Capitolo 6

Gestione dei dati

In questo capitolo e nel successivo viene presentata la fase di progettazione del sistema DEMI, cioè l'insieme delle attività che consentono di "passare" dalla specifica dei requisiti alla effettiva realizzazione del sistema.

Non viene discussa la realizzazione del global scheduler: questo modulo verrà progettato e implementato solo in una seconda fase, quando saranno meglio definiti i requisiti da considerare. Quindi nel primo prototipo si è considerato un ambiente costituito da un unico sito.

In questo capitolo in particolare si descrive il problema del trattamento delle varie informazioni, che devono essere gestite in maniera persistente. Viene presentata la progettazione dello schema concettuale di una base di dati, utilizzando il modello a oggetti, che si presta bene per "modellare" l'ambiente considerato. Viene poi descritta la trasformazione di questo schema in uno schema logico relazionale, per permettere l'implementazione della base di dati in un sistema di questo tipo.

6.1 Ambiente considerato

Nella realizzazione del primo prototipo si è considerato un ambiente costituito da un unico sito, come rappresentato in figura 6.1.

Quindi gli utenti possono definire propri task (1), che vengono utilizzati per la definizione di production (2, 3), le quali a loro volta vengono usate per sottomettere al sistema richieste computazionali (4, 5). Le richieste vengono "scomposte" in elementary request (6), che vengono analizzate dal local scheduler (7), responsabile del mapping dei corrispondenti task ai computer disponibili (8).

In questo primo prototipo non si considera quindi il global scheduler: verrà progettato e implementato solo in una seconda fase, in quanto non sono ancora ben definiti i criteri da considerare (ad esempio quando si devono "spostare" i

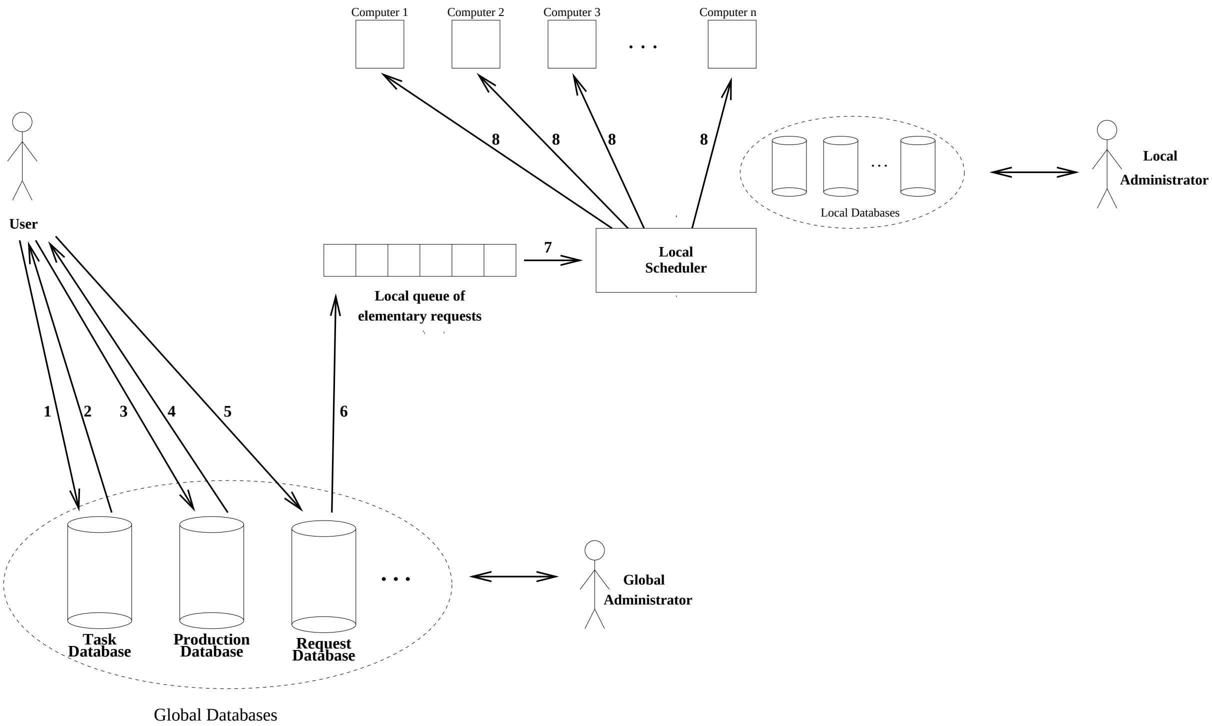

Figura 6.1: Ambiente considerato nella realizzazione del primo prototipo

dati tra i vari siti, o quando si devono invece "spostare" i job che devono essere eseguiti), e non sono ancora del tutto disponibili tutti i necessari strumenti (ad esempio validi tool per il monitor delle linee WAN in tempo reale, in modo da poter valutare l'opportunità dell'uso di tali link per la copia dei dati).

I database globali (Task Database, Production Database, Request Database, Architecture Database, ecc...) vengono gestiti dai global administrator, mentre i database locali (Local Computer Database, Local Buffer Database, Local Tape Device Database, Local Data Database, ecc...) vengono gestiti dai local administrator. Questi database sono comunque tutti fisicamente implementati sullo stesso sito.

6.2 Modello dei dati

Come visto nel capitolo 4, nel sistema DEMI devono essere considerati diversi tipi di informazioni, di natura diversa, fra le quali possono esistere specifiche relazioni, e sulle quali sono possibili determinati tipi di operazioni. È quindi fondamentale poter organizzare tutti questi dati in maniera adeguata, in modo da rendere il più efficiente possibile l'esecuzione di queste operazioni.

Tra i vari modelli esistenti per la gestione delle informazioni, negli ultimi tempi il *modello dei dati a oggetti* sta riscuotendo un grosso interesse, e sono sempre di più le implementazioni di questo modello in sistemi commerciali [6]. Il modello dei dati a oggetti, infatti, si presta bene per specificare in modo naturale la "realtà" che deve essere descritta, in quanto, a differenza degli altri modelli, permette di definire con gli stessi costrutti sia i dati, che le operazioni applicabili su di essi.

Il modello dei dati a oggetti può essere usato per la progettazione concettuale di una base di dati anche se poi si utilizzerà un DBMS non a oggetti (ad esempio un sistema relazionale). Infatti, visto il potere espressivo del modello a oggetti, è in genere conveniente usarlo sempre per la progettazione, per poi trasformare lo schema ottenuto nel modello del sistema considerato [6].

In appendice C vengono introdotti i concetti fondamentali di un modello a oggetti, descrivendo un formalismo grafico, proposto in [6], utilizzato per la definizione degli aspetti fondamentali dello schema di una base di dati.

Nel paragrafo 6.3 viene descritta la progettazione concettuale dello schema della base di dati per il sistema DEMI.

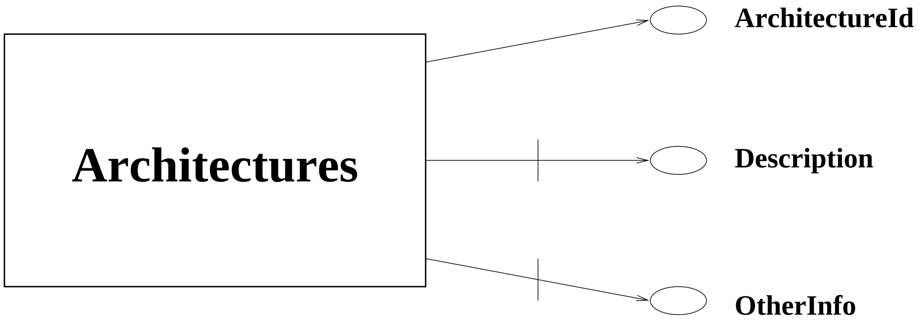

Figura 6.2: Classe 'Architectures'

6.3 Schema della base di dati per il sistema DEMI

In questa sezione viene definito lo schema concettuale della base di dati progettata per la "gestione" delle varie informazioni del sistema DEMI. Vengono prima di tutto elencate le classi utilizzate, specificando per ognuna di esse i campi dei relativi oggetti, ovvero gli attributi utilizzati per la rappresentazione dello stato. Si definiscono poi le associazioni esistenti tra le varie classi.

6.3.1 Architetture

In figura 6.2 è mostrata la classe 'Architectures', che rappresenta le architetture dei calcolatori utilizzati nel sistema DEMI. Quindi ogni elemento di questa classe rappresenta un elemento dell'Architecture Database (vedi par. 4.2.2).
Ad ogni oggetto della classe Architectures sono associati i seguenti campi:

- l'attributo *ArchitectureId* rappresenta l'identificatore dell'architettura (es. *alpha-win32*);
- l'attributo *Description* è la descrizione dell'architettura (es. *Computer con processore Alpha, con sistema operativo Windows NT*);
- l'attributo *OtherInfo* rappresenta eventuali altre informazioni relative all'architettura.

6.3.2 Modelli di tape device

In figura 6.3 è rappresentata la classe 'Tape Device Models', che definisce i modelli di tape device (e dei relativi nastri) utilizzati nel sistema DEMI. Ogni elemento

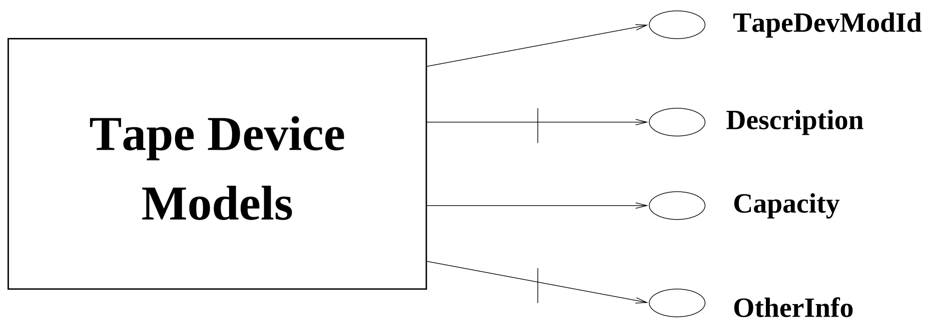

Figura 6.3: Classe 'Tape Device Models'

di questa classe rappresenta un elemento memorizzato nel Tape Device Model Database (vedi par. 4.2.3).
Lo stato di un oggetto di questa classe è definito dai seguenti attributi:

- l'attributo *TapeDevModId*, che rappresenta l'identificatore del modello di tape device (es. *DLT*);
- l'attributo *Description*, che descrive il modello di tape device (es. *tape device DLT, per nastri TZ87*);
- l'attributo *Capacity*, che definisce la capacità di un nastro di questo modello (es. *30 Gbyte*);
- l'attributo *OtherInfo*, che rappresenta eventuali altre informazioni relative a questo modello di tape device.

6.3.3 Global Administrator

In figura 6.4 è definita la classe 'Global Administrators', che rappresenta i global administrator del sistema DEMI. Questa classe corrisponde quindi al Global Administrator Database (vedi par. 4.2.4).
Gli attributi che definiscono lo stato degli oggetti della classe Global Administrators sono:

- *UserId*: specifica lo User Identifier del global administrator (es. *sgaravat*);
- *Password*: rappresenta la password (criptata) del global administrator;
- *EMailAddr*: definisce l'e-mail address del global administrator (es. *sgaravat@domain*);

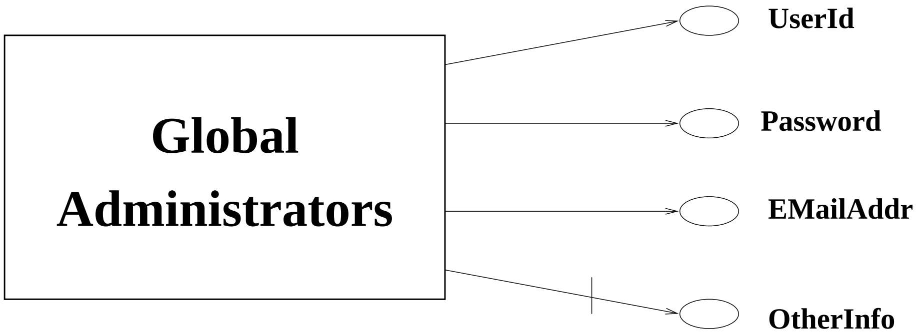

Figura 6.4: Classe 'Global Administrators'

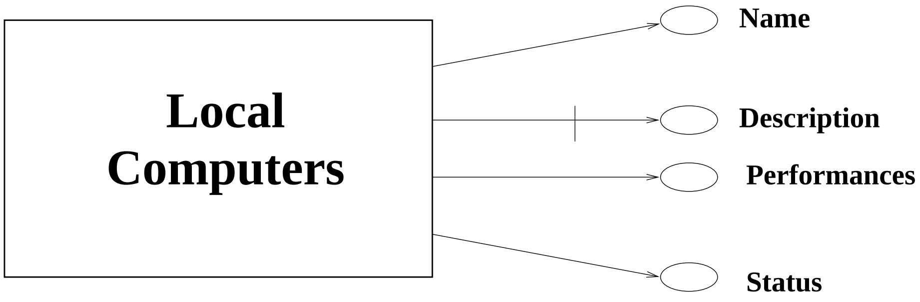

Figura 6.5: Classe 'Local Computers'

- *OtherInfo*: rappresenta eventuali altre informazioni relative al global administrator.

6.3.4 Computer

Per definire i computer presenti nel sito locale, memorizzati nel Local Computer Database (vedi par. 4.3.1), si definisce la classe 'Local Computers', rappresentata in figura 6.5.
Le proprietà degli oggetti di questa classe sono rappresentate da:

- l'attributo *Name*, che definisce il nome o l'indirizzo IP del computer (es. *ntde01.pd.infn.it*);

- l'attributo *Description*, che descrive il computer (es. *alpha station 3000/600, con sistema operativo Windows NT 4.0 Server*);

- l'attributo *Performances*, che indica le prestazioni del computer (è un valore numerico);

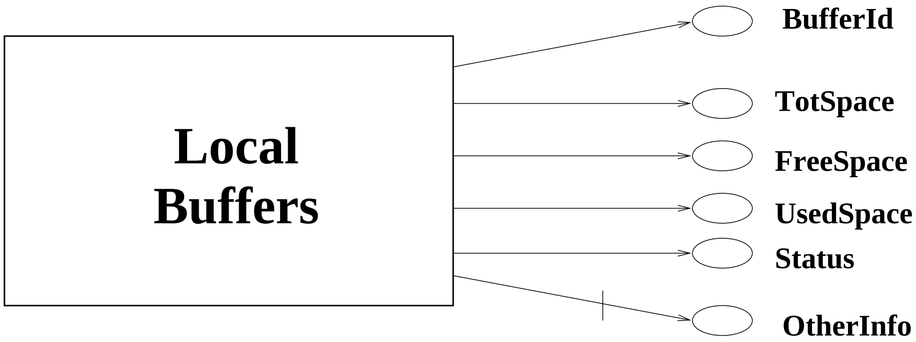

Figura 6.6: Classe 'Local Buffers'

- l'attributo *Status*, che indica se il computer è 'abilitato' o 'disabilitato' (e quindi se può "accettare" o meno job da eseguire).

6.3.5 Buffer

In figura 6.6 è descritta la classe 'Local Buffers', utilizzata per definire i disk buffer presenti nel sito locale, utilizzati dal sistema DEMI per memorizzare i file di dati (quindi i disk buffer definiti nel Local Buffer Database: vedi par. 4.3.2).
Gli attributi che permettono di definire lo stato di un oggetto di tipo Local Buffer sono:

- l'attributo *BufferId*, che rappresenta l'identificatore del disk buffer (es. *disk1*);
- l'attributo *TotSpace*, che definisce lo spazio totale di questo disk buffer assegnato per la memorizzazione di file del sistema DEMI (es. *2 Gbyte*);
- l'attributo *FreeSpace*, che indica lo spazio ancora disponibile in questo buffer (es. *540 Mbyte*);
- l'attributo *UsedSpace*, che indica lo spazio usato nel buffer per la memorizzazione di file (es. *1.3 Gbyte*).

  Il valore *TotSpace* - *FreeSpace* - *UsedSpace* rappresenta lo spazio "allocato" in questo disk buffer (infatti, prima che inizi l'esecuzione di un task, viene allocato lo spazio per la memorizzazione dei suoi output);
- l'attributo *Status*, che indica se il buffer è 'abilitato' o 'disabilitato' (e quindi se può essere utilizzato per la memorizzazione di file);
- l'attributo *OtherInfo*, che rappresenta eventuali altre informazioni relative al buffer.

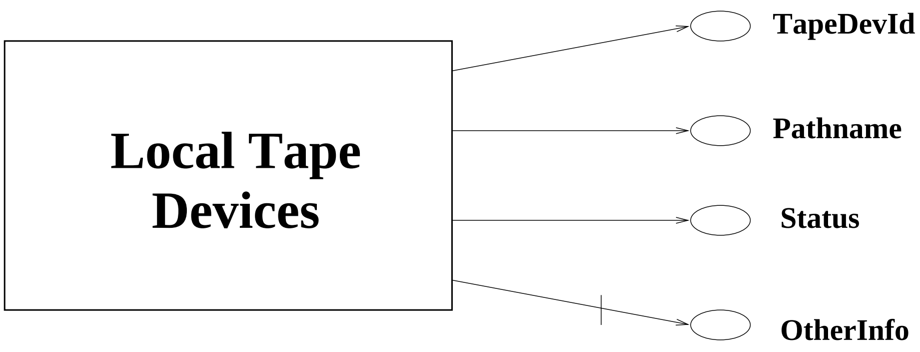

Figura 6.7: Classe 'Local Tape Devices'

6.3.6 Tape device

Per definire gli elementi del Local Tape Device Database, ovvero i tape device presenti nel sito locale, utilizzati dal sistema DEMI (vedi par. 4.3.3), si utilizza la classe 'Local Tape Devices', rappresentata in figura 6.7.
Per definire le proprietà di questi oggetti, si utilizzano i campi TapeDevId, Pathname, Status e OtherInfo.

- L'attributo *TapeDevId* rappresenta l'identificatore del tape device (es. *tapedev1*);
- l'attributo *Pathname* definisce il pathname che permette di "accedere" al tape device (es. */dev/rmt0h*);
- l'attributo *Status* indica se il tape device è 'abilitato' o 'disabilitato' (e quindi se può essere usato per la lettura/scrittura di dati da/su nastro);
- l'attributo *OtherInfo* rappresenta eventuali altre informazioni relative al tape device.

6.3.7 Utenti

In figura 6.8 è rappresentata la classe 'Users', utilizzata per definire gli utenti del sistema DEMI del sito locale, ovvero gli elementi definiti nello User Database (vedi par. 4.3.4).
La classe 'Users' è "costituita" dalle sottoclassi 'Local Administrators', 'Operators' e 'General Users'; si tratta di sottoclassi copertura, in quanto l'unione dei loro elementi coincide con gli elementi della classe 'Users', ma questi insiemi in generale non sono tra loro disgiunti: un utente può infatti avere più "ruoli" (ad esempio può essere allo stesso tempo un general user e un operator).

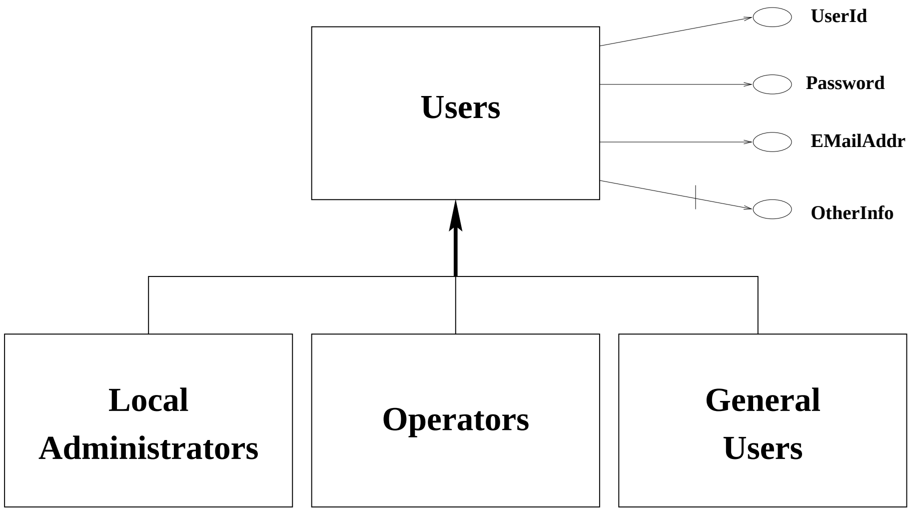

Figura 6.8: Classi 'Users', 'Local Administrators', 'Operators', 'General Users'

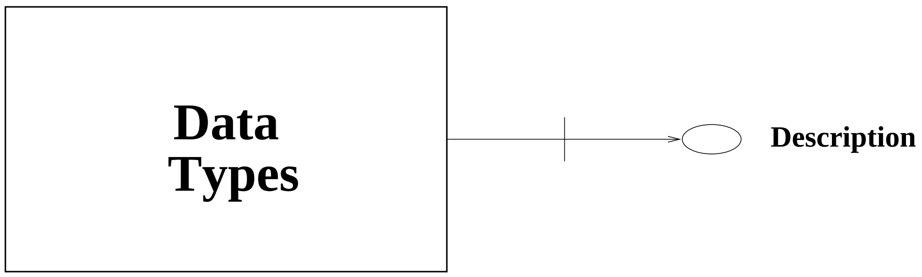

Figura 6.9: Classe 'Data Types'

- L'attributo *UserId* rappresenta lo User Identifier dell'utente (es. *rossi*);
- l'attributo *Password* rappresenta la password (criptata) dell'utente;
- l'attributo *EMailAddr* definisce l'e-mail address dell'utente (es. *rossi@local-domain*);
- l'attributo *OtherInfo* definisce eventuali altre informazioni relative all'utente.

6.3.8 Dati

In figura 6.9 è rappresentata la classe 'Data Types', che definisce i possibili tipi associati ai file di dati.

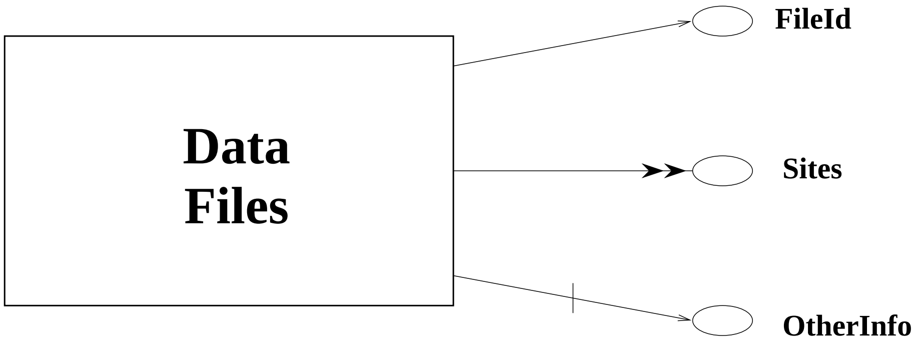

Figura 6.10: Classe 'Data Files'

Per definire lo stato di un oggetto di questa classe è sufficiente utilizzare l'attributo *Description*, che rappresenta la descrizione del tipo di dati (es. *QQPS, 183 GeV, 97_1 MC*).

I file di dati definiti nel Global Data Database (vedi par. 4.4.1) vengono rappresentati attraverso la classe 'Data Files', mostrata in figura 6.10.
I campi di un oggetto di questa classe sono:

- *FileId*: rappresenta l'identificatore del file di dati (es. *fileid1*);
- *Sites*: indica in che siti il file di dati è fisicamente presente: si tratta di un attributo multivalore, in quanto uno stesso file può essere presente in più siti. Poiché, come detto, in questa prima implementazione si considera un sistema composto da un solo sito, per il momento questo attributo non è significativo;
- *OtherInfo*: rappresenta eventuali altre informazioni relative al file di dati.

In figura 6.11 è rappresentata la classe 'Local Data Files', che definisce gli elementi memorizzati nel Local Data Database, ovvero i file di dati presenti nel sito locale considerato (vedi par. 4.4.2).
Lo stato di un oggetto di questo tipo viene definito dai seguenti attributi:

- l'attributo *FileId*, che rappresenta l'identificatore del file di dati (es. *fileid1*);
- l'attributo *Size*, che definisce la dimensione del file (es. *257 Mbyte*);
- l'attributo *OtherInfo*, che definisce eventuali altre informazioni.

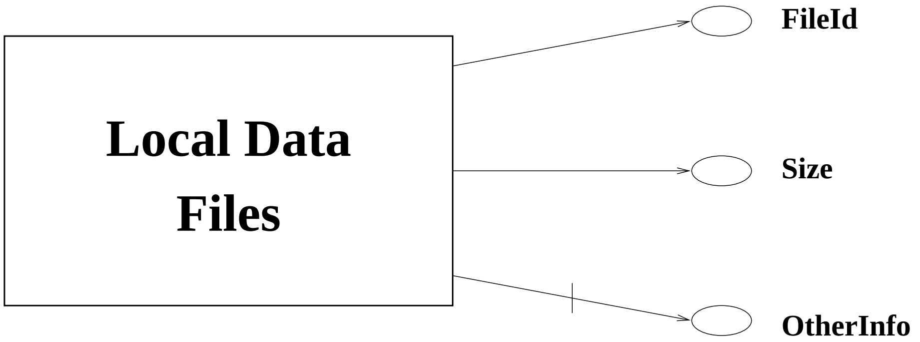

Figura 6.11: Classe 'Local Data Files'

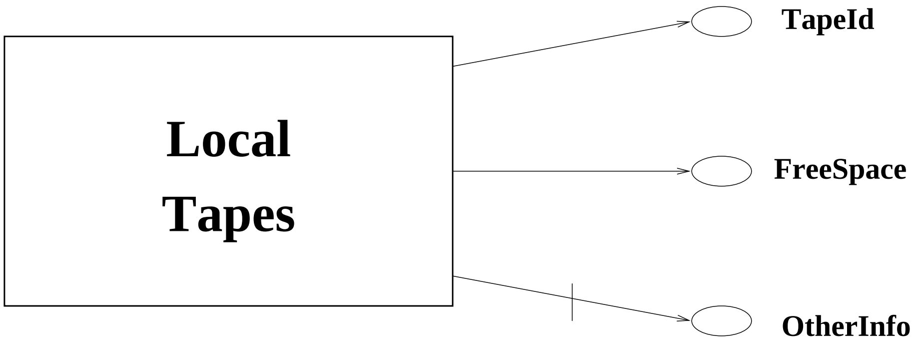

Figura 6.12: Classe 'Local Tapes'

La classe 'Local Tapes', rappresentata in figura 6.12, viene utilizzata per definire gli elementi del Local Tape Database (vedi par. 4.4.3), ovvero i nastri presenti nel sito locale, utilizzati per memorizzare file di dati.
Le proprietà di questi oggetti sono definiti da:

- l'attributo *TapeId*, che rappresenta l'identificatore del nastro (es. *tape1*);
- l'attributo *FreeSpace*, che indica quanto spazio è ancora disponibile sul nastro (es. *8 Gbyte*);
- l'attributo *OtherInfo*, che definisce eventuali altre informazioni.

6.3.9 Task

In figura 6.13 è rappresentata la classe 'Task Types', che definisce i tipi di task: ogni elemento di questa classe corrisponde ad un elemento del Task Database (vedi par. 4.5.1).

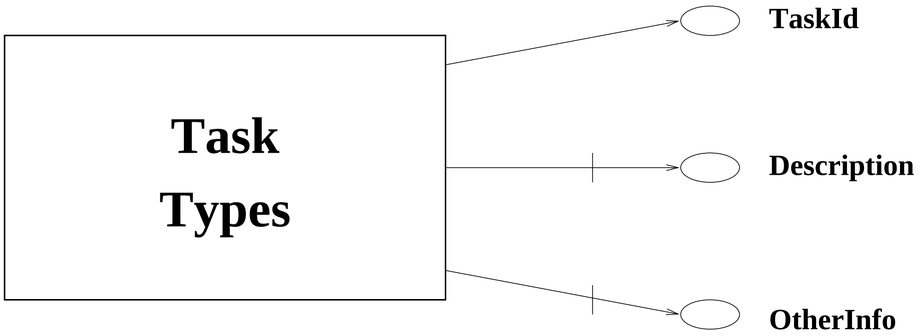

Figura 6.13: Classe 'Task Types'

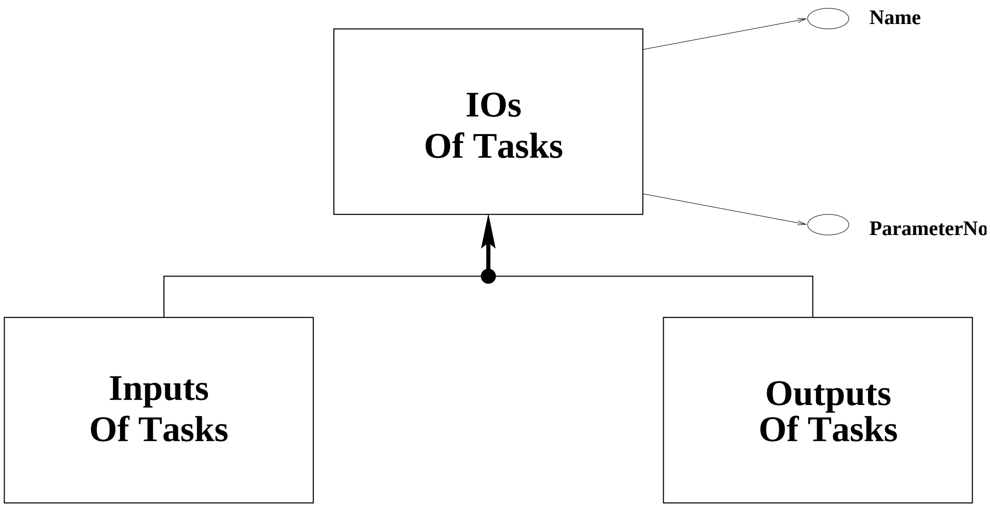

Figura 6.14: Classi 'IOs of tasks', 'Inputs Of Tasks', 'Outputs Of Tasks'

Gli attributi che vengono usati per definire lo stato di un oggetto di questa classe sono:

- *TaskId*, che rappresenta l'identificatore del tipo di task (es. *TaskTypeX*);
- *Description*, che descrive il tipo di task (es. *Task che gestisce l'attività X*);
- *OtherInfo*, che definisce eventuali altre informazioni.

In figura 6.14 è rappresentata la classe 'IOs Of Tasks', che definisce gli input-output dei tipi di task.

La classe 'IOs Of Tasks' è "costituita" dalle sottoclassi 'Inputs Of Tasks' e 'Outputs Of Tasks' (si tratta di una partizione).
Le proprietà di un oggetto di tipo 'IO Of Tasks' sono definite da:

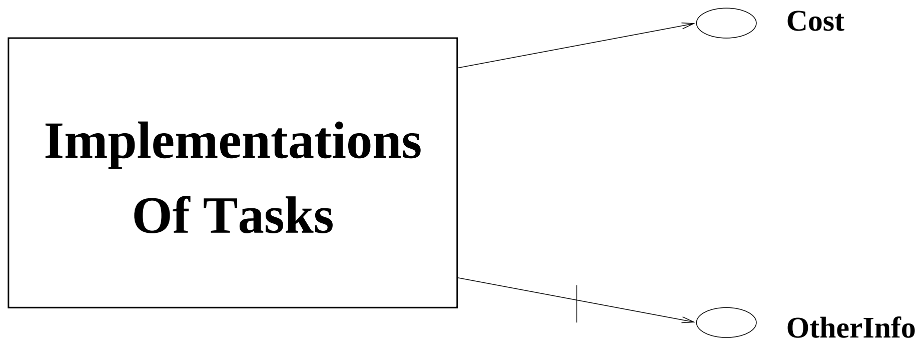

Figura 6.15: Classe 'Implementations Of Tasks'

- l'attributo *Name*, che rappresenta il nome dell'input/output considerato;
- l'attributo *ParameterNo*, che definisce il numero ordinale di parametro con cui si identifica questo input/output nella riga di comando che "invoca" l'esecuzione del corrispondente task. Infatti, come visto nel paragrafo 4.5.1, è necessario definire una corrispondenza tra input-output dei task, e i parametri che devono essere specificati nelle corrispondenti linee di comando.

L'implementazione di un task per una specifica architettura di calcolatori viene definita attraverso la classe 'Implementations Of Tasks', rappresentata in figura 6.15.
Gli attributi necessari per specificare le proprietà di questi oggetti sono:

- *Cost*: rappresenta il parametro 'Cost of the task';
- *OtherInfo*: definisce eventuali altre informazioni relative a questa implementazione.

In figura 6.16 è rappresentata la classe 'Files For Implementations', usata per definire i file necessari per l'esecuzione dei tipi di task, per le sue varie implementazioni considerate.

La classe 'Files For Implementations' ha come sottoclasse la classe 'Executable Files', usata per definire il file che deve essere indicato nella riga di comando specificata per l'esecuzione (vedi par. 4.5.1).

In generale per un'implementazione di un tipo di task per una certa architettura sono necessari vari file, uno dei quali, definito come 'eseguibile', viene specificato nella riga di comando che si utilizza quando si deve invocare l'esecuzione.

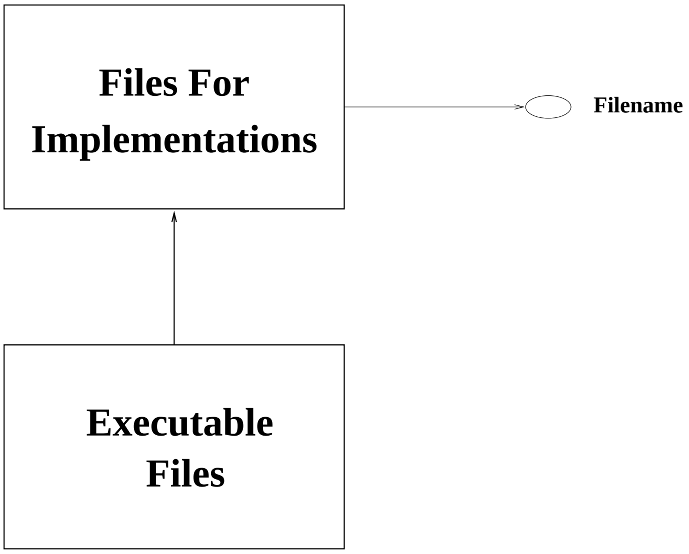

Figura 6.16: Classi 'Files For Implementations' e 'Executable Files'

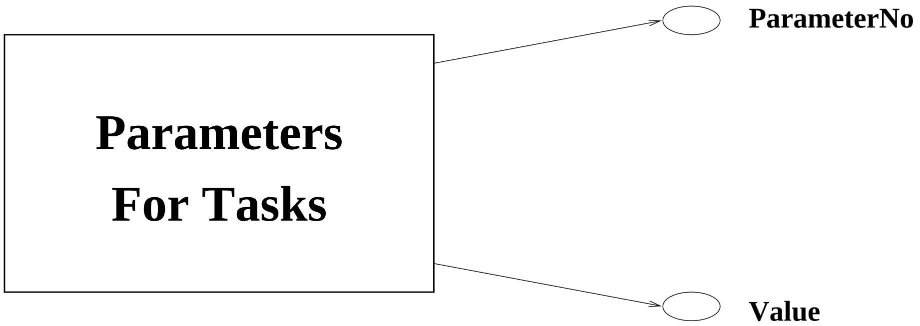

Figura 6.17: Classe 'Parameters For Tasks'

L'attributo *FileName* rappresenta il pathname del file.

Oltre ai parametri che definiscono gli input e gli output, quando si deve eseguire un task può essere necessario specificare nella corrispondente riga di comando altri parametri. Questi vengono definiti attraverso la classe 'Parameters For Tasks', rappresentata in figura 6.17.
Le proprietà degli oggetti di questo tipo sono definite dagli attributi:

- *ParameterNo*: rappresenta il numero ordinale del parametro (primo, secondo, ecc...);
- *Value*: specifica il valore associato al parametro.

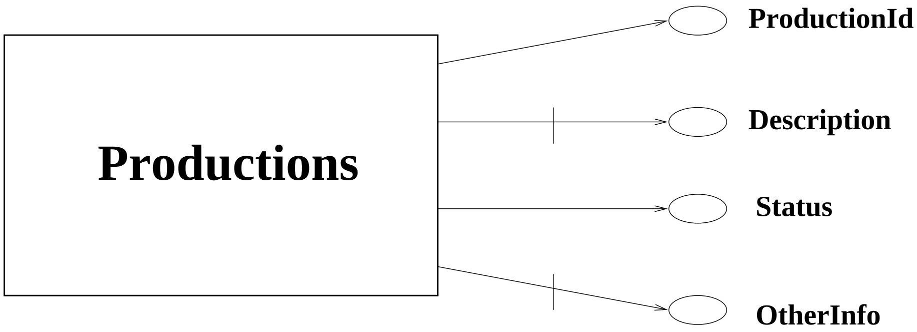

Figura 6.18: Classe 'Productions'

6.3.10 Production

In figura 6.18 è rappresentata la classe 'Productions', utilizzata per definire le produzioni; ogni oggetto di questa classe corrisponde ad un elemento del Production Database (vedi par. 4.5.2).
Per descrivere lo stato di un oggetto di tipo Production, si utilizzano gli attributi:

- *ProductionId*, che rappresenta l'identificatore della production (es. *ProdX*);
- *Description*, che descrive la produzione (es. *Production che gestisce l'elaborazione Y*);
- *Status*, che indica se la production può essere usata o meno per sottomettere nuove computational request;
- *OtherInfo*, che rappresenta eventuali altre informazioni relative alla produzione considerata.

Una produzione è costituita da più task: questi vengono rappresentati attraverso la classe 'Tasks Of Productions', descritta in figura 6.19.
L'attributo *Name* rappresenta il nome del task.

È necessario definire anche i data box delle produzioni. A tal fine si utilizza la classe 'Data Boxes Of Productions', rappresentata in figura 6.20.
La classe 'Data Boxes Of Productions' è "costituita" dalle sottoclassi 'Inputs Of Productions', 'Temporary Data Of Productions' e 'Outputs of productions' (si tratta di una partizione).
La classe 'Inputs of productions' è "costituita" dalle sottoclassi 'Singleton Inputs Of Productions' e 'Stream Inputs Of Productions' (una partizione).

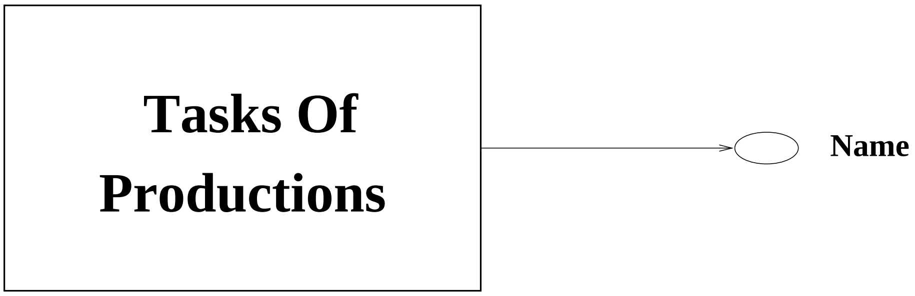

Figura 6.19: Classe 'Tasks Of Productions'

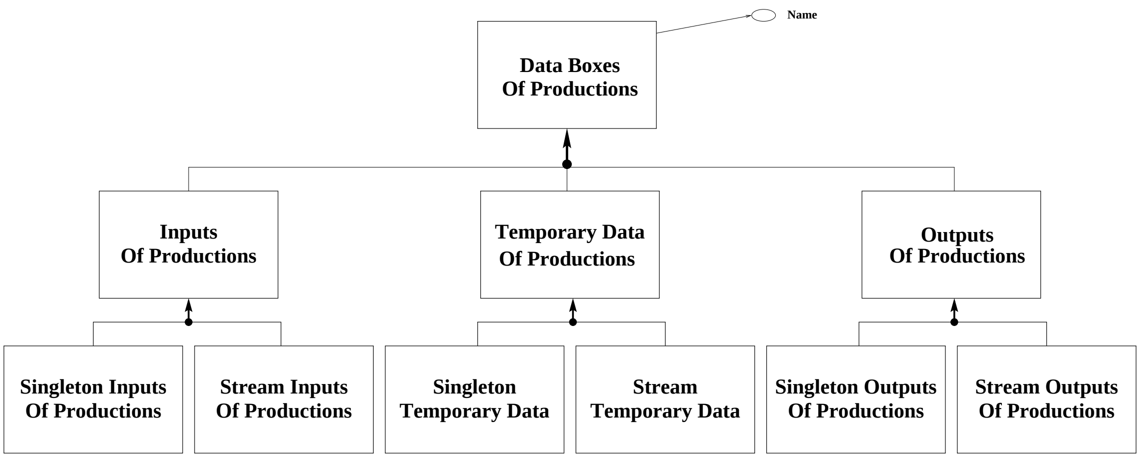

Figura 6.20: Classi 'Data Boxes Of Productions', 'Inputs Of Productions', 'Temporary Data Of Productions', 'Outputs Of Productions', 'Singleton Inputs Of Productions', 'Stream Inputs Of Productions', 'Singleton Temporary Data', 'Stream Temporary Data', 'Singleton Outputs Of Productions', 'Stream Outputs Of Productions'

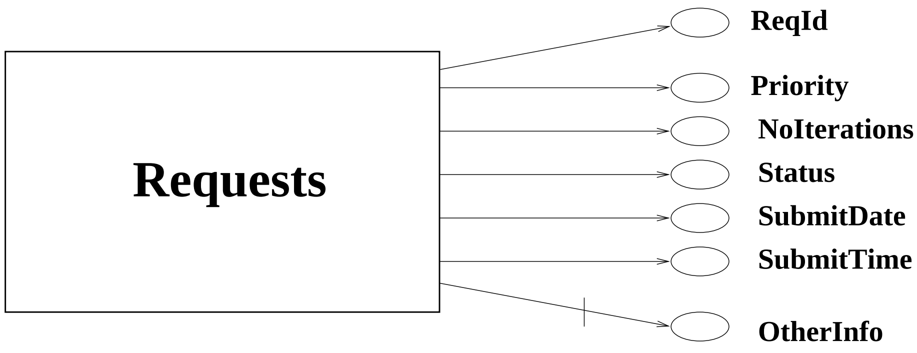

Figura 6.21: Classe 'Requests'

Una partizione analoga viene definita anche per le classi 'Temporary Data Of Productions' e 'Outputs Of Productions'.

I data box delle produzioni vengono quindi "divisi" in base al loro ruolo (input, output, temporanei), e poi vengono classificati in base al loro tipo (stream o singleton).

Il campo *Name* rappresenta il nome del data box.

6.3.11 Computational request

In figura 6.21 è rappresentata la classe 'Requests', usata per definire le computational request sottomesse al sistema DEMI dagli utenti. Ogni oggetto di questa classe corrisponde ad un elemento del Request Database (vedi par. 4.5.3).
Lo stato degli oggetti di tipo Request è definito dagli attributi:

- *ReqId*, che rappresenta l'identificatore della computational request;
- *Priority*, che indica il livello di priorità;
- *NoIterations*, che indica il numero di volte in cui deve essere eseguita la relativa production (nel caso in cui non ci siano input di tipo stream);
- *Status*, che indica lo stato di elaborazione della richiesta (soddisfatta, processing in corso, ecc...);
- *SubmitDate*, che rappresenta la data in cui la richiesta è stata sottomessa al sistema da parte di un utente;
- *SubmitTime*, che rappresenta l'ora in cui la richiesta è stata sottomessa al sistema DEMI;

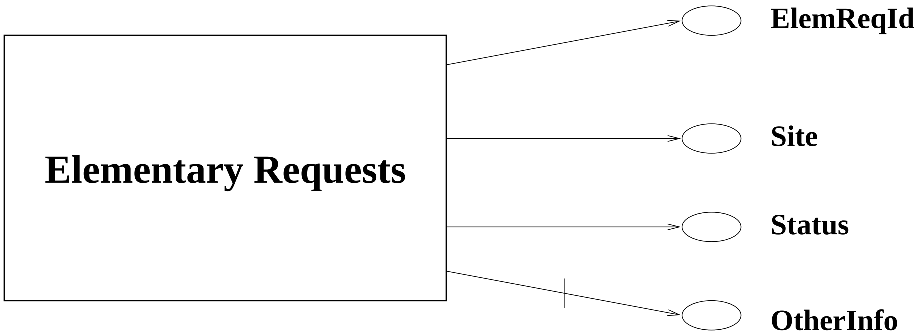

Figura 6.22: Classe 'Elementary requests'

- *OtherInfo*, che definisce eventuali altre informazioni.

6.3.12 Processing

Una computational request viene "scomposta" in elementary request, ognuna delle quali deve poi essere elaborata. Per la definizione di queste entità si utilizza la classe 'Elementary Requests', rappresentata in figura 6.22. Ogni elemento di questa classe corrisponde ad un elemento del Global Processing Database (vedi par. 4.8.1). Gli attributi usati per definire lo stato di questi oggetti sono:

- *ElemReqId*: rappresenta l'elementary request identifier (l'identificatore dell'elementary request);
- *Site*: indica in che sito l'elementary request è stata assegnata per la relativa elaborazione (considerando un ambiente costituito da un unico sito, questo attributo per il momento non è significativo);
- *Status*: indica lo stato dell'elaborazione di questa elementary request (in esecuzione, elaborazione completata con successo, processing fallito, ecc...);
- *OtherInfo*: definisce eventuali altre informazioni.

In figura 6.23 è rappresentata la classe 'Local Elementary Processings', utilizzata per definire le elaborazioni delle elementary request "assegnate" al sito locale (ovvero gli elementi del Local Processing Database: vedi par. 4.8.2).
Gli attributi da considerare sono:

- *ElemReqId*, che rappresenta l'elementary request identifier;

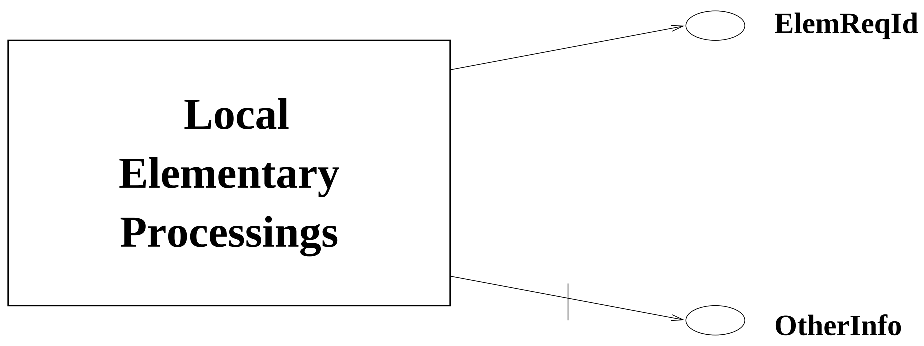

Figura 6.23: Classe 'Local Elementary Processings'

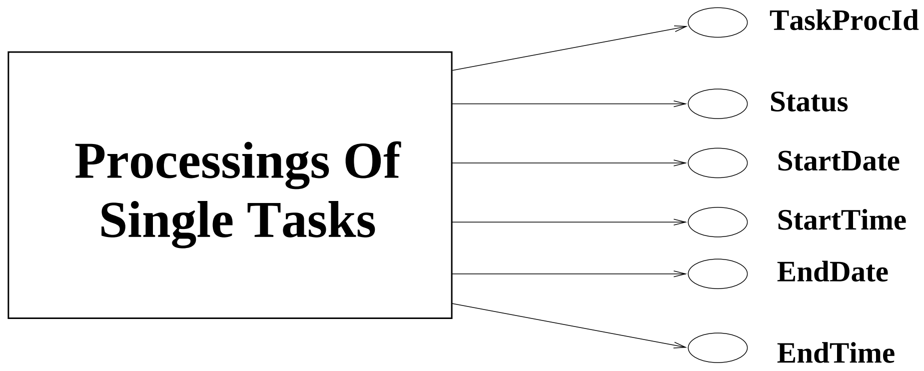

Figura 6.24: Classe 'Processings Of Single tasks'

- *OtherInfo*, che definisce eventuali altre informazioni.

Nel Processing Database è necessario "descrivere" anche le elaborazioni dei singoli task componenti la production, per ogni elementary request considerata. A tal fine si considera la classe 'Processings Of Single Tasks', rappresentata in figura 6.24. Gli attributi da considerare per la descrizione dello stato sono:

- l'attributo *TaskProcId*: rappresenta l'identificatore del processing;
- l'attributo *Status*: definisce lo stato dell'elaborazione del task (in esecuzione, elaborazione completata con successo, processing fallito, ecc...);
- l'attributo *StartDate*: indica la data in cui è iniziata l'esecuzione del task;
- l'attributo *StartTime*: indica l'ora in cui è iniziata l'elaborazione del task;
- l'attributo *EndDate*: indica la data in cui è terminata l'esecuzione del task;
- l'attributo *EndTime*: indica l'ora in cui è terminata l'elaborazione del task.

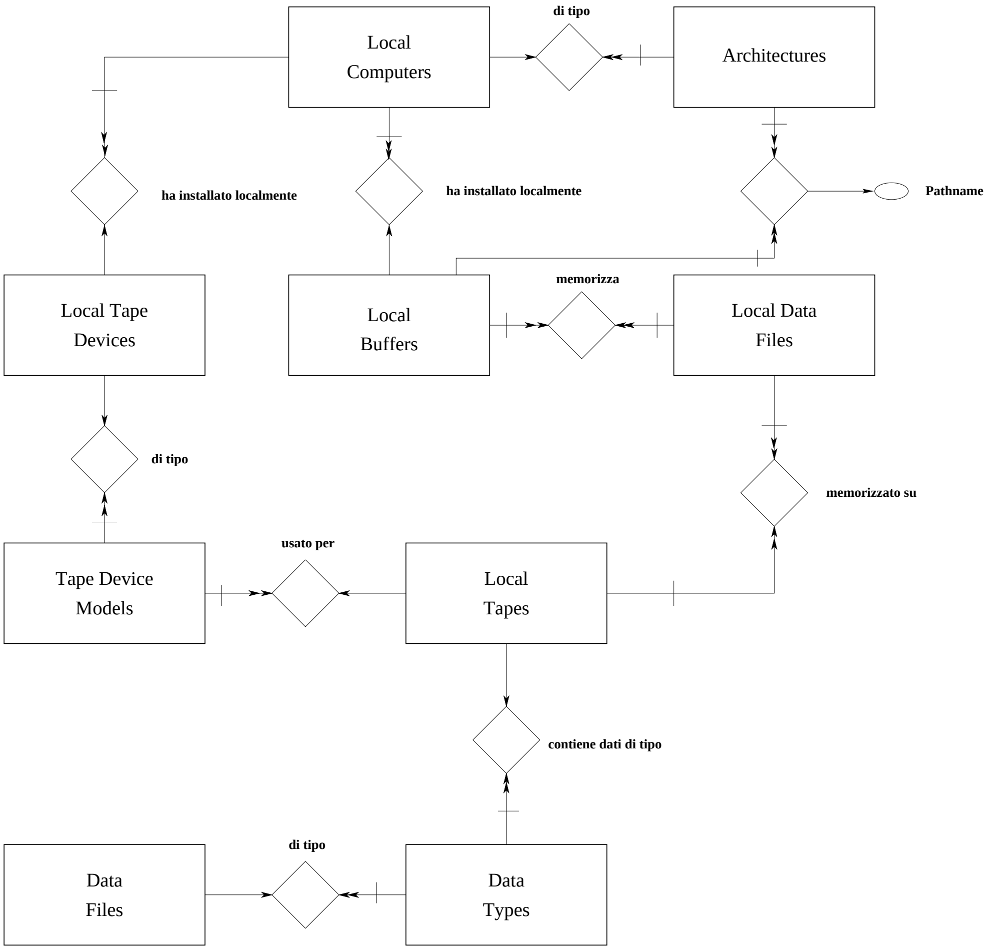

Figura 6.25: Schema della base di dati (parte I)

6.3.13 Associazioni

In questo paragrafo vengono definite le associazioni esistenti tra le varie classi prima definite.

In figura 6.25 sono rappresentate le associazioni tra le classi 'Local Computers', 'Architectures', 'Local Tape Devices', 'Local Buffers', 'Local Data Files', 'Tape Device Models', 'Local Tapes', 'Data Types' e 'Data Files'.

- Associazione tra 'Local Computers' e 'Architectures'.

  Si definisce che ogni computer è di una specifica architettura.

  La relazione diretta è univoca e totale (un computer è di un'unica architettura). La relazione inversa è multivalore (possono esserci più computer di una certa architettura) e parziale (potrebbero non esserci computer di una certa architettura).

- Associazione tra 'Local Computers' e 'Local Buffers'.

  Si specifica che i disk buffer sono fisicamente “installati” su computer.

  La relazione diretta è multivalore (su un computer possono essere installati più disk buffer) e parziale (su un computer potrebbero non essere presenti buffer). La relazione inversa è univoca e totale (un disk buffer è fisicamente installato su un unico computer).

- Associazione tra 'Local Buffers' e 'Architectures'.

  Questa associazione con attributo definisce che ad ogni buffer è associato un pathname, per ogni architettura (quindi tutti i computer della stessa architettura accedono al buffer con uno stesso pathname[1]).

- Associazione tra 'Local Computers' e 'Local Tape Devices'.

  La relazione indica che sui computer possono essere installati tape device.

  La relazione diretta è multivalore (su un computer possono essere installati più tape device) e parziale (su un computer potrebbero non essere installati tape device). La relazione inversa è univoca e totale (un tape device è fisicamente installato su un unico computer).

- Associazione tra 'Local Tape Devices' e 'Tape Device Models'.

  Si definisce che i tape device sono di specifici tipi.

  La relazione diretta è univoca e totale (un tape device è di un unico modello). La relazione inversa è multivalore (possono esserci più tape device di un certo modello) e parziale (potrebbero non esserci tape device di un certo modello).

- Associazione tra 'Local Buffers' e 'Local Data Files'.

  Si indica che i disk buffer vengono usati per memorizzare file di dati.

  La relazione diretta è multivalore (su un disk buffer possono essere memorizzati più file) e parziale (un disk buffer potrebbe essere “vuoto”, senza file memorizzati). La relazione inversa è multivalore (uno stesso file può essere memorizzato su più disk buffer) e parziale (un file potrebbe non essere memorizzato in nessun disk buffer: potrebbe essere presente solo su nastro).

[1] Sarebbe stato più corretto definire questa associazione tra le classi 'Local Buffers' e 'Local Computers', in quanto non necessariamente due computer della stessa architettura accedono ad uno stesso buffer con lo stesso pathname. In questo primo prototipo si è scelto questo approccio per semplificare la struttura.

- Associazione tra 'Local Data Files' e 'Local Tapes'.

  Si indica che i nastri vengono usati per memorizzare file di dati.

  La relazione diretta è multivalore (uno stesso file può essere memorizzato su più nastri) e parziale (un file potrebbe non essere memorizzato in nessun nastro: potrebbe essere presente solo su disco). La relazione inversa è multivalore (su un nastro possono essere memorizzati più file) e parziale (un nastro potrebbe essere "vuoto", senza file memorizzati).

- Associazione tra 'Local Tapes' e 'Tape Device Models'.

  La relazione specifica che ogni nastro è di un certo tipo (i modelli dei tape device corrispondono anche ai modelli dei relativi nastri).

  La relazione diretta è univoca e totale (un nastro è di un unico modello). La relazione inversa è multivalore (possono esserci più nastri di un certo modello) e parziale (potrebbero non esserci nastri di un certo modello).

- Associazione tra 'Local Tapes' e 'Data Types'.

  Si indica che su un nastro vengono memorizzati file di dati tutti di uno stesso tipo.

  La relazione diretta è univoca e totale (un nastro contiene dati di un unico tipo). La relazione inversa è multivalore (possono esserci più nastri che contengono dati di un certo tipo) e parziale (potrebbero non esserci nastri che contengono file di un certo tipo).

- Associazione tra 'Data Files' e 'Data Types'.

  Si definisce che ogni file di dati è di un certo tipo.

  La relazione diretta è univoca e totale (un file di dati è di un unico tipo). La relazione inversa è multivalore (possono esserci più file di un certo tipo) e parziale (potrebbero non esserci file di un certo tipo).

In figura 6.26 sono rappresentate le associazioni tra le classi 'General Users', 'Task Types', 'Inputs Of Tasks', 'Outputs Of Tasks', 'Architectures', 'Implementations Of Tasks', 'Parameters For Tasks', 'Files For Implementations' e 'Executable Files'.

- Associazione tra 'General Users' e 'Task Types'.

  Si indica che i general user possono definire tipi di task.

  La relazione diretta è multivalore (un general user può aver definito più tipi di task) e parziale (un general user potrebbe non aver definito nessun tipo

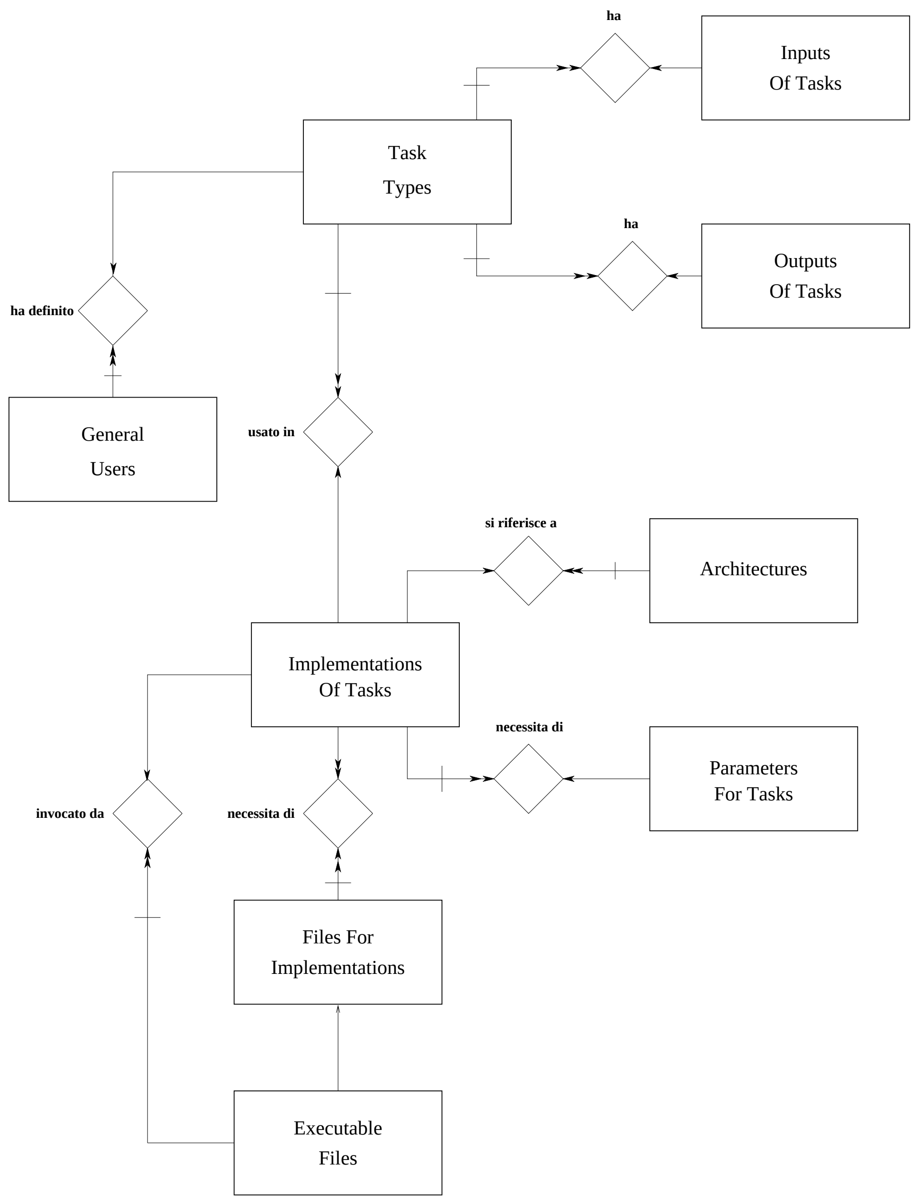

Figura 6.26: Schema della base di dati (parte II)

di task). La relazione inversa è univoca (un tipo di task è stato definito da un unico general user) e totale (un tipo di task ha sempre un utente che l'ha definito).

- Associazione tra 'Task Types' e 'Inputs Of Tasks'.

  Si specifica che un tipo di task può avere input.

  La relazione diretta è multivalore (un tipo di task può avere più input) e parziale (un tipo di task potrebbe non avere nessun input). La relazione inversa è univoca e totale (uno specifico input è associato ad un unico tipo di task).

- Associazione tra 'Task Types' e 'Outputs Of Tasks'.

  Si definisce che un tipo di task può avere output.

  La relazione diretta è multivalore (un tipo di task può avere più output) e parziale (un tipo di task potrebbe non avere output). La relazione inversa è univoca e totale (uno specifico output è associato ad un unico tipo di task).

- Associazione tra 'Task Types' e 'Implementations Of Tasks'.

  La relazione indica che i tipi di task possono essere implementati per diverse architetture.

  La relazione diretta è multivalore (ad un tipo di task possono essere associate più implementazioni) e parziale (un tipo di task potrebbe non essere stato implementato per nessuna architettura). La relazione inversa è univoca e totale (un'implementazione è associata ad un unico tipo di task).

- Associazione tra 'Implementations Of Tasks' e 'Architectures'.

  Si specifica che un'implementazione di un tipo di task si riferisce ad una specifica architettura di calcolatori.

  La relazione diretta è univoca e totale (ad una implementazione è associata un'unica architettura). La relazione inversa è multivalore (un'architettura potrebbe essere usata in più implementazioni) e parziale (un'architettura potrebbe non essere usata in nessuna implementazione).

- Associazione tra 'Implementations Of Tasks' e 'Files For Implementations'.

  La relazione rappresenta il fatto che, per eseguire un tipo di task, considerando una sua specifica implementazione per un'architettura, sono indispensabili dei file.

La relazione diretta è multivalore (per una implementazione possono essere necessari più file) e totale (per una implementazione è sempre necessario almeno un file). La relazione inversa è multivalore (uno stesso file potrebbe essere usato in diverse implementazioni) e parziale (un file potrebbe non essere usato in nessuna implementazione).

- Associazione tra 'Implementations Of Tasks' e 'Executable files'.

  Si specifica che per eseguire un task, considerando una sua specifica implementazione, deve essere invocato uno specifico programma eseguibile/script.

  La relazione diretta è univoca e totale (ad una implementazione è associato un unico file eseguibile). La relazione inversa è multivalore (un file eseguibile/script potrebbe essere usato in più implementazioni) e parziale (un file eseguibile potrebbe non essere usato in nessuna implementazione).

- Associazione tra 'Implementations Of Tasks' e 'Parameters For Tasks'.

  La relazione indica che per eseguire un task, considerando una sua specifica implementazione, può essere necessario specificare nella riga di comando alcuni parametri (oltre a quelli di input - output).

  La relazione diretta è multivalore (ad una implementazione di un tipo di task possono essere associati più parametri) e parziale (per una implementazione di un task potrebbe non essere necessario specificare parametri nella riga di comando, oltre ai parametri di input-output). La relazione inversa è univoca e totale (uno specifico parametro è associato ad un unica implementazione).

In figura 6.27 sono rappresentate le associazioni tra le classi 'General Users', 'Productions', 'Tasks Of Productions', 'Data Boxes Of Productions', 'IOs Of Tasks', 'Task Types'.

- Associazione tra 'General Users' e 'Productions'.

  Si indica che i general user possono definire produzioni.

  La relazione diretta è multivalore (un general user può aver definito più production) e parziale (un general user potrebbe non aver definito nessuna production). La relazione inversa è univoca (una production è definita da un unico general user) e totale (una production ha sempre un utente che l'ha definita).

- Associazione tra 'Productions' e 'Tasks Of Productions'.

  Si indica che le production sono costituite da task.

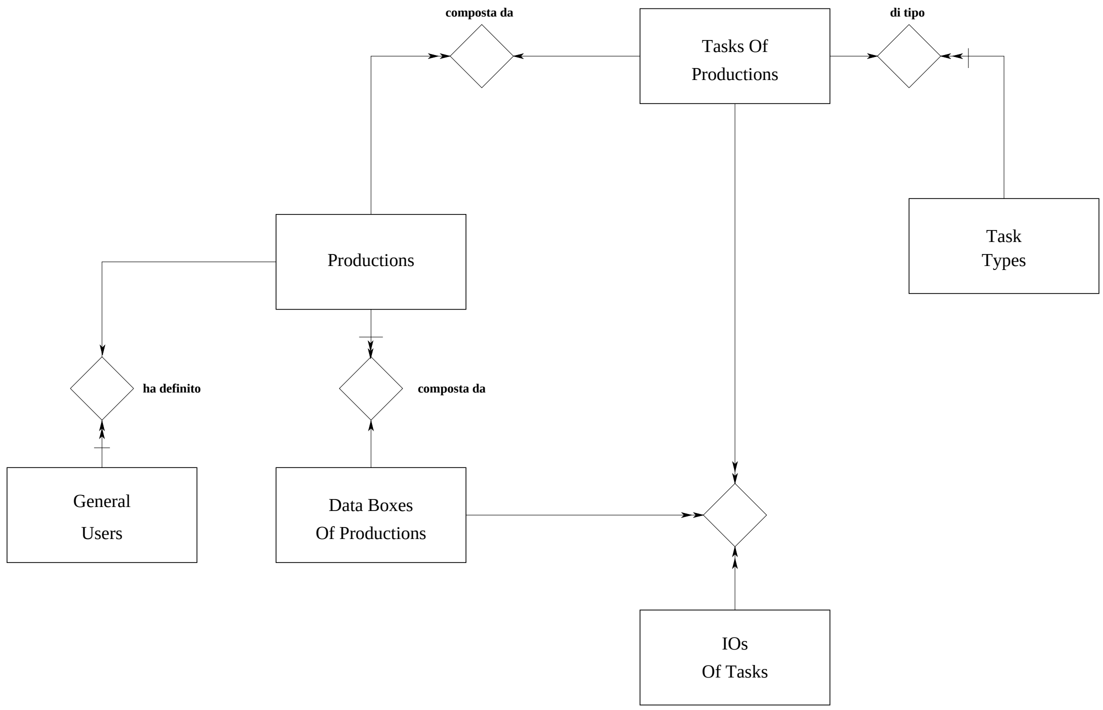

Figura 6.27: Schema della base di dati (parte III)

La relazione diretta è multivalore (una production può essere composta da più task) e totale (una production è composta da almeno un task). La relazione inversa è univoca e totale (uno specifico task di una production è associato ad un'unica production).

- Associazione tra 'Tasks Of Productions' e 'Task Types'.

  La relazione specifica che ogni task componente di una produzione è di un certo tipo.

  La relazione diretta è univoca e totale (un task è di un unico tipo). La relazione inversa è multivalore (ad un tipo di task possono essere associati più task componenti di production) e parziale (un tipo di task potrebbe non essere usato in nessun task componente di production).

- Associazione tra 'Productions' e 'Data Boxes Of Productions'.

  Si indica che nelle production si devono considerare i data box.

  La relazione diretta è multivalore (una production può essere composta da più data box) e parziale (in una production potrebbero non esserci data box). La relazione inversa è univoca e totale (uno specifico data box di una production è associato ad una singola production).

- Associazione tra 'Data Boxes Of Productions', 'Tasks Of Productions', 'IOs Of Tasks'.

  Questa associazione ternaria viene utilizzata per definire le relazioni tra i task che compongono le produzioni. Si definisce che ad ogni data box di una produzione corrispondono uno o più input/output dei task che la compongono.

  Come descritto in [6], per l'effettiva implementazione di questa associazione si può definire una quarta classe, e definire associazioni tra questa classe e, rispettivamente, le classi 'Data Boxes Of Productions', 'Tasks Of Productions' e 'IOs Of Tasks'.

In figura 6.28 sono rappresentate le associazioni tra le classi 'General Users', 'Requests', 'Productions', 'Data Boxes Of Productions', 'Outputs Of Productions', e 'Data Types'.

- Associazione tra 'General Users' e 'Requests'.

  Si indica che i general user possono sottomettere computational request.

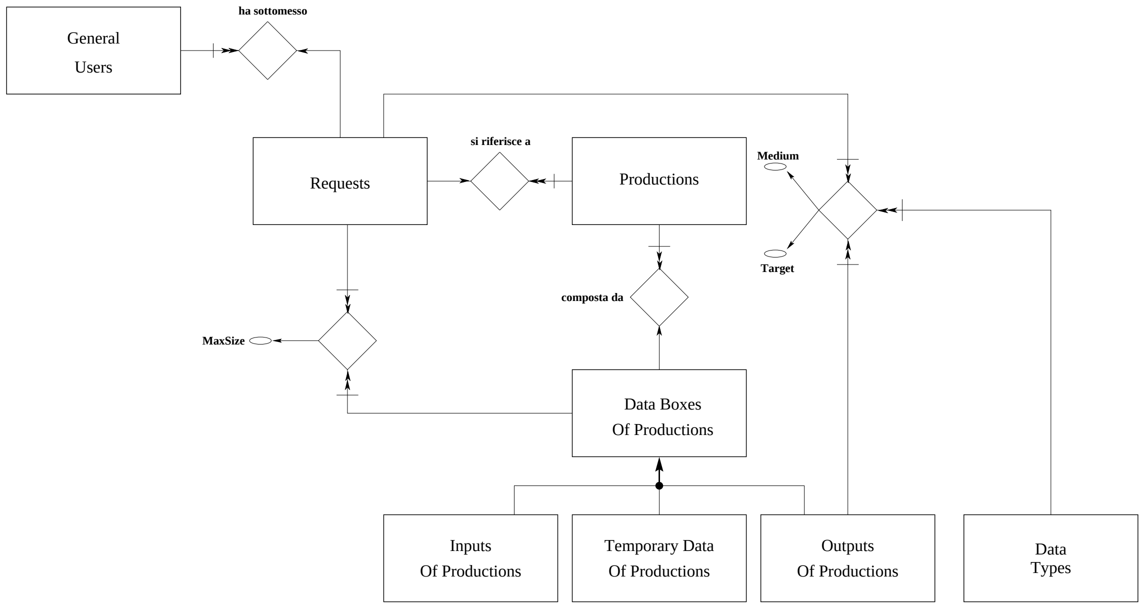

Figura 6.28: Schema della base di dati (parte IV)

La relazione diretta è multivalore (un general user può aver sottomesso più richieste) e parziale (un general user potrebbe non avere sottomesso nessuna richiesta). La relazione inversa è univoca e totale (una richiesta è sottomessa da un unico general user).

- Associazione tra 'Requests' e 'Productions'.

  La relazione indica che ogni richiesta fa riferimento ad una specifica produzione.

  La relazione diretta è univoca e totale (una computational request fa riferimento ad un'unica production). La relazione inversa è multivalore (una produzione può essere usata in più richieste) e parziale (una produzione potrebbe non essere stata usata in nessuna richiesta).

- Associazione tra 'Requests' e 'Data Boxes Of Productions'.

  Questa associazione con attributo indica che, per ogni richiesta, si devono "descrivere" i data box della corrispondente produzione (per definire la massima dimensione di un file associato al data box considerato, prodotto in questa richiesta).

  Per l'effettiva implementazione si potrà definire una nuova classe, con un attributo 'MaxSize', con associazioni tra questa classe e le classi 'Requests' e 'Data Boxes Of Productions'.

- Associazione tra 'Requests', 'Outputs Of Productions' e 'Data Types'.

  Con questa relazione ternaria si definisce che, per ogni richiesta, è necessario specificare il tipo dei file prodotti per ogni output della corrispondente produzione. L'attributo 'Medium' viene usato per indicare dove questi file devono essere memorizzati (su disk buffer, su nastro, o su una "directory privata"). L'attributo 'Target' assume valori diversi a seconda del valore dell'attributo 'Medium': se 'Medium' è pari a 'nastro', se ne definisce il corrispondente tipo; se invece 'Medium' specifica che i file di questo output devono essere memorizzati in una directory privata, questo attributo viene usato per definire il relativo pathname.

  Anche in questo caso per l'effettiva implementazione si può considerare un'ulteriore classe, con campi pari a 'Medium' e 'Target', in relazione con le altre tre classi.

In figura 6.29 sono rappresentate le associazioni tra le classi 'Stream Inputs Of Productions', 'Singleton Inputs Of Productions', 'Singleton Outputs Of Productions', 'Stream Outputs Of Productions', 'Data Files', 'Requests' e 'Elementary Requests'.

- Associazione tra 'Requests' e 'Elementary Requests'.

  La relazione rappresenta il fatto che le richieste vengono "scomposte" in elementary request.

  La relazione diretta è multivalore (una richiesta può essere scomposta in più elementary request) e parziale. La relazione inversa è univoca e totale (una elementary request fa riferimento ad un'unica richiesta).

- Associazione tra 'Singleton Inputs Of Productions', 'Requests' e 'Data Files'.

  Con questa associazione ternaria si definisce che per ogni richiesta, per ogni input di tipo singleton della corrispondente produzione, è associato un file (un singleton input file).

  Anche per questa relazione ternaria, e per quelle descritte in seguito, valgono i discorsi già fatti, ovvero l'introduzione di una quarta classe per l'effettiva implementazione della relazione ternaria.

- Associazione tra 'Stream Inputs Of Productions', 'Elementary Requests' e 'Data Files'.

  Questa associazione ternaria indica che per ogni elementary request, all'eventuale input di tipo stream della corrispondente production, è associato un file di dati (uno stream input file), che deve essere elaborato.

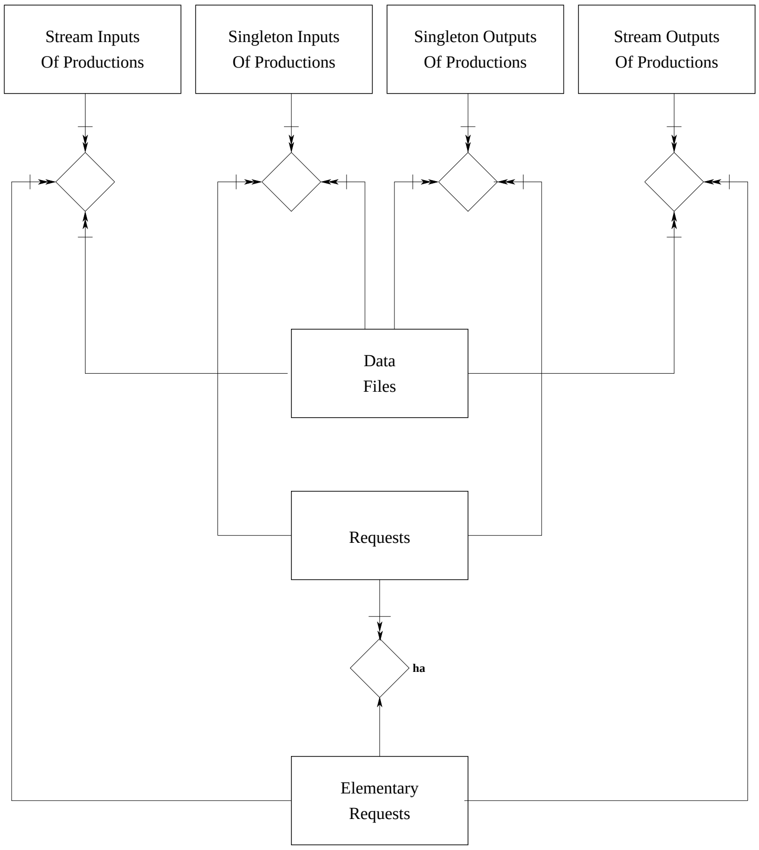

Figura 6.29: Schema della base di dati (parte V)

- Associazione tra 'Singleton Outputs Of Productions', 'Requests' e 'Data Files'.

  Questa relazione ternaria definisce che per ogni richiesta, per ogni output di tipo singleton della corrispondente produzione viene prodotto un file (un singleton output file).

- Associazione tra 'Stream Outputs Of Productions', 'Elementary Requests' e 'Data Files'.

  Questa associazione specifica che per ogni elementary request, per ogni output di tipo stream della corrispondente production viene generato un file di dati (uno stream output file).

In figura 6.30 sono rappresentate le associazioni tra le classi 'Local Elementary Processings', 'Processings Of Single Tasks', 'Tasks Of Productions', e 'Local Computers'.

- Associazione tra 'Local Elementary Processings' e 'Processings Of Single Tasks'.

  La relazione indica che ad ogni elementary request processata localmente sono associati i processing dei task che compongono la corrispondente production.

  La relazione diretta è multivalore (ad un elementary processing possono essere associati più processing di singoli task, nel caso in cui la corrispondente production sia costituita da più task o anche, in presenza di un singolo task, se è stato necessario elaborarlo più volte) e parziale (potrebbero non esserci processing di singoli task associati ad un elementary processing, ad esempio perché non si è ancora elaborato nessun task componente la relativa produzione). La relazione inversa è univoca e totale (un processing di un singolo task è associato ad un singolo elementary processing).

- Associazione tra 'Processings Of Single Tasks' e 'Tasks Of Productions'.

  Si indica che, per ogni elementary request, i task componenti la produzione vengono elaborati.

  La relazione diretta è univoca e totale (un processing è associato ad un singolo task). La relazione inversa è multivalore (ad un task componente di una production possono essere associati più processing) e parziale (potrebbero non esserci processing per un task componente di una production).

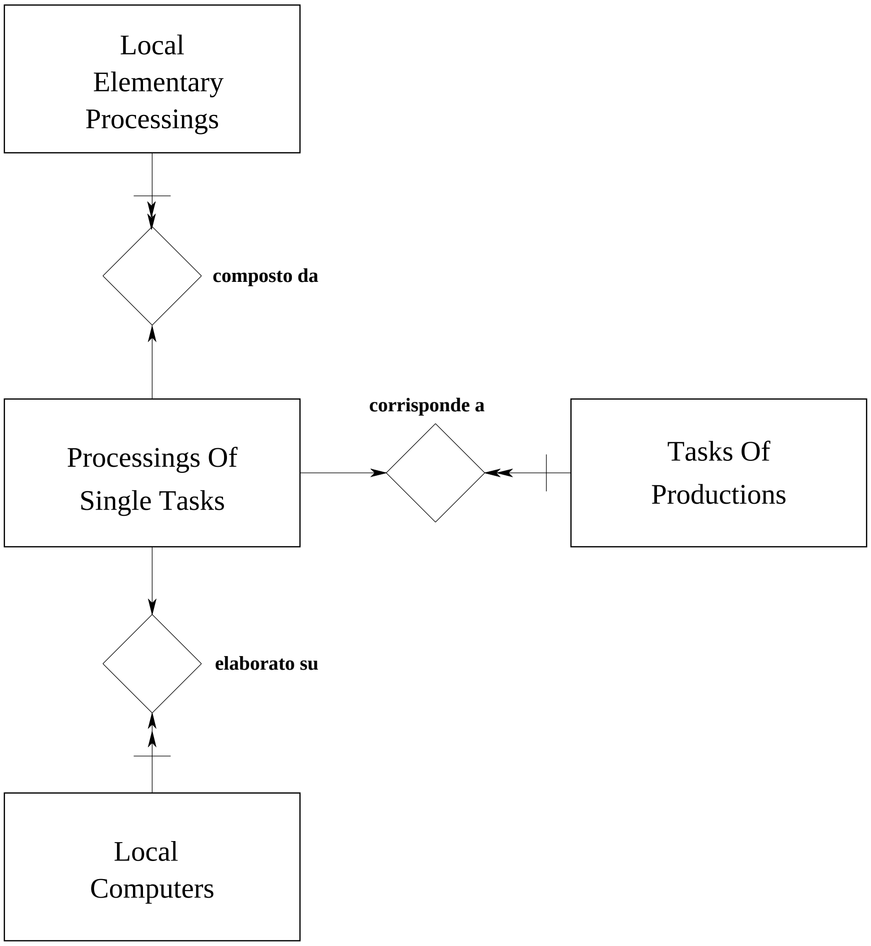

Figura 6.30: Schema della base di dati (parte VI)

- Associazione tra 'Processings Of Single Tasks' e 'Local Computers'.

  La relazione rappresenta il fatto che il processing di un task è eseguito su un computer.

  La relazione diretta è univoca e totale (il processing di un task è fatto su un singolo computer). La relazione inversa è multivalore (un computer può avere elaborato più task) e parziale (un computer potrebbe non avere processato nessun task).

6.4 Rappresentazione con il modello relazionale

Come descritto nel capitolo 8, nell'implementazione del primo prototipo si è utilizzato un database relazione. Lo schema della base di dati rappresentato con i costrutti del modello a oggetti è stato quindi trasformato in uno schema relazionale.

Sostanzialmente per rappresentare una classe nel modello relazionale si considera una relazione in cui si riportano gli attributi della classe considerata. Si considerano poi le seguenti regole [6]:

- un'associazione tra una classe A e una classe B, con diretta multivalore e inversa univoca, viene definita nel modello relazionale attraverso gli schemi R_A e R_B, con R_A associato alla classe A, e R_B associato alla classe B. Agli attributi di R_B si aggiunge una chiave esterna che riferisce la relazione R_A;

- per un'associazione tra una classe A e una classe B, con diretta e inversa univoca, la chiave esterna si aggiunge ad una qualunque delle relazioni, preferendo quella rispetto a cui l'associazione è totale;

- un'associazione tra una classe A e una classe B, con diretta e inversa multivalore, viene definita nel modello relazionale attraverso le relazioni R_A (associata alla classe A), R_B (associata alla classe B), e da una terza relazione, che contiene due chiavi esterne che riferiscono le due relazioni coinvolte;

- per rappresentare un'associazione tra n classi, si considera un'ulteriore relazione che contiene le chiavi esterne.

Per questi due ultimi casi considerati, se l'associazione ha degli attributi, questi vengono aggiunti alla nuova relazione.

Vengono di seguito descritti gli schemi di relazione della base di dati per il sistema DEMI.

Attributo	Tipo	Descrizione
ArchitectId	String	L'identificatore dell'architettura
Description	String	La descrizione dell'architettura
OtherInfo	String	Altre eventuali informazioni

Tabella 6.1: Schema di relazione 'Architectures'

ArchitectId	Description	OtherInfo
win32	Computer con processore Intel OS: Windows 32 bit	Windows NT e Windows 95
axp-win32	Computer con processore Alpha OS: Windows 32 bit	Windows NT e Windows 95
axp-du	Computer con processore Alpha OS: Digital Unix	
hp-ux	Workstation HP OS: HP-UX 10.20	
...	...	...

Tabella 6.2: Esempio di relazione

6.4.1 Architetture

La tabella 6.1 definisce lo schema di relazione *Architectures*, usato per rappresentare la classe 'Architectures' del modello a oggetti.

La chiave primaria di questa tabella è costituita dall'attributo *ArchitectId*.

In tabella 6.2 è mostrato un esempio di istanza di questo schema.

6.4.2 Modelli di tape device

Per definire la classe 'Tape Device Models' del modello a oggetti si considera lo schema di relazione *TapeDeviceModels*, rappresentato in tabella 6.3.

La chiave primaria è rappresentata dall'attributo *TapeDevModId*.

Un esempio di relazione è riportato in tabella 6.4.

6.4.3 Global administrator

La tabella 6.5 definisce lo schema di relazione *GlobalAdministrators*, utilizzato per rappresentare la classe 'Global Administrators' del modello a oggetti.

Attributo	Tipo	Descrizione
TapeDevModId	String	L'identificatore del modello di tape device
Description	String	La descrizione del modello di tape device
Capacity	Real	La capacità (espressa ad esempio in MB) di un nastro di questo modello
OtherInfo	String	Altre eventuali informazioni riguardanti il modello di tape device

Tabella 6.3: Schema di relazione 'TapeDeviceModels'

TapeDevModId	Description	Capacity	OtherInfo
dlt	DLT, per nastri TZ87	30000	
exab	Exabyte	3500	
...	...	...	...

Tabella 6.4: Esempio di relazione

Attributo	Tipo	Descrizione
UserId	String	L'identificatore del global administrator
Password	String	La password (criptata) del global administrator
EMailAddr	String	L'e-mail address del global administrator
OtherInfo	String	Altre eventuali informazioni riguardanti il global administrator

Tabella 6.5: Schema di relazione 'GlobalAdministrators'

UserId	Password	EMailAddr	OtherInfo
globadm	d51e@qeg78s	globadm@domain	Default global administrator
sgaravat	d3+egk*y3s	sgaravat@pd.infn.it	Massimo Sgaravatto Tel. ++39 49 8277029
rossi	3efgy3s	rossi@domain	Mario Rossi Tel. ++39 49 123456
...	...	...	...

Tabella 6.6: Esempio di relazione

Attributo	Tipo	Descrizione
CompName	String	Il nome o l'indirizzo IP del computer
Description	String	La descrizione del computer
ArchitectId	String	L'identificatore che specifica l'architettura del computer
Perform	Real	Il parametro che definisce le prestazioni del computer
Status	Boolean	Indica se il computer è 'abilitato' o 'disabilitato'

Tabella 6.7: Schema di relazione 'LocalComputers'

La chiave primaria di questo schema di relazione è rappresentata dall'attributo *UserId*.

Un esempio di relazione è riportato in tabella 6.6.

6.4.4 Computer

Per definire la classe 'Local Computers' del modello a oggetti si considera uno specifico schema di relazione, rappresentato in tabella 6.7, chiamato *LocalComputers*.

La chiave primaria è costituita dall'attributo *CompName*.

Con l'attributo *ArchitectId* si rappresenta l'associazione tra le classi 'Local Computers' e 'Architectures'.

Un'istanza di relazione relativa a questo schema è mostrata in tabella 6.8.

CompName	Description	ArchitectId	Perform	Status
osde00	Alpha Server 1000	axp-du	100	Enabled
hpcms01		hp-ux	105	Enabled
pccms1	Pentium Pro 200	win32	75	Enabled
...	...	...	...	...

Tabella 6.8: Esempio di relazione

Attributo	Tipo	Descrizione
BufferId	String	L'identificatore del disk buffer
CompName	String	Il nome del computer dove il disco è installato
TotSp	Real	Lo spazio totale (espresso ad esempio in MB) assegnato per la memorizzazione di file DEMI
UsSp	Real	Lo spazio (espresso ad esempio in MB) già usato in questo buffer
FreeSp	Real	Lo spazio (espresso ad esempio in MB) ancora disponibile in questo buffer
Status	Boolean	Indica se il buffer è 'abilitato' o 'disabilitato'
OtherInfo	String	Altre eventuali informazioni

Tabella 6.9: Schema di relazione 'LocalBuffers'

6.4.5 Buffer

Lo schema di relazione *LocalBuffers*, rappresentato in tabella 6.9, definisce la classe 'Local Buffers' del modello a oggetti.

La chiave primaria di questo schema di relazione è costituita dall'attributo *BufferId*.

L'attributo *CompName* è la chiave esterna che permette di definire l'associazione tra le classi 'Local Computers' e 'Local Buffers'.

La tabella 6.10 mostra un esempio di relazione.

In tabella 6.11 è rappresentato lo schema di relazione *BufferPathnames*, usato per definire l'associazione con attributo tra le classi 'Local Buffers' e 'Architectures'; un esempio di istanza di questo schema è mostrato in tabella 6.12.

BufferId	CompName	TotSp	UsSp	FreeSp	Status	OtherInfo
buffer1	osde00	1000	570	300	Enabl	
buffer2	osde00	2000	1300	700	Enabl	
buffer3	hpcms01	500	300	113	Enabl	
...	...	...	...	...	...	...

Tabella 6.10: Esempio di relazione

Attributo	Tipo	Descrizione
BufferId	String	L'identificatore del disk buffer
ArchitectId	String	L'identificatore dell'architettura
PathName	String	Il pathname del buffer

Tabella 6.11: Schema di relazione 'BufferPathnames'

BufferId	ArchitectId	Pathname
buffer1	axp-du	/DEMI/Buffer1
buffer1	win32	%DEMIBUFFER1%
buffer2	axp-du	/DEMI/Buffer1
buffer2	win32	%DEMIBUFFER2%
...	...	...

Tabella 6.12: Esempio di relazione

Attributo	Tipo	Descrizione
TapeDevId	String	L'identificatore del tape device
TapeDevModId	String	L'identificatore del modello di tape device
CompName	String	Il nome del computer dove il tape device è installato
PathName	String	Il pathname con cui si "accede" al tape device
Status	Boolean	Indica se il tape device è 'abilitato' o 'disabilitato'
OtherInfo	String	Altre eventuali informazioni

Tabella 6.13: Schema di relazione 'LocalTapeDevices'

TapeDevId	TapeDevModId	CompName	PathName	Status	OtherInfo
tapedev1	dlt	osde00	/dev/rmt0h	Enabled	
tapedev2	dlt	hpcms01	/dev/rmt0h	Enabled	
tapedev3	exab	hpcms01	/dev/rmt1h	Enabled	
...	...	...	...	...	...

Tabella 6.14: Esempio di relazione

6.4.6 Tape Device

In tabella 6.13 è definito lo schema di relazione *LocalTapeDevices*, che permette di rappresentare la corrispondente classe del modello a oggetti.

La chiave primaria è definita dall'attributo *TapeDevId*.

L'attributo *TapeDevModId* permette di definire l'associazione tra le classi 'Local Tape Devices' e 'Tape Device Models', mentre con l'attributo *CompName* si specifica la relazione tra le classi 'Local Tape Devices' e 'Local Computers'.

La tabella 6.14 rappresenta un esempio di relazione.

6.4.7 Utenti

Per definire la classe 'Users' e le sue sottoclassi 'Local Administrators', 'Operators' e 'General Users', si considera lo schema di relazione *Users*, rappresentato in tabella 6.15.

La chiave primaria di questo schema è costituita dall'attributo *UserId*.

Un esempio di relazione, relativo a questo schema, è riportata in figura 6.16.

Attributo	Tipo	Descrizione
UserId	String	Lo user identifier dell'utente
Password	String	La password (criptata) dell'utente
EMailAddr	String	L' e-mail address dell'utente
LocAdm	Boolean	Definisce se l'utente è un local administrator
Operat	Boolean	Specifica se l'utente è un operator
GenUser	Boolean	Definisce se l'utente è un general user
OtherInfo	String	Altre eventuali informazioni riguardanti l'utente

Tabella 6.15: Schema di relazione 'Users'

UserId	Password	EMailAddr	LocAdm	Operat	GenUser	OtherInfo
locadm	wdd@t43	locad@domain	Yes	No	No	Default Local Administrator
operator	w!45f67u	oper@domain	No	Yes	No	Default Operator
rossi	3efg*y3s	rossi@domain	No	Yes	Yes	Mario Rossi Tel. 123456
neri	326g)yds	neri@domain	No	No	Yes	Luca Neri Tel. 314436
...	...	...	...	...	...	...

Tabella 6.16: Esempio di relazione

Attributo	Tipo	Descrizione
FileId	String	L'identificatore del file
Description	String	La descrizione del tipo di dati rappresentati dal file
OtherInfo	String	Altre eventuali informazioni

Tabella 6.17: Schema di relazione 'DataFiles'

FileId	Description	OtherInfo
id00001	QQPS 183 GeV DSIM97-1	
id00002	BAFO 172 GeV DSIM96-1	
id00003	QQPS 183 GeV DSIM97-1	
...	...	...

Tabella 6.18: Esempio di relazione

6.4.8 Dati

Lo schema di relazione *DataFiles*, rappresentato in tabella 6.17, definisce la corrispondente classe del modello a oggetti.

La chiave primaria è costituita dall'attributo *FileId*.

L'attributo *Description* è la chiave esterna che permette di definire l'associazione tra le classi 'Data Files' e 'Data Types'.

Un esempio di istanza dello schema è mostrato in tabella 6.18.

Lo schema di relazione *LocalDataFiles*, mostrato in tabella 6.19, permette di rappresentare la classe 'Local Data Files'.

La chiave primaria di questo schema di relazione è definita dall'attributo *FileId*.

Un esempio di relazione, considerando questo schema, è rappresentato in tabella 6.20.

Attributo	Tipo	Descrizione
FileId	String	L'identificatore del file
Size	Real	La dimensione (espressa ad esempio in MB) del file
OtherInfo	String	Altre eventuali informazioni

Tabella 6.19: Schema di relazione 'LocalDataFiles'

FileId	Size	OtherInfo
id00001	100	
id00002	135	
...	...	...

Tabella 6.20: Esempio di relazione

Attributo	Tipo	Descrizione
TapeId	String	L'identificatore del nastro
FreeSpac	Real	Lo spazio ancora disponibile sul nastro
TapeDevModId	String	Il modello del nastro
Description	String	Il tipo dei file memorizzati nel nastro
OtherInfo	String	Altre eventuali informazioni

Tabella 6.21: Schema di relazione 'LocalTapes'

In tabella 6.21 è mostrato lo schema di relazione *LocalTapes*, con cui si definisce la relativa classe del modello a oggetti.

La chiave primaria è costituita dall'attributo *TapeId*.

L'attributo *TapeDevModId* permette di definire l'associazione tra le classi 'Local Tapes' e 'Tape Device Models'. Per rappresentare la relazione tra le classi 'Local Tapes' e 'Data Types', si usa invece l'attributo *Description*.

La tabella 6.22 mostra un esempio di relazione.

Lo schema di relazione *FilesOnBuffers*, definito in tabella 6.23, è utilizzato per definire l'associazione tra le classi 'Local Buffers' e 'Local Data Files' del modello a oggetti.

Un esempio di relazione, con riferimento a questo schema, è rappresentato in tabella 6.24.

TapeId	FreeSpace	TapeDevModId	Description	OtherInfo
tapeid1	900	dlt	QQPS 183 GeV DSIM97-1	
tapeid2	1200	exab	BAFO 172 GeV DSIM96-1	
tapeid3	2680	dlt	QQPS 183 GeV DSIM97-1	
...	...	...	...	...

Tabella 6.22: Esempio di relazione

Attributo	Tipo	Descrizione
FileId	String	L'identificatore del file
BufferId	String	L'identificatore del disk buffer

Tabella 6.23: Schema di relazione 'FilesOnBuffers'

FileId	BufferId
id00001	buffer1
id00001	buffer2
id00002	buffer1
...	...

Tabella 6.24: Esempio di relazione

Per definire la relazione tra la classe 'Local Data Files' e 'Local Tapes', si specifica uno schema di relazione chiamato *FilesOnTapes*, rappresentato in tabella 6.25.

In tabella 6.26 è presentato un esempio di istanza di relazione.

6.4.9 Task

Lo schema *TaskTypes*, rappresentato in tabella 6.27, permette di definire la corrispondente classe del modello a oggetti.

La chiave primaria di questo schema è definita dall'attributo *TaskId*.

L'attributo *UserId* è la chiave esterna che definisce la relazione tra le classi 'Task Types' e 'General Users'.

Un esempio di istanza, corrispondente a questo schema di relazione, è mostrato in tabella 6.28.

In tabella 6.29 è definito lo schema di relazione *IOsOfTasks*, usato per rappresentare la classe 'IOs Of Tasks', e le relative sottoclassi.

La chiave primaria di questo schema è costituita dall'attributo *IOId*.

Attributo	Tipo	Descrizione
FileId	String	L'identificatore del file
TapeId	String	L'identificatore del nastro

Tabella 6.25: Schema di relazione 'FilesOnTapes'

FileId	TapeId
id00001	tapeid1
id00001	tapeid2
id00002	tapeid1
...	...

Tabella 6.26: Esempio di relazione

Attributo	Tipo	Descrizione
TaskId	String	L'identificatore del tipo di task
Description	String	La descrizione del tipo di task
UserId	String	L'identificatore dell'utente che ha definito questo tipo di task
OtherInfo	String	Altre eventuali informazioni

Tabella 6.27: Schema di relazione 'TaskTypes'

TaskId	Description	UserId	OtherInfo
tasktype1	Task che esegue l'elaborazione X	rossi	
tasktype2	Task che esegue l'elaborazione Y	bianchi	
tasktype3	Task che esegue l'elaborazione Z	rossi	
...	...	...	...

Tabella 6.28: Esempio di relazione

Attributo	Tipo	Descrizione
IOId	String	L'identificatore dell'input/output
Name	String	Il nome dell'input/output
TaskId	String	L'identificatore del tipo di task a cui questo input/output fa riferimento
IO	Boolean	L'indicazione se si tratta di un input o di un output
ParNo	Integer	Il numero di parametro con cui si identifica l'input/output nella riga di comando che "invoca" l'esecuzione del corrispondente task

Tabella 6.29: Schema di relazione 'IOsOfTasks'

IOId	Name	TaskId	IO	ParNo
tasktype1.i1	i1	tasktype1	input	1
tasktype1.i2	i2	tasktype1	input	2
tasktype1.i3	i3	tasktype1	input	3
tasktype1.o1	o1	tasktype1	output	4
tasktype1.o2	o2	tasktype1	output	5
tasktype2.i1	i1	tasktype2	input	1
tasktype2.i2	i2	tasktype2	input	2
tasktype2.o1	o1	tasktype2	output	3
tasktype3.i1	i1	tasktype3	input	1
tasktype3.i2	i2	tasktype3	input	2
tasktype3.o1	o1	tasktype3	output	3
tasktype3.o2	o2	tasktype3	output	4
...	...	...	...	...

Tabella 6.30: Esempio di relazione

L'attributo *TaskId* permette di definire l'associazione tra la classe 'Task Types' e le classi che definiscono i corrispondenti input-output.

Con riferimento ai tipi di task rappresentati in figura 6.31, in tabella 6.30 è presentato un esempio di relazione.

Lo schema di relazione *ImplementationsOfTasks* è usato per definire la relativa classe del modello a oggetti. Questo schema è rappresentato in tabella 6.31.

La chiave primaria è rappresentata dall'attributo *TaskImplemId*.

L'attributo *TaskId* permette di definire l'associazione tra le classi 'Implementations Of Tasks' e 'Task Types', mentre l'attributo *ArchitectId* viene usato per specificare la relazione tra le classi 'Implementations Of Tasks' e 'Architectures'. Infine con l'attributo *ExecFile* si definisce l'associazione tra le classi 'Implementations Of Tasks' e 'Executable Files'.

In tabella 6.32 è mostrato un esempio di istanza di questo schema.

Lo schema di relazione *FilesForImplementations*, rappresentato in tabella 6.33, consente di definire la classe 'Files For Implementations' del modello a oggetti, e l'associazione tra questa classe e la classe 'Implementations Of Tasks'.

Un esempio di relazione è riportato in tabella 6.34.

Lo schema *ParametersForTasks*, definito in tabella 6.35, viene utilizzato per rappresentare la corrispondente classe del modello a oggetti.

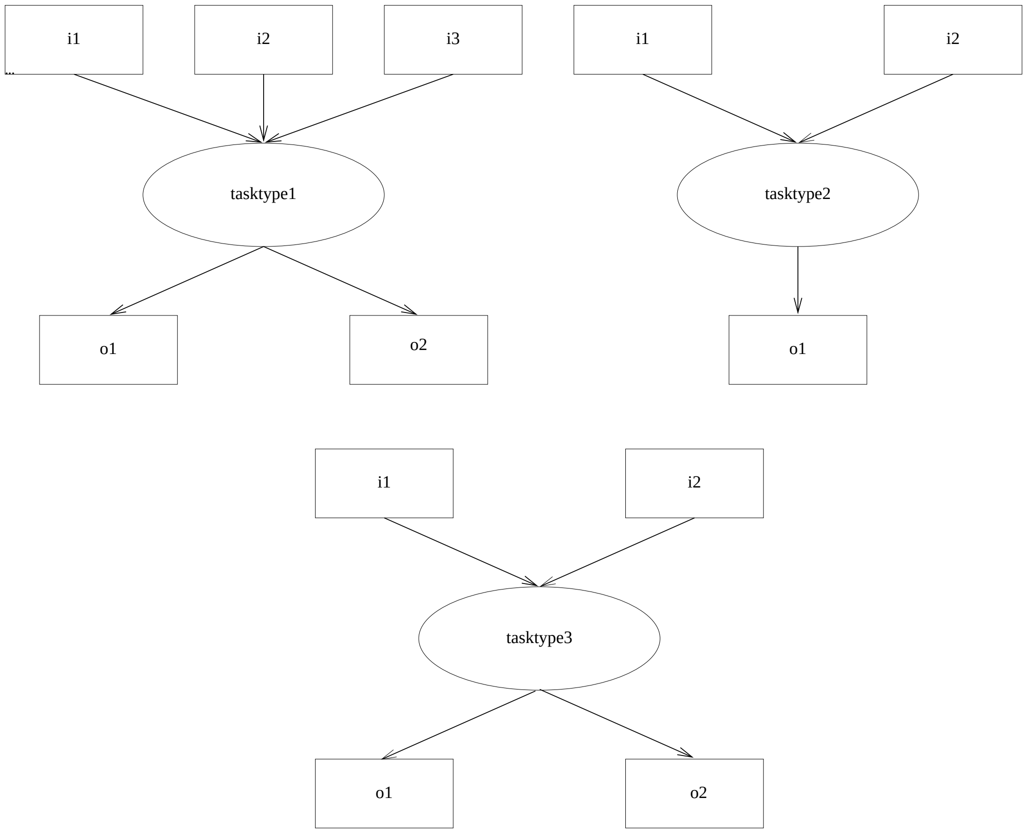

Figura 6.31: Esempi di tipi di task

Attributo	Tipo	Descrizione
TaskImplemId	String	L'identificatore dell'implementazione
TaskId	String	L'identificatore del tipo di task
ArchitectId	String	L'identificatore dell'architettura
ExecFile	String	Il nome del file che deve essere specificato nella riga di comando per l'esecuzione
Cost	Real	Il parametetro 'Cost of the task'
OtherInfo	String	Altre eventuali informazioni

Tabella 6.31: Schema di relazione 'ImplementationsOfTasks'

TaskImplemId	TaskId	ArchitectId	ExecFile	Cost
tasktype1.win32	tasktype1	win32	$DEMI/TaskType1/ win32/start.bat	5
tasktype1.axp-du	tasktype1	axp-du	$DEMI/TaskType1/ axp-du/start.sh	5
tasktype2.hp-ux	tasktype2	hp-ux	$DEMI/TaskType2/ hp-ux/start.sh	5
tasktype2.axp-win32	tasktype2	axp-win32	$DEMI/TaskType2/ axp-win32/start.bat	5
tasktype3.hp-ux	tasktype3	hp-ux	$DEMI/TaskType3/ hp-ux/start.sh	5
...	...	...	...	...

Tabella 6.32: Esempio di relazione

Attributo	Tipo	Descrizione
TaskImplemId	String	L'identificatore dell'implementazione del tipo di task
FileName	String	Il pathname del file

Tabella 6.33: Schema di relazione 'FilesForImplementations'

TaskImplemId	FileName
tasktype1.win32	$DEMI/TaskType1/win32/start.bat
tasktype1.win32	$DEMI/TaskType1/win32/execfile.exe
tasktype1.win32	$DEMI/TaskType1/win32/titlefile.tit
tasktype1.axp-du	$DEMI/TaskType1/axp-du/start.sh
tasktype1.axp-du	$DEMI/TaskType1/axp-du/execfile
tasktype1.axp-du	$DEMI/TaskType1/axp-du/titlefile.tit
tasktype2.hp-ux	$DEMI/TaskType2/hp-ux/start.sh
tasktype2.hp-ux	$DEMI/TaskType2/hp-ux/execfile
tasktype2.axp-win32	$DEMI/TaskType2/axp-win32/start.bat
tasktype2.axp-win32	$DEMI/TaskType2/axp-win32/execfile.exe
tasktype3.hp-ux	$DEMI/TaskType3/hp-ux/start.sh
tasktype3.hp-ux	$DEMI/TaskType3/hp-ux/execfile
tasktype3.hp-ux	$DEMI/TaskType3/hp-ux/titlefile.tit
...	...

Tabella 6.34: Esempio di relazione

Attributo	Tipo	Descrizione
TaskImplemId	String	L'identificatore dell'implementazione del tipo di task
ParNo	Integer	Il numero ordinale del parametro (primo/secondo/...)
Value	String	Il valore associato al parametro

Tabella 6.35: Schema di relazione 'ParametersForTasks'

TaskImplemId	ParNo	Value
tasktype1.win32	6	100
tasktype2.hp-ux	4	$DEMI/TaskType2/hp-ux/title.tit
tasktype2.hp-ux	5	200
...	...	...

Tabella 6.36: Esempio di relazione

Attributo	Tipo	Descrizione
ProdId	String	L'identificatore della produzione
Description	String	La descrizione della produzione
UserId	String	L'identifier dell'utente che ha definito la produzione
Status	Boolean	Indica se la produzione può essere usata per sottomettere richieste
OtherInfo	String	Altre eventuali informazioni

Tabella 6.37: Schema di relazione 'Productions'

Utilizzando la chiave esterna *TaskImplemId* si rappresenta l'associazione tra le classi 'Implementations Of Tasks' e 'Parameters For Tasks'.

In tabella 6.36 è mostrato l'esempio di un'istanza dello schema.

6.4.10 Production

In tabella 6.37 è mostrato lo schema di relazione *Productions*, utilizzato per la rappresentazione della relativa classe del modello a oggetti.

La chiave primaria è costituita dall'attributo *ProdId*.

L'attributo *UserId* permette di "modellare" l'associazione tra le classi 'General Users' e 'Productions'.

In tabella 6.38 è mostrato un esempio di istanza per lo schema considerato.

La classe 'Tasks Of Productions' del modello a oggetti viene rappresentata attraverso lo schema di relazione *TasksOfProductions*, definito in tabella 6.39.

La chiave primaria è rappresentata dall'attributo *TaskOfProdId*.

L'attributo *TaskId* è la chiave esterna che definisce la relazione tra le classi 'Tasks Of Productions' e 'Task Types', mentre l'associazione tra le classi 'Tasks Of Productions' e 'Productions' è rappresentata grazie alla chiave esterna *ProdId*.

ProdId	Description	UserId	Status	OtherInfo
prod1	Produzione che gestisce l'attività X	rossi	available	
prod2	Produzione che gestisce l'attività Y	bianchi	available	
prod3	Produzione che gestisce l'attività Z	rossi	not available	
prod4	Produzione che gestisce l'attività W	rossi	not available	
...	...	...	...	...

Tabella 6.38: Esempio di relazione

Attributo	Tipo	Descrizione
TaskOfProdId	String	L'identificatore del task componente di una produzione
Name	String	Nome del task
TaskId	String	L'identificatore del tipo di task
ProdId	String	L'identificatore della produzione

Tabella 6.39: Schema di relazione 'TasksOfProductions'

TaskOfProdId	Name	TaskId	ProdId
prod1.t1	t1	tasktype1	prod1
prod1.t2	t2	tasktype2	prod1
prod1.t3	t3	tasktype3	prod1
prod2.t1	t1	tasktype2	prod2
...	...	...	...

Tabella 6.40: Esempio di relazione

Attributo	Tipo	Descrizione
DataBoxId	String	L'identificatore del data box di una produzione
Name	String	Nome del data box
ProdId	String	L'identificatore della corrispondente production
Type	Boolean	L'indicazione se si tratta di un data box di tipo stream o di tipo singleton
Role	String	"Ruolo" di questo data box (input/output/temporary)

Tabella 6.41: Schema di relazione 'DataBoxesOfProductions'

Con riferimento alle produzioni rappresentate in figura 6.32, definite attraverso i tipi di task prima mostrati in figura 6.31, in tabella 6.40 è riportato un esempio di relazione.

Lo schema *DataBoxesOfProductions*, definito in tabella 6.41, rappresenta la classe 'Data Boxes of Productions' del modello a oggetti, e le relative sottoclassi.

La chiave primaria di questo schema è definito dall'attributo *DataBoxId*.

Attraverso la chiave esterna *ProdId* si definisce la relazione tra le classi 'Data Boxes Of Productions' e 'Productions'.

Considerando sempre le produzioni di figura 6.32, in tabella 6.42 è mostrato un esempio di relazione.

Lo schema di relazione *Relations*, definito in tabella 6.43, permette di rappresentare l'associazione ternaria tra le classi 'Tasks Of Productions', 'Data Boxes Of Productions' e 'IOs Of Tasks'.

L'istanza di relazione relativa alle produzioni di figura 6.32 è presentata in tabella 6.44

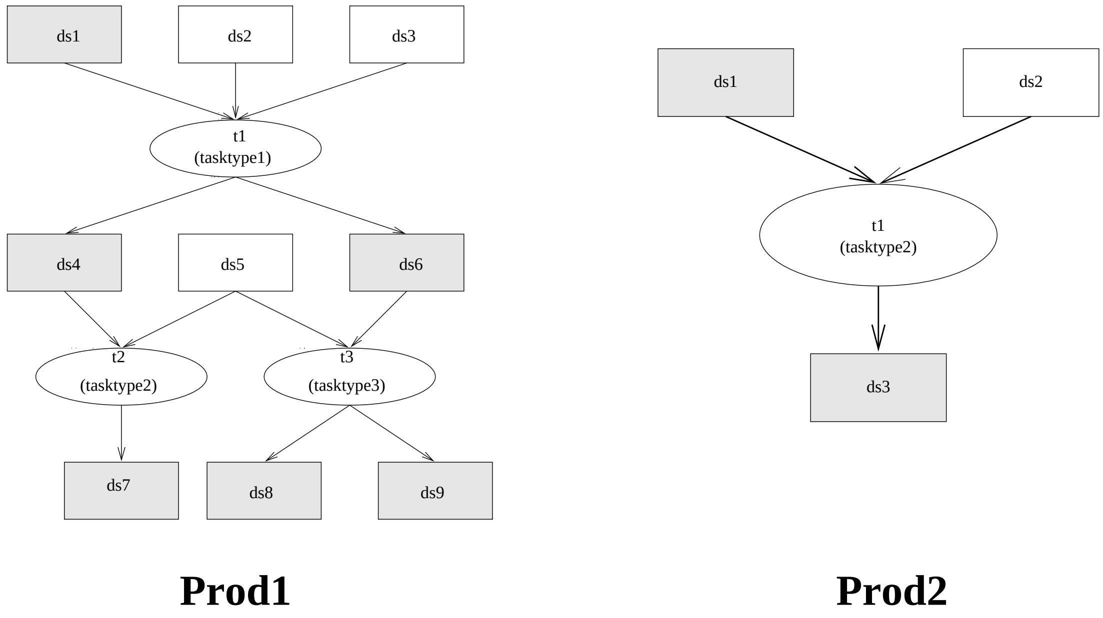

Figura 6.32: Esempi di produzioni

DataBoxId	Name	ProdId	Type	Role
prod1.ds1	ds1	prod1	stream	input
prod1.ds2	ds2	prod1	singleton	input
prod1.ds3	ds3	prod1	singleton	input
prod1.ds4	ds4	prod1	stream	output
prod1.ds5	ds5	prod1	singleton	input
prod1.ds6	ds6	prod1	stream	temp
prod1.ds7	ds7	prod1	stream	output
prod1.ds8	ds8	prod1	stream	output
prod1.ds9	ds9	prod1	stream	output
prod2.ds1	ds1	prod2	stream	input
prod2.ds2	ds2	prod2	singleton	input
prod2.ds3	ds3	prod2	stream	output
...	...	...	..	... .

Tabella 6.42: Esempio di relazione

Attributo	Tipo	Descrizione
DataBoxId	String	L'identificatore di un data box di una produzione
IOId	String	L'identificatore di un input/output di un tipo di task
TaskOfProdId	String	L'identificatore di un task componente di una produzione

Tabella 6.43: Schema di relazione 'Relations'

DataBoxId	IOId	TaskOfProdId
prod1.ds1	tasktype1.i1	prod1.t1
prod1.ds2	tasktype1.i2	prod1.t1
prod1.ds3	tasktype1.i3	prod1.t1
prod1.ds4	tasktype1.o1	prod1.t1
prod1.ds4	tasktype2.i1	prod1.t2
prod1.ds5	tasktype2.i2	prod1.t2
prod1.ds5	tasktype3.i1	prod1.t3
prod1.ds6	tasktype1.o2	prod1.t1
prod1.ds6	tasktype3.i2	prod1.t3
prod1.ds7	tasktype2.o1	prod1.t2
prod1.ds8	tasktype3.o1	prod1.t3
prod1.ds9	tasktype3.o2	prod1.t3
prod2.ds1	tasktype2.i1	prod2.t1
prod2.ds2	tasktype2.i2	prod2.t1
prod2.ds3	tasktype2.o1	prod2.t1

Tabella 6.44: Esempio di relazione

Attributo	Tipo	Descrizione
ReqId	String	L'identificatore della richiesta
UserId	String	L'identificatore dell'utente che ha sottomesso la richiesta
ProdId	String	L'identificatore della produzione
Prior	Integer	Il livello di priorità della richiesta
SubmitTime	Date-Time	La data e l'ora in cui la richiesta è stata sottomessa
NumIt	Integer	Il numero di volte per cui la relativa produzione deve essere eseguita (se non ci sono stream input)
Status	String	Lo stato dell'elaborazione
OtherInfo	String	Altre eventuali informazioni

Tabella 6.45: Schema di relazione 'Requests'

ReqId	UserId	ProdId	Prior	SubmitTime	NumIt	Status	OtherInfo
req001	rossi	prod1	5	21-03-1998 13:42	0	Satisfied	
req002	bianchi	prod2	4	21-03-1998 15:13	0	In execution	
...	...	...	...	...	...	...	...

Tabella 6.46: Esempio di relazione

6.4.11 Computational request

Lo schema di relazione *Requests*, rappresentato in tabella 6.45, viene utilizzato per definire la relativa classe del modello a oggetti.

La chiave primaria di questo schema di relazione è definita dall'attributo *ReqId*.

L'attributo *UserId* definisce l'associazione tra le classi 'General Users' e 'Requests', mentre con l'attributo *ProdId* si rappresenta l'associazione tra le classi 'Requests' e 'Productions'.

Un esempio di istanza di questo schema di relazione è rappresentato in tabella 6.46.

L'associazione con attributi tra le classi 'Requests' e 'Data Boxes Of Productions' viene rappresentata attraverso lo schema di relazione *DataBoxesOfRequests*, rappresentato in tabella 6.47.

Una possibile istanza relativa a questo schema, è riportata in tabella 6.48

Attributo	Tipo	Descrizione
ReqId	String	L'identificatore della richiesta
DataBoxId	String	L'identificatore del data box della produzione
MaxSize	Real	La dimensione massima (espressa ad esempio in MB) di un file che verrà associato al data box, per la richiesta considerata

Tabella 6.47: Schema di relazione 'DataBoxesOfRequests'

ReqId	DataBoxId	MaxSize
req001	prod1.ds1	20
req001	prod1.ds2	2
req001	prod1.ds3	2
req001	prod1.ds4	120
req001	prod1.ds5	3.5
req001	prod1.ds6	85
req001	prod1.ds7	25
req001	prod1.ds8	50
req002	prod2.ds1	100
req002	prod2.ds2	1
req002	prod2.ds3	10
...	...	...

Tabella 6.48: Esempio di relazione

Attributo	Tipo	Descrizione
ReqId	String	L'identificatore della richiesta
DataBoxId	String	L'identificatore del data box della produzione
Description	String	Il tipo dei file prodotti con questo output
Medium	String	Il "supporto" da utilizzare per la memorizzazione dei file corrispondenti a questo output (buffer, nastro, directory "privata")
Target	String	Il nome della directory privata, o il tipo di nastro (a seconda del valore dell'attributo Medium)

Tabella 6.49: Schema di relazione 'DescriptionsOfOutputsOfRequests'

ReqId	DataBoxId	Description	Medium	Target
req001	prod1.ds4	QQPS ZEB971	buffer	
req001	prod1.ds7	QQPS DST971	tape	dlt
req001	prod1.ds8	QQPS ANA971	tape	
req001	prod1.ds9	QQPS PTR971	tape	
req002	prod2.ds3	BAFO PHI971	privdir	$DISK1/Output1

Tabella 6.50: Esempio di relazione

Con lo schema di relazione *DescriptionsOfOutputsOfRequests*, mostrato in tabella 6.49, si "modella" l'associazione ternaria con attributi tra le classi 'Requests', 'Outputs Of Productions' e 'Data Types'.

In tabella 6.50 è presentato un esempio di istanza di relazione.

La relazione *SingletionIOOfRequests*, rappresentata in tabella 6.51, definisce le associazioni ternarie tra le classi 'Requests', 'Singleton Inputs Of Productions' e 'Data Files', e tra le classi 'Requests', 'Singleton Outputs Of Productions' e 'Data Files'.

Attributo	Tipo	Descrizione
ReqId	String	L'identificatore della richiesta
DataBoxId	String	L'identificatore del data box della produzione
FileId	String	L'identificatore del file

Tabella 6.51: Schema di relazione 'SingletonIOOfRequests'

ReqId	DataBoxId	FileId
req001	prod1.ds2	id00001
req001	prod1.ds3	id00005
req001	prod1.ds5	id00105
req002	prod2.ds2	id10113

Tabella 6.52: Esempio di relazione

Attributo	Tipo	Descrizione
ElemProcId	String	L'identificatore dell'elementary request
DataBoxId	String	L'identificatore del data box della produzione
FileId	String	L'identificatore del file

Tabella 6.53: Schema di relazione 'StreamIOOfElementaryRequests'

Un esempio di relazione, relativa a questo schema, è presentato in tabella 6.52.

Lo schema di relazione *StreamIOOfElementaryRequests*, rappresentato in tabella 6.53, definisce le associazioni ternarie tra le classi 'Elementary Requests', 'Stream Inputs Of Productions' e 'Data Files', e tra le classi 'Elementary Requests', 'Stream Outputs Of Productions' e 'Data Files'.

In tabella 6.54 è mostrato un esempio di relazione.

ElemProcId	DataBoxId	FileId
elproc1001	prod1.ds1	id20001
elproc1002	prod1.ds1	id20002
elproc1003	prod1.ds1	id20003
elproc2001	prod2.ds1	id31001
elproc2002	prod2.ds1	id31002
elproc2003	prod2.ds1	id31003
elproc2004	prod2.ds1	id31004
elproc2005	prod2.ds1	id31005

Tabella 6.54: Esempio di relazione

Attributo	Tipo	Descrizione
ElemReqId	String	L'identificatore della elementary request (e del relativo processing)
ReqId	String	L'identificatore della corrispondente richiesta
Status	String	Lo stato di esecuzione del processing
OtherInfo	String	Altre eventuali informazioni

Tabella 6.55: Schema di relazione 'ElementaryRequests'

ElemReqId	ReqId	Status	OtherInfo
elproc00001	req001	succesfully completed	
elproc00002	req001	in execution	
elproc00003	req002	aborted	
...	...	...	...

Tabella 6.56: Esempio di relazione

6.4.12 Processing

In tabella 6.55 è definito lo schema *ElementaryRequests*, che rappresenta la classe 'Elementary Requests' del modello a oggetti.

La chiave primaria è definita dall'attributo *ElemReqId*.

L'attributo *ReqId* è la chiave esterna che definisce l'associazione tra le classi 'Requests' e 'Elementary Requests'.

Un esempio di istanza per questo schema di relazione è rappresentato in tabella 6.56.

Per rappresentare la classe 'Local Elementary Processings' del modello a oggetti si utilizza lo schema di relazione *LocalElementaryProcessings*, definito in tabella 6.57.

La chiave primaria è costituita dall'attributo *ElemReqId*.

Attributo	Tipo	Descrizione
ElemReqId	String	L'identificatore dell'elementary request (e del relativo processing)
OtherInfo	String	Altre eventuali informazioni

Tabella 6.57: Schema di relazione 'LocalElementaryProcessings'

ElemReqId	OtherInfo
elp00001	
elp00003	
elp00013	
...	...

Tabella 6.58: Esempio di relazione

Attributo	Tipo	Descrizione
ElemReqId	String	L'identificatore dell'elementary request (e del relativo processing)
TaskOfProdId	String	L'identificatore del task componente la produzione
CompName	String	Il nome del computer a cui è stata assegnata l'esecuzione del task
Status	String	Lo stato dell'esecuzione
StartTime	Date-Time	La data e l'ora in cui è iniziata l'esecuzione
EndTime	Date-Time	La data e l'ora in cui è terminata l'esecuzione

Tabella 6.59: Schema di relazione 'ProcessingsOfSingleTasks

Un esempio di relazione è presentata in tabella 6.58.

In tabella 6.59 è definito lo schema di relazione *ProcessingsOfSingleTasks*, usato per rappresentare la relativa classe del modello a oggetti.

L'attributo *ElemReqId* permette di definire l'associazione tra le classi 'Local Elementary Processings' e 'Processings Of Single Tasks', mentre per rappresentare l'associazione tra le classi 'Processings Of Single Tasks' e 'Tasks Of Productions' si considera l'attributo *TaskOfProdId*. Infine l'attributo *CompName* definisce la relazione tra le classi 'Processings Of Single Tasks' e 'Local Computers'.

In tabella 6.60 si presenta un esempio di relazione.

ElemReqId	TaskOfProdId	CompNam	Status	StartTime	EndTime
elp00001	prod1.t1	osde00	failed	20-01-1998 23:52	
elp00001	prod1.t1	osde00	succesfully completed	21-01-1998 08:02	21-01-1998 12:53
elp00001	prod1.t2	pccms1	succesfully completed	21-01-1998 13:01	21-01-1998 13:23
elp00001	prod1.t3	hpcms01	succesfully completed	21-01-1998 12:59	21-01-1998 19:23
elp00003	prod2.t1	pccms1	in execution	22-01-1998 11:50	
...	...	...	...	...	...

Tabella 6.60: Esempio di relazione