiche della prassi compositiva e improvvisativa moderna (trasposizioni, inversioni speculari, retrogradazioni, ecc.), che agiranno sui parametri altezza e durata degli elementi dei frammenti musicali. L'esecuzione provocherà la generazione di sequenze di dati MIDI per il controllo di unità esterne di sintesi e campionamento del suono, collegate al Macintosh mediante interfaccia MIDI. Le sequenze potranno anche essere registrate, durante l'esecuzione, in archivi in formato Standard MIDI File v.1.0. Durante l'esecuzione sarà visibile sullo schermo del Macintosh una partitura grafica a scansione temporale, con la visualizzazione dei frammenti musicali attivi, distribuiti in varie fasce orizzontali corrispondenti ai canali MIDI utilizzati.

TEMPER

Il modulo **TEMPER** (TEssellating Music PERformer) consisterà in un'applicazione, sviluppata in ambiente Macintosh, che permetterà di generare performance audiovisuali basandosi su tassellazioni del piano nello stile dell'artista grafico olandese M.C.Escher.

L'applicazione sarà composta da due sezioni: un *editor* che consentirà di costruire, memorizzare e modificare sequenze di tassellazioni (una tassellazione è una partizione del piano in insiemi connessi di punti; in altri termini, il piano viene suddiviso in più figure senza "vuoti", ovvero con i contorni combacianti) appartenenti a una classe particolare (gruppo spaziale bidimensionale P1); ed un *performer* che eseguirà trasformazioni grafiche delle tassellazioni indicate dall'utente, accompagnate eventualmente dalla corrispondente esecuzione musicale. Per quanto riguarda quest'ultima, saranno generate sequenze di dati MIDI utilizzabili per controllare direttamente unità esterne per la sintesi e/o il campionamento del suono, o per essere registrate da un programma di tipo "sequencer". La generazione delle sequenze di dati MIDI avverrà in tempo reale e produrrà eventi musicali sincronizzati con le trasformazioni grafiche, secondo la forma dei contorni delle figure e alcuni parametri specificati dall'utente (velocità di esecuzione, scala musicale, ecc.).

ScoreSegmenter

Il modulo **ScoreSegmenter** consisterà in un'applicazione, sviluppata in ambiente Macintosh, che permetterà di segmentare partiture date in oggetti musicali.

La segmentazione di brani musicali è da considerarsi quale primo passo per una futura loro strumentazione automatica proposta dall'elaboratore. Si tratta cioè della ricerca dei vari oggetti musicali di cui è composto il brano, dove, in questo contesto, per oggetti si intendono quei fraseggi o pensieri musicali che l'autore ha espresso e che ha ripreso e trasformato, secondo i vari canoni musicali, in relazione al periodo storico e alla forma compositiva. La realizzazione di **ScoreSegmenter** rappresenterà il primo tentativo di costruire uno strumento informatico che renda possibile l'accostarsi ai testi musicali non come mera sequenza di note, cioè di codifica di eventi sonori, come la pratica strumentale richiede, ma considerandoli maggiormente come impianti strutturali di una particolare forma espressiva (quella del compositore) che si reggono su elementi base anche molto complessi (materiale tematico e armonico di impianto). Gli algoritmi per la ricerca delle occorrenze degli oggetti musicali e per la successiva segmentazione di un brano musicale, faranno inizialmente riferimento alla fuga e alla forma sonata per poi essere successivamente generalizzati ad altre forme musicali.

SampleBuster

Il modulo **SampleBuster** consisterà in un sistema ipertestuale, sviluppato in ambiente HyperCard su computer Macintosh, che permetterà di selezionare campioni sonori, appartenenti ad una banca di 7000 suoni, sulla base di attributi multipli.

Il sistema si articolerà in più stack, attivi in ambiente HyperCard, che consentiranno di effettuare le seguenti attività: controllo diretto dei campionatori Akai S950 e S1000, collegati al Macintosh via MIDI, per il trasferimento (modalità *MIDI Sistema Esclusivo*) e l'editing di campioni sonori e di patch; consultazione di una biblioteca di informazioni sugli strumenti musicali, sia di carattere organologico, sia riguardanti le possibilità di collocazione degli strumenti stessi in organici per l'esecuzione di musiche appartenenti a generi diversi (dalla musica sinfonica e operistica, al jazz e al rock); ricerca selettiva, sulla base di attributi multipli selezionabili dall'utente, di campioni sonori appartenenti ad una banca costituita da diverse collane di suoni, tra cui Akai "Workstation" e CD McGill University "Master Samples", distribuite su supporti differenziati (floppy disk per campionatori, CD-DA).

ENA

Il modulo **ENA** (Experimental Neural Accompanist), sviluppato in ambiente Macintosh, consisterà in un'applicazione per la generazione in tempo reale di accompagnamenti in corrispondenza dell'esecuzione di parti musicali solistiche.

Il modulo sarà basato su reti neurali di tipo "back propagation" che verranno istruite mediante un insieme di melodie tonali, con relativi accompagnamenti e armonizzazioni, di autori classici del Settecento. Il solista potrà utilizzare per l'esecuzione una tastiera MIDI collegata al Macintosh oppure la tastiera ASCII. Il modulo si presterà particolarmente per applicazioni nel campo della valutazione delle performance musicali o per l'esecuzione diretta di composizioni ad hoc.

DIST.Music.Tool

Il modulo **DIST.Music.Tool** consisterà in un ambiente (programmi e librerie) per la generazione e l'editing di campioni sonori.

In particolare, DIST.Music.Tool sarà costituito dai seguenti componenti: *STED* (programma per l'editing di campioni); *AKAIComm* (programma per la gestione della comunicazione PC-campionatore Akai S900); *HDMusic Toolkit* (libreria di routine e programma "monitor" per la ricezione/trasmissione in DMA di campioni sonori via ADC/DAC per mezzo dell'apposita scheda compresa nel kit e montata nel PC, e per la scrittura/lettura su hard disk dei dati corrispondenti). Inoltre, sempre nell'ambito DIST.Music.Tool, verranno portati in ambiente PC, adattati ed espansi, il modulo *CARL* (sviluppato all'Università della California di S.Diego) contenente il linguaggio per la composizione musicale C-Music, e il modulo *CMU MIDI Toolkit* (sviluppato all'Università Carnegie-Mellon) per la gestione di campioni in ambiente MIDI (mediante l'interfaccia MIDI per PC Roland MPU-401).

Rappresentazione della conoscenza musicale: *Key-Music*, *HARP*, *JAM*

I moduli **Key-Music** ed **HARP** (Hybrid Action Representation and Planning) hanno l'obiettivo di fornire la soluzione per alcuni dei problemi più significativi legati alla rappresentazione di conoscenza in ambito compositivo. In particolare, *Key-Music* è un sistema per la rappresentazione di conoscenza musicale basato su reti di Petri e reti semantiche di frame ad eredità multipla; *HARP* consiste invece in un sistema ibrido, ovvero che integra aspetti simbolici e analogici.

Entrambi traggono i loro fondamenti dal fatto che, in musica, il **tempo** gioca un ruolo fondamentale: un fenomeno musicale (ad esempio l'esecuzione di un brano) può essere rappresentato mediante attori musicali interagenti e temporizzati (l'analogia orchestra/direttore con processi concorrenti/scheduler è, seppur banale, evocativa dell'importanza di tale aspetto nella rappresentazione).

Key-Music è basato su un modello a reti semantiche per la rappresentazione della **informazione descrizionale**, e sulle reti di Petri per la **informazione fattuale** su attività concorrenti e temporizzate. Verrà utilizzato un sistema basato su reti semantiche e frame per rappresentare la conoscenza che concerne definizioni generali su processi musicali temporizzati, sotto forma di reti di frame; in particolare, verranno sfruttate le capacità di inferenza caratteristiche di tale formalismo: **classificazione ed ereditarietà**. Rappresentazioni di processi individuali o attuali possono essere automaticamente generate come **istanze** di una definizione generale, in termini di **reti di Petri**.

Tale modello è alla base di un modulo della stazione di lavoro dedicato all'ausilio alla composizione ed alla didattica. Una implementazione prototipale è in corso di realizzazione, utilizzando il linguaggio Lisp (CLOS - Common Lisp Object System) su workstation Unix/X-Window.

Le informazioni contenute in un tale tipo di base di conoscenza sono la rappresentazione astratta simbolica in grado di generare **automaticamente** molte specifiche istanze in termini di reti di Petri: tale modello è stato denominato **A-Net** (Action Nets). Le A-Net possono essere viste come una "superclasse" di reti di Petri ad un livello di astrazione superiore. A questo livello elaboriamo oggetti più astratti sui quali possiamo intervenire in due modi: derivando particolari oggetti individuali (reti di Petri) o ragionando attorno alle loro proprietà come in qualsiasi sistema di rappresentazione della conoscenza. Ad esempio, possiamo dire che il livello più alto consente di descrivere **azioni** di tipo generale quali "comporre una fuga" o "improvvisare su un tema"; il corrispondente livello di rete di Petri (il più basso) è, ad esempio, in questo caso "componi una fuga a 3 voci, in Re maggiore, utilizzando il soggetto X ed il controsggetto Y" o "improvvisa sul tema X...".

La metodologia descritta è alla base di un sistema prototipale, denominato **JAM**, per la rappresentazione di conoscenza musicale. In tale prototipo verrà introdotta conoscenza riguardante armonia funzionale, pattern melodici e ritmici, riguardante un ben determinato stile musicale. Il sistema dovrà essere in grado di generare (in modo automatico o guidato) improvvisazioni su schemi armonici fissate, eventualmente interagendo con l'utente in tempo reale in tutte le fasi della creazione. La base di conoscenza verrà strutturata in forma di Action-Nets.

SOUL

L'obiettivo dell'ambito di ricerca del modulo **SOUL** (Self-Organizing Universal Listener) è lo studio preliminare di un sensore acustico intelligente adattivo per il riconoscimento e l'inseguimento di una sorgente sonora, basato sull'impiego di modelli connessionistici. In ambito musicale e psico-acustico, è nota la capacità biomorfa di estrarre la singola sorgente sonora (lo strumento) da un background non necessariamente definibile in termini di grandezze tipiche della teoria dei segnali (frequenze, intensità, correlazione, ecc.). Si tratta di una attività intelligente (coinvolge livelli di astrazione) e adattiva (basata sull'apprendimento). La tecnologia del sensore periferico è già ben consolidata e disponibile a basso costo (il trasduttore elettroacustico), mentre è da investigare la

parte intelligente del sistema sensoriale, per cui la tecnologia corrente (elaborazione digitale su h/w dedicato, sistemi basati su conoscenza, data fusion) non è soddisfacente.

La ricerca si basa sull'impiego di reti neurali; il modello che più sembra significativo è quello delle **self-organizing network**, sviluppato da Kohonen. Tale tipo di reti neurali dovrebbe consentire, dopo opportuno training, la formazione di aggregazioni di unità accordate a particolari pattern.

Il modulo prototipale SOUL consisterà di un sistema di simulazione integrato alla acquisizione e pre-processing del suono, e ad una interfaccia grafica per il monitoraggio della rete. Il modello dell'ascoltatore sarà formalizzato.

Routine di lettura/scrittura di Standard MIDI File v 1.0

Il modulo **Routine di lettura/scrittura di Standard MIDI File v.1.0** consisterà in una libreria di routine, sviluppata in ambiente MPW su computer Macintosh, che permetteranno di leggere e scrivere archivi di dati MIDI secondo le specifiche fornite dalla *International MIDI Association*..

Le routine saranno utilizzabili in ambiente MPW, e consentiranno di svolgere le funzioni inerenti la scrittura/codifica e la lettura/decodifica di Standard MIDI File (versione 1.0) nei formati 0 e 1, secondo le specifiche fornite dalla *International MIDI Association*. In particolare è prevista l'implementazione di routine per la codifica e la decodifica di numeri espressi in forma di "variable-length quantity" (rappresentazione "compressa" di valori numerici utilizzata negli SMF), routine per la lettura e la scrittura di header generali e header di traccia, routine di lettura/decodifica e codifica/scrittura di metaeventi, routine di lettura/decodifica e codifica/scrittura di eventi, routine di supporto per la diagnostica.

Ambiente sperimentale per microprogrammi DSP per Sound Accelerator Card

Il modulo **Ambiente sperimentale per microprogrammi DSP per Sound Accelerator Card**, sviluppato in ambiente MPW su computer Macintosh con scheda Digidesign Sound Accelerator (basata su DSP Motorola 56001), consentirà di costruire e modificare modelli timbrici basati su alcune tecniche specifiche di sintesi del suono (Wavelet, Waveshaping, TVF, Automi cellulari).

In particolare sarà costituito da quattro sottoambienti, uno per ciascuna tecnica di sintesi implementata: uno per l'editing di modelli timbrici basati sulla tecnica di sintesi per Wavelet; un altro per l'editing di modelli timbrici basati

sulla tecnica di sintesi per Waveshaping; un altro ancora per l'editing di modelli timbrici basati sulla tecnica di sintesi per funzioni a due variabili; infine l'ultimo, *LASy* (Linear Automata Synthesis) per l'editing di modelli timbrici basati su una tecnica di sintesi che utilizza automi cellulari come modelli per l'evoluzione dei suoni. I modelli timbrici generati per mezzo delle quattro tecniche di sintesi potranno essere archiviati in formato Soundfile Digidesign per poi essere utilizzati da altri moduli della SLMi (ad es. *Sequencer MIDI/DSP*) o dalle applicazioni Digidesign (ad es. *SoftSynth*).

P5. Standard utilizzati nell'implementazione.

Al fine di rendere i moduli della SLMi integrati pienamente con i moduli applicativi reperibili in commercio è stato convenuto di adottare un insieme di standard per la codifica dei dati e per lo sviluppo del software di cui i seguenti sono i più rilevanti:

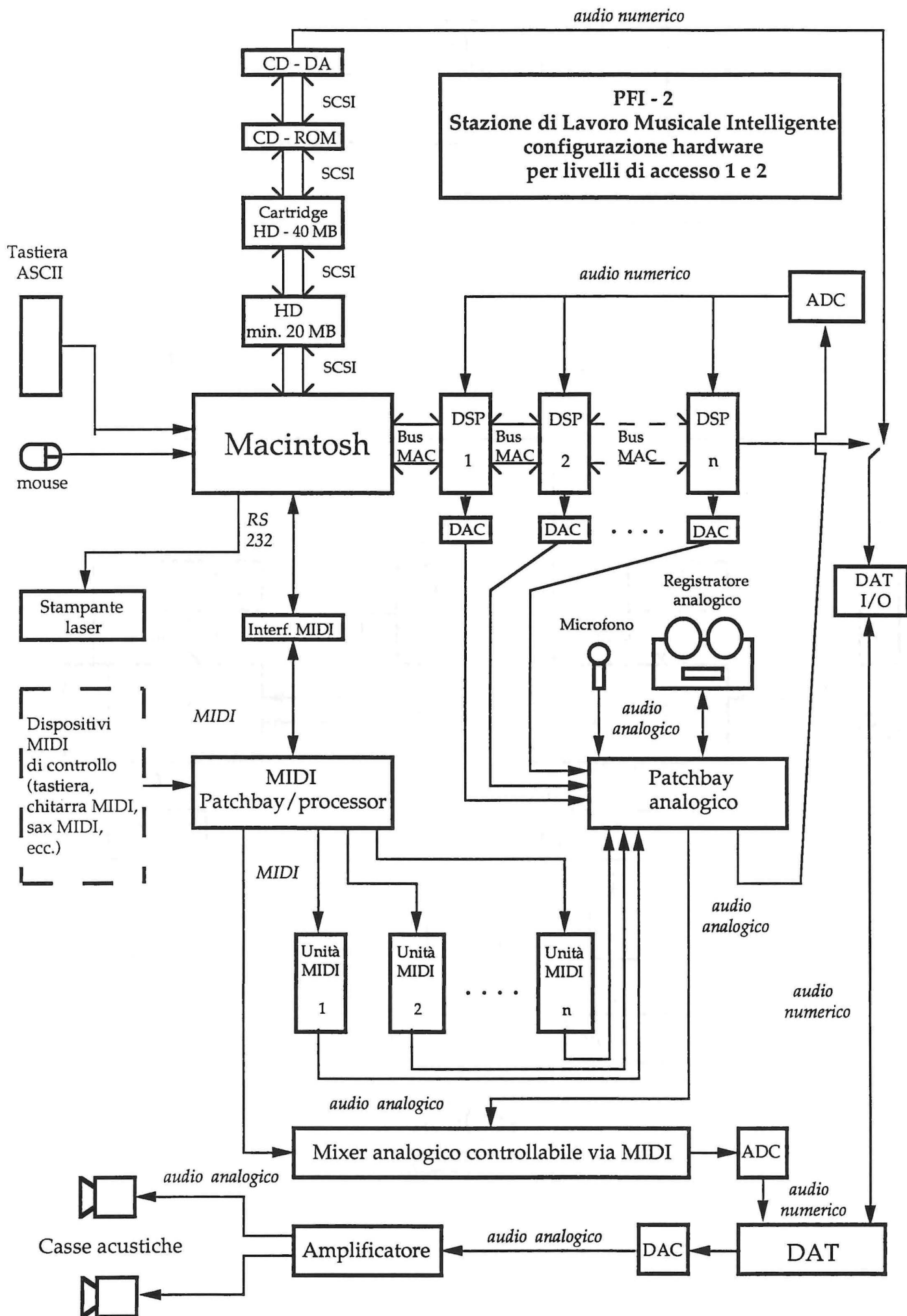
s1) *MIDI Manager Tool Set* : come ambiente standard per lo sviluppo di moduli che utilizzano l'interfaccia MIDI sotto Multifinder (ambiente multitasking per Macintosh); consente di connettere logicamente in tempo reale i flussi MIDI in ingresso e in uscita ai/dai vari moduli della SLMi e dei moduli commerciali implementati nel medesimo ambiente; in tal modo i vari moduli si scambiano flussi di dati in tempo reale consentendo così di costruire applicazioni più complesse combinando opportunamente i singoli moduli;

s2) *Standard MIDI File v.1.0* : come standard per la codifica dei file contenenti partiture a livello MIDI; consente di scambiare tali informazioni tra tutti i moduli della SLMi e con i moduli commerciali che operano a livello di partitura MIDI;

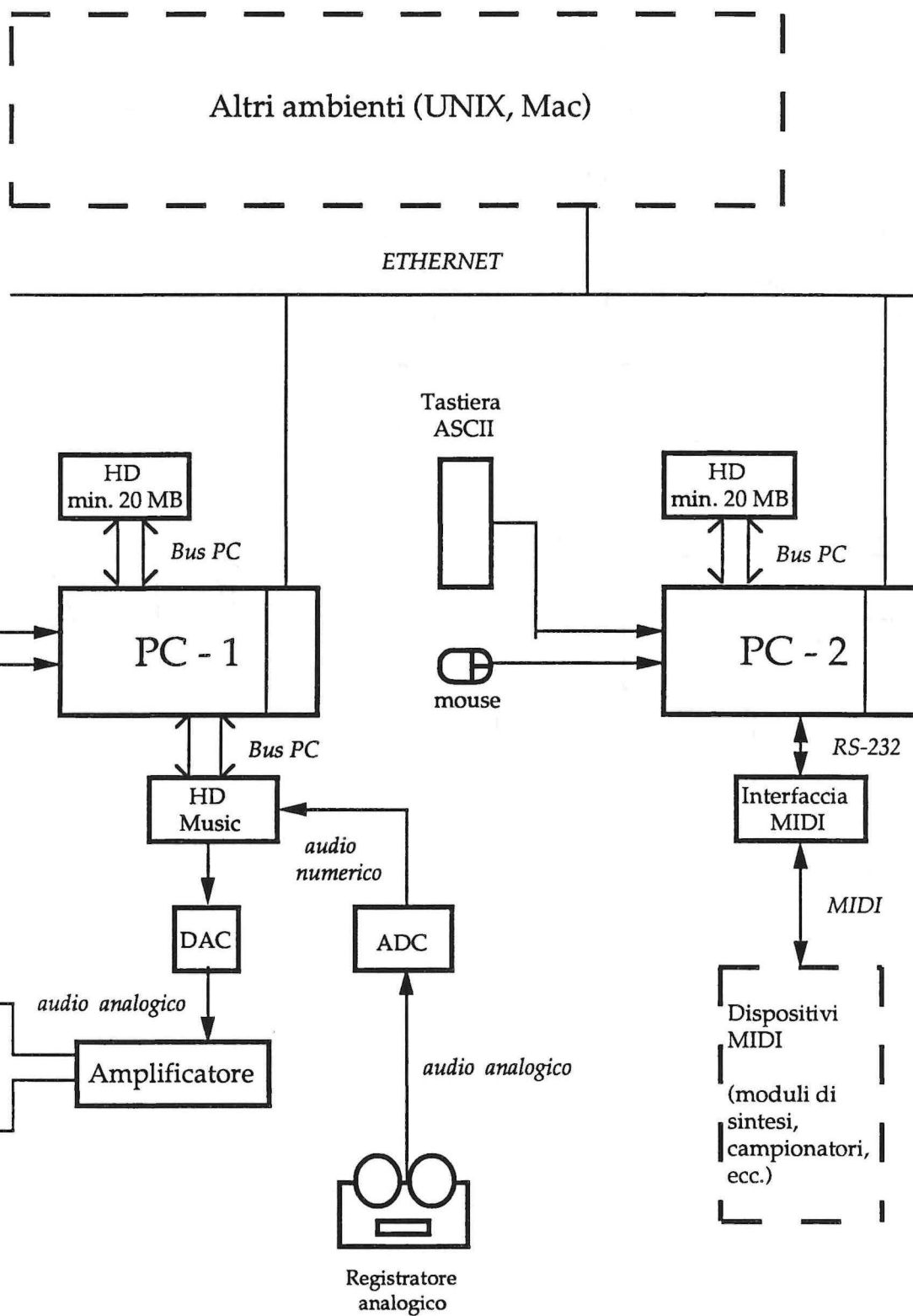
s3) *Digidesign Soundfile* : come standard per la codifica di dati di tipo sonoro (campioni sonori, master digitali); consente di scambiare dati di tipo sonoro tra tutti i moduli di editing, di registrazione, di sintesi e di analisi del suono in forma numerica.

P6. Configurazione hardware.

Nei diagrammi presenti nelle figure delle due pagine seguenti, sono rappresentate le configurazioni hardware che supportano i livelli S/W (1, 2 e 3) di accesso alla SLMi.



PFI - 2
Stazione di Lavoro Musicale Intelligente
configurazione hardware
per livello di accesso 3



Parte II.

P7. Introduzione.

Nei paragrafi che seguono è contenuta una guida schematica ed esauriente per i ricercatori che sviluppano software nell'ambito della **SLMI**, affinché il nucleo della documentazione che riguarda l'installazione e l'uso dei singoli moduli e le informazioni per altri ricercatori che tali moduli dovranno modificare o integrare, risulti il più possibile coerente e completo.

P7.1 Generalità.

Per ogni modulo, il nucleo della documentazione deve necessariamente comprendere due parti: la prima riguarda le specifiche funzionali del modulo sotto forma di paragrafo del Rapporto tecnico R3 (ved. P8); la seconda consiste in una guida operativa per l'utilizzazione del modulo, sotto forma di "Documento operativo" (ved. P9). Per quanto riguarda quest'ultimo, bisogna tener presente che nel caso in cui il modulo sia un'applicazione, la guida operativa sarà costituita a sua volta da due componenti indipendenti: "Manuale d'uso" (ved. P9.1 e A1) e "Note tecniche" (ved. P9.2 e A2); mentre nel caso in cui il modulo sia di supporto al software di sistema (libreria, driver, ecc.), la guida operativa sarà costituita da un'unica componente: "Manuale di riferimento" (ved. A3).

P7.2 Struttura della Parte II.

Il paragrafo P8 traccia le linee generali di impostazione per la stesura dei paragrafi del Rapporto tecnico R3 ("Specifiche funzionali dei moduli della STAZIONE DI LAVORO MUSICALE INTELLIGENTE"), e comprende i sottoparagrafi P8.1 e P8.2 in cui vengono descritti sinteticamente gli strumenti formali utilizzati per la descrizione delle specifiche funzionali dei moduli appartenenti rispettivamente ai livelli 1 o 2 e al livello 3.

Il paragrafo P9 traccia le linee generali di impostazione per la stesura dei "Documenti operativi" di tipo "Guida operativa del modulo ...", e comprende i sottoparagrafi P9.1 e P9.2 in cui viene descritto sinteticamente il contenuto

minimo che deve essere compreso nel "Manuale d'uso" e nelle "Note tecniche".

Il paragrafo P10 elenca alcune note di stile la cui osservanza, se non imperativa, si deve ritenere vivamente consigliata, onde rendere la documentazione il più possibile intelligibile e omogenea a prescindere dal dominio applicativo dell'argomento trattato.

In Appendice A1 e A2 sono riportati, per comodità del lettore, gli indici tipo del "Manuale d'uso" e delle "Note tecniche" senza le note descrittive che compaiono nei paragrafi P9.1 e P9.2.

Infine in Appendice A3 viene descritto un esempio di "Manuale di riferimento" utile per chi deve sviluppare software di supporto a quello di sistema (driver, librerie, ecc.).

Gli indici tipo dovrebbero costituire lo scheletro della documentazione, la cui suddivisione in paragrafi rispecchierà fedelmente, anche come successione sequenziale, la struttura degli indici. Ferma restando la possibilità di omettere o variare, in funzione del software da documentare, le singole voci degli indici.

P7.3 Nota preliminare.

In questo documento compaiono precisi riferimenti, talvolta molto espliciti, all'ambiente Macintosh. La motivazione di questa preferenza è da ricercarsi nel fatto che la maggior parte dei moduli che costituiscono la SLMI verranno sviluppati in ambiente Macintosh. Tuttavia l'impianto generale proposto per la documentazione del software, si intende applicabile anche ad altri ambienti di sviluppo (ad es. Unix/X-window).

P8. Rapporto tecnico R3.

In questo paragrafo vengono date alcune indicazioni basilari per la compilazione dei paragrafi del Rapporto tecnico R3 ("Specifiche funzionali dei moduli della STAZIONE DI LAVORO MUSICALE INTELLIGENTE"). I paragrafi in cui sarà suddiviso tale rapporto, uno per ciascun modulo della SLMI, conterranno informazioni atte a definire in modo completo le caratteristiche di ogni modulo, sia in ambito locale (ambientazione teorico-concettuale, descrizione non formale delle funzionalità, specifiche funzionali espresse in modo formale), sia nell'ambito della SLMI (livello di accesso di appartenenza,

sottoambiente di definizione, relazioni con altri moduli). A causa della complessità di struttura della SLMI e dell'appartenenza dei singoli moduli a differenti livelli di accesso e sottoambienti operativi, oltre che a differenti domini concettuali, non si propone di seguito un'organizzazione rigida del documento, bensì vengono date precise indicazioni di contenuto in forma di indice qualitativo.

Termini di contenuto:

- Parte introduttiva con ambientazione culturale/concettuale necessaria per la comprensione delle funzionalità del modulo e delle sue applicazioni in campo informatico musicale.
- Descrizione informale (in linguaggio naturale) delle funzionalità del modulo.
- Collocazione del modulo nell'architettura e nell'ambito funzionale della SLMI.
- Definizione dei requisiti non funzionali (requisiti hardware, software e firmware necessari per rendere operativo il modulo, eventuali limiti imposti sulle prestazioni e/o sulle funzionalità operative).
- Specifiche funzionali, espresse utilizzando i formalismi indicati nei sottoparagrafi 8.1 e 8.2, secondo l'appartenenza del modulo ad un particolare livello di accesso alla SLMI.
- Informazioni accessorie da riportare in appendice ed eventualmente termini specialistici e/o ambigui da riportare nel glossario del rapporto.

Per quanto riguarda i formalismi adottati per la stesura delle specifiche funzionali, viene fatta una precisa distinzione di metodo per quanto riguarda i moduli appartenenti alla shell di ausilio alla composizione musicale (livello cognitivo) rispetto a quelli appartenenti agli altri due livelli (simbolico/strutturale, esecutivo/operativo). Per i primi verrà utilizzato un formalismo basato su reti semantiche ad eredità multipla di tipo KL-ONE (ved.

sottoparagrafo P8.2); mentre per quanto riguarda i secondi verrà utilizzato un formalismo basato su reti di Petri (Haus, 1980) opportunamente adattato e supportato da descrizioni procedurali e di strutture dati espresse in una particolare variante del linguaggio Ada, denominata Ada/PDL (Sommerville, 1989) (ved. sottoparagrafo P8.1).

P8.1 Strumenti formali per la descrizione delle specifiche funzionali dei moduli dei livelli 1 e 2.

Il formalismo adottato per descrivere le specifiche funzionali dei moduli appartenenti ai livelli simbolico/strutturale ed esecutivo/operativo di accesso alla SLMi (ved. Parte I di questo rapporto), è basato su una metodologia che mutua due approcci complementari alla descrizione di modelli funzionali di sistemi informativi.

Il primo consente di descrivere graficamente un modulo software evidenziando le relazioni funzionali e di flussi informativi tra procedure, strutture dati e archivi. E' basato sull'utilizzazione di reti di Petri PT (Posti/Transizioni) strutturate, interpretate in termini di risorse e attività, ed è descritto diffusamente in (Haus, 1980). La notazione specifica dei nodi verrà adattata a seconda delle caratteristiche del modulo da descrivere e la relativa descrizione in dettaglio sarà contenuta in R3.

Il secondo approccio consente di completare la descrizione dei moduli, realizzata mediante l'approccio precedente, formalizzando in termini di pseudocodice Ada-like e di strutture dati astratte le risorse e le attività. E' basato su un'estensione del linguaggio Ada (Ada/PDL) che tende a superare alcuni limiti del linguaggio rispetto alla descrizione di specifiche, ed è descritto diffusamente in (Sommerville, 1989 - Cap. 5).

P8.2 Strumenti formali per la descrizione delle specifiche funzionali dei moduli del livello 3.

Il progetto di sistemi basati su conoscenza è un processo complesso. Spesso applicati a domini scarsamente formalizzati, per i quali non esistono soluzioni all'interno delle tecniche standard di programmazione, lo sviluppo di tali sistemi richiede, attualmente, la collaborazione di specialisti del dominio applicativo e di ingegneri della conoscenza. Il processo è un raffinamento iterativo: lo specialista

descrive conoscenza sul dominio all'ingegnere, che a sua volta codifica tale conoscenza utilizzando strumenti e tecniche di rappresentazione della conoscenza; successivamente, tale processo è ripetuto. Tale procedimento soffre di numerose difficoltà, che si manifestano in tutti i numerosi passi coinvolti nel trasferimento di conoscenza dallo specialista al sistema.

La particolare classe di sistemi software basati su tecniche di intelligenza artificiale (IA) è quindi caratterizzata da particolari approcci di ingegneria del software, tuttora oggetto di ricerca. Diverse tecniche per lo sviluppo di sistemi basati sulla conoscenza sono attualmente oggetto di studio (si vedano ad es. Schoen, Smith & Buchanan, 1988); esse riguardano essenzialmente l'automatizzazione delle fasi di acquisizione della conoscenza relativa al dominio applicativo.

Tali considerazioni sono rilevanti nell'ambito della produzione di documentazione e specifiche funzionali per sistemi basati su conoscenza. Tali specifiche devono in particolare comprendere la definizione delle tecniche utilizzate per la rappresentazione della conoscenza, della struttura e organizzazione della base di conoscenza del dominio, dei metodi di interfaccia utente.

I moduli basati su tecniche di IA riguardano essenzialmente sistemi per la rappresentazione di conoscenza musicale. Essi sono caratterizzati dall'utilizzo di tecniche non standard di rappresentazione della conoscenza: i moduli Key-Music e Jam si basano su un modello di rappresentazione che integra reti semantiche ad eredità multipla e reti di Petri, Harp su reti semantiche di tipo KL-ONE e tecniche di rappresentazione subsimboliche (reti neurali). Le specifiche funzionali per tale tipo di moduli si basano sui formalismi adottati per la rappresentazione della conoscenza: una base di conoscenza KL-ONE costituirà lo schema concettuale del sistema in oggetto. In particolare, verrà utilizzata una estensione del KL-ONE (Brachman & Schmolze, 1985), (caratterizzato da utili caratteristiche, quali classificazione, ereditarietà, object-oriented, definizione formale precisa), e le reti di Petri, diffuse come approccio funzionale nell'ambito dell'ingegneria del software. Tale approccio ibrido complessivo per la descrizione del processo di sviluppo del software, integrante sia gli aspetti statici (reti semantiche di tipo KL-ONE) che dinamici (reti di Petri), si basa su concetti derivati da formalismi recentemente sviluppati (Kung, 1989; Liu & Horowitz, 1989).

P9. Documentazione operativa.

In questo paragrafo viene descritto in modo essenziale come compilare i "Documenti operativi" di tipo "Guida operativa del modulo ...", tenendo presente che nel nostro contesto si possono presentare due casi tipici, in corrispondenza di moduli software di natura sostanzialmente diversa: applicazioni o moduli di supporto al software di sistema (librerie, driver, ecc.).

Nel caso in cui il modulo da documentare sia un'applicazione, la guida operativa sarà costituita a sua volta da due componenti indipendenti: "Manuale d'uso" (ved. P9.1 e A1) e "Note tecniche" (ved. P9.2 e A2), che sono in pratica un adattamento al nostro contesto dei manuali standard "User's Guide" e "Reference Manual", forniti generalmente con le applicazioni commerciali.

Invece nel caso in cui il modulo da documentare sia un modulo di supporto al software di sistema, la guida operativa sarà costituita da un'unica componente: "Manuale di riferimento" (ved. A3), adattamento al nostro contesto del manuale "Reference Manual", fornito generalmente con analogo software commerciale.

P9.1 "Manuale d'uso".

Il **Manuale d'uso** è un documento che deve mettere un qualsiasi utente nella condizione di poter utilizzare agevolmente l'applicazione. Quindi è molto importante che le informazioni in esso contenute, oltre che essere attendibili e complete, siano presentate in modo chiaro, semplice e possibilmente utilizzando un linguaggio di uso comune. Qualora si utilizzino termini specialistici o gergali (ad esempio parole straniere difficilmente traducibili in modo intelligibile), è buona norma corredarli di una spiegazione e, se il numero di tali termini è sufficientemente elevato, conviene dotare il documento di un **glossario**.

Nel seguito viene proposto un indice tipo che comprende le informazioni essenziali che dovranno essere contenute nel "Manuale d'uso".

1. Introduzione.

1.1 Generalità sull'applicazione <nome_applicazione>.

In questo paragrafo, che per ragioni di impostazione generale deve essere di dimensioni molto ridotte, sono contenute in modo estremamente sintetico le informazioni essenziali per far capire all'utente:

- Chi ha sviluppato l'applicazione; dove e quando è stato svolto il lavoro.
- In quale ambiente hardware/software l'applicazione può essere eseguita (in breve, senza scendere in dettaglio).
- Che cosa fa l'applicazione e a che cosa serve.

1.2 Requisiti hardware per il corretto funzionamento.

- Descrizione dettagliata dell'hardware (modello o famiglia di computer) richiesto per il corretto funzionamento dell'applicazione.
- Eventuali componenti o dispositivi accessori necessari o consigliabili per il corretto funzionamento dell'applicazione o per estenderne le possibilità operative.
Ad es. interfaccia MIDI, tastiera musicale, ecc.

1.3 Requisiti software per il corretto funzionamento.

- Descrizione dettagliata del software (versioni di System e Finder o Multifinder, eventuali altri moduli software con relative versioni) richiesto per il corretto funzionamento dell'applicazione.
- Eventuali moduli o applicazioni accessorie necessari o consigliabili per il corretto funzionamento dell'applicazione o per estenderne le possibilità operative.

Ad es. Accessori o Font particolari, altre applicazioni che possono interagire con l'applicazione o che possono elaborarne i dati, ecc.

1.4 Requisiti firmware per il corretto funzionamento.

- Descrizione dettagliata del firmware (versione ROM, eventuali Manager residenti utilizzati, ecc.) richiesto per il corretto funzionamento dell'applicazione.

1.5 Note sulla fase di testing.

- Descrizione dettagliata della configurazione hardware e software su cui è stata effettuata la fase di testing dell'applicazione (in altre parole, della configurazione hardware e software per cui si garantisce la piena operatività e funzionalità dell'applicazione).
- Descrizione operativa della fase di testing.

1.6 Note per l'installazione.

- elenco dettagliato dei files che costituiscono l'applicazione con relativa descrizione funzionale.

Ad es. l'applicazione è costituita da:

Appl programma principale.

Applhelp file contenente i dati per l'help in linea.

Applutil programma per la conversione di formato dei files.

- operazioni particolari che bisogna effettuare per rendere l'applicazione correttamente operativa.

Ad es. se bisogna copiare i files dell'applicazione in una cartella (directory) particolare; oppure se bisogna copiare un file di INIT nella cartella (directory) di sistema; ecc.

2. Introduzione all'uso dell'applicazione <nome_applicazione>.

2.1 L'interfaccia utente.

- Descrizione dettagliata dell'interfaccia utente con la spiegazione di tutte le convenzioni adottate (grafiche e non) e dei simboli utilizzati.

2.2 Un esempio introduttivo.

- Questo paragrafo dovrebbe essere costruito sul modello della sezione "Getting started" che si trova comunemente (sotto varie denominazioni) nei manuali operativi delle applicazioni diffuse commercialmente. Quindi deve consistere in una guida accurata per la realizzazione pratica di un esempio di utilizzazione dell'applicazione che sia allo stesso tempo semplice, significativo e che faccia riferimento a situazioni concrete.

3. Guida operativa dell'applicazione <nome_applicazione>.

- Manuale dettagliato organizzato per Menu gerarchici. L'organizzazione di questo capitolo deve rispecchiare la struttura dell'applicazione, e dovrebbe costituirne una specie di carta di navigazione. In questo modo l'utente dovrebbe essere messo in grado di trovare facilmente le informazioni che gli sono necessarie durante l'utilizzazione dell'applicazione.

4. Esempi di utilizzazione avanzata dell'applicazione <nome_applicazione>.

- Questo capitolo conterrà alcuni esempi significativi più complessi di quello descritto nel paragrafo 2.2, scelti in modo da presentare all'utente alcune caratteristiche funzionali e/o modalità operative dell'applicazione, di attuazione non intuitiva o di particolare interesse applicativo.

Appendici.

Le Appendici sono una parte molto importante della documentazione. In esse è possibile inserire tutte le informazioni accessorie, ma non per questo trascurabili, che possono essere utili per una proficua utilizzazione dell'applicazione.

- Se nell'interfaccia utente dell'applicazione vengono utilizzate icone per guidare la scelta e l'attivazione di moduli operativi, sarebbe opportuno riunirle in Appendice, sotto forma di elenco, con una breve descrizione operativa delle relative funzionalità e il riferimento alla parte di manuale in cui queste ultime vengono descritte in dettaglio.
- Inserimento di informazioni utili connesse a moduli hardware e/o software particolari, necessari per il pieno e corretto funzionamento dell'applicazione; oppure informazioni inerenti al dominio applicativo in cui è ambientata l'applicazione; oppure ancora informazioni sull'uso dell'applicazione in concomitanza con altre applicazioni; ecc.
- Se risulta necessario (ved. introduzione al paragrafo P2.1), può essere inserito in Appendice un Glossario dei termini specialistici e/o ambigui che compaiono nel documento.

Guida rapida all'uso dell'applicazione <nome_applicazione>.

Un'utile componente opzionale della documentazione, soprattutto nel caso di applicazioni dotate di interfacce utente particolarmente complesse, può essere una "Guida rapida"; ovvero un opuscolo separabile dal documento che contenga tutte le informazioni essenziali (composizione Menu, significato icone, tasti funzionali, opzioni principali, valori numerici di parametri significativi, ecc.) affinché un utente evoluto possa utilizzare l'applicazione senza dover consultare necessariamente il manuale.

P9.2 "Note tecniche".

Il documento **Note tecniche** deve mettere un programmatore nella condizione di poter modificare (anche solo localmente), o di estendere, o di far comunicare con altre applicazioni, l'applicazione. Quindi è molto importante che le informazioni in esso contenute, oltre che essere attendibili e complete, siano presentate in modo chiaro, sintetico e preciso, utilizzando un linguaggio di uso comune in campo informatico. Qualora si utilizzino termini specialistici legati ad un particolare dominio applicativo, è buona norma corredarli di una spiegazione e se il numero di tali termini è sufficientemente elevato, conviene dotare il documento di un **glossario**. Nel seguito viene proposto un indice tipo che comprende le informazioni essenziali che dovranno essere contenute nel documento "Note tecniche".

1. **Introduzione.**
 - 1.1 Generalità sull'applicazione <nome_applicazione>.
 - 1.2 Requisiti hardware per il corretto funzionamento.
 - 1.3 Requisiti software per il corretto funzionamento.
 - 1.4 Requisiti firmware per il corretto funzionamento.
 - 1.5 Note sulla fase di testing.
 - 1.6 Note per l'installazione.

(per la stesura dei paragrafi precedenti, fare riferimento alle note descrittive che compaiono nei sottoparagrafi corrispondenti presenti nel paragrafo P9.1).

2. **Strutture dati.**
 - 2.1 **Strutture dati interne.**

- Descrizione dettagliata delle strutture dati interne all'applicazione (strutture, array, record, ecc.), espressa nel linguaggio di programmazione ad alto livello utilizzato per lo sviluppo, e facendo largo uso di commenti.

- 2.2 **Strutture dati esterne.**

- Descrizione dettagliata dei formati dei files utilizzati e/o generati dall'applicazione.

3. Struttura dell'applicazione.

- Descrizione dettagliata della struttura dell'applicazione, corredata da grafici con diagrammi funzionali (Reti di Petri) commentati.

4. Algoritmi.

- Descrizione dettagliata degli algoritmi utilizzati nell'applicazione, riportando anche la corrispondente parte di codice sorgente; oppure, se il codice è riportato integralmente in Appendice, fornendo precisi riferimenti per individuarne la parte di specifico interesse.

Appendici (ved. in P9.1-Appendici le note riguardanti "informazioni utili connesse a moduli hardware e/o software particolari" e relative a "Glossario").

Inoltre:

- Può essere necessario inserire in un'apposita Appendice il codice sorgente dell'applicazione (elenco dei moduli sorgenti, organizzato seguendo un criterio di adattamento alla struttura dell'applicazione come descritta nel capitolo 3-"Struttura dell'applicazione") commentato (commenti che spiegano almeno le attività di funzioni e procedure e il significato di tutte le variabili globali).

P10. Note di stile.

Le seguenti "Note di stile" non vanno interpretate come regole assolute, e quindi la loro osservanza non è da ritenersi imperativa. Invece si consiglia vivamente di usarle come guida e applicarle laddove sia necessario, onde rendere la documentazione il più possibile intelligibile e omogenea a prescindere dal dominio applicativo dell'argomento trattato.

In sintesi, e questo valga per tutte le "Note di stile", massima libertà di espressione, ma guidata dal buon gusto e illuminata dal buon senso!

1. *Meglio che usiate la forma personale (2° pers. plur. = "voi") piuttosto che la forma impersonale.*

Es. "Per installare il programma, dovete mandare in esecuzione ..." e non "Per installare il programma, si deve mandare in esecuzione ...".

2. *Non fate riferimenti a informazioni solamente per mezzo di un codice numerico.*

Invece dovete fornire il codice numerico di riferimento e contemporaneamente ricordare al lettore quello che il testo referenziato descrive. Es. piuttosto che scrivere "Nel paragrafo 1.3 ...", meglio scrivere "Nel paragrafo 1.3, che descrive la procedura di installazione del programma, ...".

3. *Dove possibile, enumerate soggetti differenti.*

E' di norma più chiaro descrivere soggetti differenti in forma di elenco piuttosto che in un'unica frase. Utilizzate attributi di testo (corsivo, sottolineato) per enfatizzare le parole chiave dell'elenco.

4. *Se una descrizione è complessa, siate ridondanti.*

E' spesso una buona idea presentare due o più descrizioni differenti dello stesso soggetto. Se il lettore fallisce nel comprendere completamente una descrizione, può beneficiare di altre possibilità (lo stesso soggetto spiegato in modo diverso) per chiarirsi le idee.

5. *Siate precisi e definite i termini che usate.*

La terminologia informatica è molto fluida e molti termini hanno più di un significato. Quindi, se vengono usati termini specifici (ad es. *modulo* o

processo), assicuratevi che la vostra definizione sia chiara. Se avete bisogno di definire un certo numero di parole in funzione di un argomento particolare, o se state scrivendo per lettori con una conoscenza limitata o addirittura nulla della terminologia informatica, dovrete dotare il vostro documento di un glossario. Questo deve contenere le definizioni di tutti i termini che possono non essere completamente capiti dal lettore.

6. *Non usate due o più termini riferiti allo stesso soggetto.
Non usate un unico termine riferito a più soggetti differenti.*

E' molto facile cadere in trappole linguistiche che ingenerano ambiguità. Quindi dovete fare molta attenzione nel mantenere la massima coerenza quando usate un termine per indicare un determinato soggetto. Non dovete assolutamente cambiarlo nel corso dell'esposizione (anche rinunciando in qualche caso a un po' di eleganza espositiva!). Naturalmente dovete cercare anche di non cadere nella trappola opposta, cioè usare un unico termine riferendolo a due o più soggetti differenti.

7. *Se dovete usare parole inglesi, non italianizzatele.*

Spesso tradurre un termine tecnico in italiano, anche nel caso in cui sia possibile, può ingenerare delle ambiguità. D'altra parte molti autori hanno contratto la cattiva abitudine di "italianizzare" i vocaboli tecnici stranieri, con risultati di dubbio gusto oltre che non sempre adeguati alle necessità espressive. Quindi è preferibile, anche se richiede qualche piccolo sforzo aggiuntivo, utilizzare perifrasi contenenti il vocabolo in lingua originale, piuttosto che inventare neologismi o utilizzare la prosa disdicevole di certa stampa divulgativa o di certi gerghi aziendali. Es. piuttosto che scrivere "settare", "debuggare", "digitare", ecc. meglio scrivere "effettuare il setting", "effettuare il debugging", "immettere dati con la tastiera alfanumerica", ecc. Anche se sono tollerabili italianizzazioni del tipo "compilare", "implementare", ecc. Per risolvere i dubbi, lasciatevi guidare dal buon senso e dal buon gusto!

NOTA: Si ricorda che i termini stranieri inseriti in un testo italiano vanno scritti sempre nella forma singolare, anche se l'uso è riferito a un numero plurale.

8. *Non siate prolissi.*

Se potete dire qualcosa in cinque parole fatelo, piuttosto che usare dieci parole per far sembrare la descrizione più accurata. Non ci sono vantaggi nella quantità di documentazione. La qualità è più importante.

9. *Usate titoli e sottotitoli.*

Questi suddividono un capitolo in parti che possono essere lette separatamente. Assicuratevi sempre di usare per questo scopo una convenzione di numerazione coerente.

10. *Mantenete brevi i paragrafi.*

Come regola generale, nessun paragrafo dovrebbe comprendere più di sette frasi. In tal modo tutti i concetti in un singolo paragrafo possono essere mantenuti dal lettore nella sua memoria a breve termine. Questo a causa delle limitazioni della nostra memoria a breve termine che è appunto in grado di mantenere memorizzati contemporaneamente al massimo sette elementi.

11. *Non usate un'unica frase lunga per descrivere più soggetti differenti.*

E' meglio usare diverse frasi brevi. Ogni frase può quindi essere assimilata autonomamente. Il lettore non deve così tenere a mente diversi pezzi di informazione in una volta per comprendere tutta la frase.

12. *Usate costrutti grammaticalmente corretti.*

Non curare l'aspetto dell'impaginazione e l'ortografia del testo irrita molti lettori e riduce la credibilità dello scrivente ai loro occhi, oltre che compromettere l'intelligibilità del testo stesso.

P10. Riferimenti bibliografici.

Brachman & Schmolze (1985). An Overview of the KL-ONE Knowledge Representation System. Cognitive Science. 9(2), 171-216.

Haus, G. (1980). Metodi di definizione di specifiche funzionali di procedure software con uso di reti di Petri. Documento n. METOD-24, PFI-CNR, Sottoprogetto P1, Obiettivo METOD.

Kung, C.H. (1989). Conceptual Modeling in the Context of software development. IEEE Trans. on Software Engineering, 15(10), 1176-1187.

Liu, L.C., & Horowitz, H. (1989). A Formal Model for Software Project Management. IEEE Trans. on Software Engineering, 15(10), 1280-1293.

Schoen, E., Smith, R.G., & Buchanan, B.G. (1988). Design of Knowledge-based Systems with a Knowledge-based Assistant. IEEE Trans. on Software Engineering, 14(12), 1771-1790.

Sommerville, I. (1989). Software Engineering. Addison-Wesley Publ.

A1. Indice schematico del "Manuale d'uso".

- 1. Introduzione.**
 - 1.1 Generalità sull'applicazione <nome_applicazione>.**
 - 1.2 Requisiti hardware per il corretto funzionamento.**
 - 1.3 Requisiti software per il corretto funzionamento.**
 - 1.4 Requisiti firmware per il corretto funzionamento.**
 - 1.5 Note sulla fase di testing.**
 - 1.6 Note per l'installazione.**
- 2. Introduzione all'uso dell'applicazione <nome_applicazione>.**
 - 2.1 L'interfaccia utente.**
 - 2.2 Un esempio introduttivo.**
- 3. Guida operativa dell'applicazione <nome_applicazione>.**
- 4. Esempi di utilizzazione avanzata dell'applicazione <nome_applicazione>.**

Appendici.

Guida rapida all'uso dell'applicazione <nome_applicazione>.

A2. Indice schematico di "Note tecniche".

- 1. Introduzione.**
 - 1.1 Generalità sull'applicazione <nome_applicazione>.**

- 1.2 Requisiti hardware per il corretto funzionamento.
- 1.3 Requisiti software per il corretto funzionamento.
- 1.4 Requisiti firmware per il corretto funzionamento.
- 1.5 Note sulla fase di testing.
- 1.6 Note per l'installazione.

- 2. Strutture dati.
 - 2.1 Strutture dati interne.
 - 2.2 Strutture dati esterne.

- 3. Struttura dell'applicazione.

- 4. Algoritmi.

Appendici.

A3. Esempio di indice per un "Manuale di riferimento".

Nei casi in cui si devono documentare moduli di supporto al software di sistema (librerie, driver, ecc.), i manuali descritti precedentemente ("Manuale d'uso" e "Note tecniche") possono non essere adeguati. In tal caso sarà meglio utilizzare uno schema del tipo "Manuale di riferimento". Di seguito viene proposto, a titolo orientativo, un indice per un "Manuale di riferimento" tipo. L'indice è riferito a un modello reale costituito dal manuale che verrà realizzato per documentare una libreria di routine in fase di sviluppo nell'ambito del PFI-2 (ved. Rapporto D16 del presente progetto).

- 1. Introduzione.
 - 1.1 La LIBRERIA <nome_libreria> (ved. P9.1-1.1).
 - 1.2 Requisiti hardware e software (comprendente la descrizione della fase di testing; ved. P9.1-1.2, 1.3, 1.4, 1.5).
 - 1.3 Avvertenze per i programmatori (ved. P9.1-1.6).
- 2. Utilizzazione della LIBRERIA <nome_libreria>.
 - 2.1 Struttura della LIBRERIA <nome_libreria> (ved. P9.2-3,P9.2-4).
 - 2.2 Struttura interna dei dati (ved. P9.2-2.1).

2.3 Un esempio di utilizzazione (ved. P9.1-2.2).

- 3. Sezione di riferimento** (elenco delle routine, organizzato per categorie e per ogni categoria in ordine alfabetico, con la sintassi delle chiamate e la descrizione dettagliata degli argomenti e di tutti i parametri coinvolti).
- 4. Codice** (opzionale; elenco delle routine, organizzato come nel capitolo precedente, con il codice sorgente delle routine).

Indice (indice alfabetico delle routine, con riferimenti per individuare il dettaglio delle routine nei capitoli 3-"Sezione di riferimento" e 4-"Codice").

