

Numero 17
Aprile 2014

ArsT_EXnica

Rivista italiana di T_EX e L^AT_EX

GU^{IT}

<http://www.guitex.org/arstexnica>



G_UI_T – Gruppo Utilizzatori Italiani di T_EX

ArsT_EXnica è la pubblicazione ufficiale del G_UI_T

Comitato di Redazione

Claudio Beccari – *Direttore*

Roberto Giacomelli – *Comitato scientifico*

Enrico Gregorio – *Comitato scientifico*

Ivan Valbusa – *Comitato scientifico*

Lorena Rachele Badile, Renato Battistin,

Riccardo Campana, Massimo Caschili,

Gustavo Cevolani, Massimiliano Dominici,

Tommaso Gordini, Carlo Marmo,

Gianluca Pignalberi, Ottavio Rizzo,

Gianpaolo Ruocco, Enrico Spinielli,

Emiliano Vavassori

ArsT_EXnica è la prima rivista italiana dedicata a T_EX, a L^AT_EX ed alla tipografia digitale. Lo scopo che la rivista si prefigge è quello di diventare uno dei principali canali italiani di diffusione di informazioni e conoscenze sul programma ideato quasi trent'anni fa da Donald Knuth.

Le uscite avranno, almeno inizialmente, cadenza semestrale e verranno pubblicate nei mesi di Aprile e Ottobre. In particolare, la seconda uscita dell'anno conterrà gli Atti del Convegno Annuale del G_UI_T, che si tiene in quel periodo.

La rivista è aperta al contributo di tutti coloro che vogliano partecipare con un proprio articolo. Questo dovrà essere inviato alla redazione di ArsT_EXnica, per essere sottoposto alla valutazione di recensori. È necessario che gli autori utilizzino la classe di documento ufficiale della rivista; l'autore troverà raccomandazioni e istruzioni più dettagliate all'interno del file di esempio (.tex). Tutto il materiale è reperibile all'indirizzo web della rivista.

Gli articoli potranno trattare di qualsiasi argomento inerente al mondo di T_EX e L^AT_EX e non dovranno necessariamente essere indirizzati ad un pubblico esperto. In particolare tutorials, rassegne e analisi comparate di pacchetti di uso comune, studi di applicazioni reali, saranno bene accettati, così come articoli riguardanti l'interazione con altre tecnologie correlate.

Di volta in volta verrà fissato, e reso pubblico sulla pagina web, un termine di scadenza per la presentazione degli articoli da pubblicare nel numero in preparazione della rivista. Tuttavia gli articoli potranno essere inviati in qualsiasi momento e troveranno collocazione, eventualmente, nei numeri seguenti.

Chiunque, poi, volesse collaborare con la rivista a qualsiasi titolo (recensore, revisore di bozze, grafico, etc.) può contattare la redazione all'indirizzo:

arstexnica@sssup.it.

Nota sul Copyright

Il presente documento e il suo contenuto è distribuito con licenza © Creative Commons 2.0 di tipo “Non commerciale, non opere derivate”. È possibile, riprodurre, distribuire, comunicare al pubblico, esporre al pubblico, rappresentare, eseguire o recitare il presente documento alle seguenti condizioni:

① **Attribuzione:** devi riconoscere il contributo dell'autore originario.

© **Non commerciale:** non puoi usare quest'opera per scopi commerciali.

⊖ **Non opere derivate:** non puoi alterare, trasformare o sviluppare quest'opera.

In occasione di ogni atto di riutilizzazione o distribuzione, devi chiarire agli altri i termini della licenza di quest'opera; se ottieni il permesso dal titolare del diritto d'autore, è possibile rinunciare ad ognuna di queste condizioni.

Per maggiori informazioni:

<http://www.creativecommons.org>

Associarsi a G_UI_T

Fornire il tuo contributo a quest'iniziativa come membro, e non solo come semplice utente, è un presupposto fondamentale per aiutare la diffusione di T_EX e L^AT_EX anche nel nostro paese. L'adesione al Gruppo prevede una quota di iscrizione annuale diversificata: 30,00 € soci ordinari, 20,00 (12,00) € studenti (junior), 75,00 € Enti e Istituzioni.

Indirizzi

Gruppo Utilizzatori Italiani di T_EX:

c/o Ufficio Statistica

Scuola Superiore Sant'Anna

Piazza Martiri della Libertà 33

56127 Pisa, Italia.

<http://www.guitex.org>

guit@sssup.it

Redazione ArsT_EXnica:

<http://www.guitex.org/arstexnica/>

arstexnica@sssup.it

Codice ISSN 1828-2369

Stampata in Italia

Pisa: 15 Aprile 2014

ArsT_EXnica

Rivista italiana di T_EX e L^AT_EX

Numero 17, Aprile 2014

Editoriale	
<i>Claudio Beccari</i>	3
I comandi condizionali	
<i>Claudio Beccari</i>	4
ArsT _E Xnica si rinnova	
<i>Claudio Beccari, Massimiliano Dominici, Gianluca Pignalberi</i>	12
Realizzare semplici animazioni in figure: come usare TikZ in Beamer	
<i>Claudio Fiandrino</i>	18
Testatine e piè di pagina laterali	
<i>Gianluca Pignalberi</i>	24
Un primo sguardo al modulo publications	
<i>Luigi Scarso</i>	29

Gruppo Utilizzatori Italiani di T_EX

Editoriale

Claudio Beccari

Questo è il mio primo editoriale.

Quando nell'editoriale del numero 16 della rivista il precedente direttore Gianluca Pignalberi dave se stesso per defunto (come direttore, s'intende) durante la seduta del Meeting 2013 fu fatto il mio nome come suo successore.

Ero molto perplesso, perché passo molto tempo in viaggio e all'estero; non sono un turista "coatto", ma avendo due su tre figli residenti all'estero e quattro su sei nipotini residenti all'estero, mi sembra doveroso muovermi fra i loro luoghi di residenza, insieme a mia moglie, ovviamente, per mantenere contatti stabili con figli, generi, nuore, e nipotini.

Ma ora eccomi qui; nominato direttore di *Ar_sTeXnica* dal Consiglio /Direttivo del *G_UIT*, sono qui a presentare il primo numero che porta il mio nome come direttore.

Questo numero non è particolarmente ricco di articoli, ma sono tutti eccellenti, escluso quello che ho scritto io, che è un banale tutorial sull'uso dei comandi condizionali; meno male, dirà qualche lettore assiduo, così almeno non ci tormenta più come l'adattamento di *L^ATeX* alle lingue romanze alpine. No, come direttore in futuro dovrò trattenermi perché chiaramente l'accettazione dei miei articoli potrebbe essere sospetta. Per fortuna ci sono i *referee* che sanno essere assolutamente *super partes* e già per questi articoli hanno fatto un lavoro stupendo; li ringrazio moltissimo e spero che continueranno a svolgere questa attività importantissima per molti anni.

Ma la mia firma compare in un altro articolo insieme a quella dei due precedenti direttori di *Ar_sTeXnica*, Massimiliano Dominici e Gianluca Pignalberi. Abbiamo voluto non solo descrivere il processo piuttosto elaborato che porta al risultato finale che state leggendo, ma anche ai piani per il futuro. A parte gli aspetti tecnici, che ovviamente sono essenziali, resta aperta la questione di comporre la rivista in formato B5, magari rifilato alle dimensioni di 170 mm per 240 mm, il formato prediletto da Ivan Valbusa che l'ha proposto diverso tempo fa. Abbiamo fatto diversi test e ora abbiamo un'idea abbastanza precisa sull'aumento delle pagine che la riduzione del formato comporterebbe; non abbiamo ancora un'idea precisa dell'aumento dei costi che questa riduzione di formato implica. Non abbiamo nemmeno idea circa il gradimento di questa riduzione di formato da parte dei lettori, innanzi tutto quelli che ricevono la rivista in quanto soci. L'articolo pone domande in questo senso ai lettori.

L'articolo di Claudio Fiandrino descrive diversi metodi per integrare le funzionalità grafiche di TikZ con quelle della classe Beamer al fine di creare delle presentazioni "dinamiche"; presentazioni, cioè in cui il contenuto di un determinato quadro cambia in proiezioni successive; l'autore chiama "animazioni" questi effetti, ma si riferisce a cose ben diverse da soluzioni che potrebbero ricordare i cartoni animati; l'articolo non è un tutorial e descrive funzionalità che tutti vorremmo per maneggiare con la sua stessa maestria per rendere le nostre presentazioni più efficaci nel tramettere il messaggio desiderato.

L'articolo di Gianluca Pignalberi riguarda un certo modo di comporre le pagine spostando quanto normalmente si trova nelle testatine e nei piè di pagina nei margini e fuori dello "specchio di stampa"; si tratta di un modo diverso di decorare la pagine, mantenendone la stessa funzionalità, ma con un approccio più moderno; egli mostra come usare il pacchetto di personalizzazione *fancyhdr* per ottenere questo tipo di "decorazioni" della pagina.

L'ultimo articolo è scritto da Luigi Scarso; riguarda il modo di usare il programma avanzato Con_TEXt, con il sottostante Lua_{TeX}, per gestire la bibliografia; l'articolo è decisamente avanzato, come lo sono tutti gli articoli che l'autore pubblica su *Ar_sTeXnica*; ma vedere che cosa riesce a fare lui con la sua profonda conoscenza non solo dei programmi delle distribuzioni del sistema *TeX*, ma anche con le altre raffinatezze di programmazione di alto livello, è una cosa che affascina; ci fa anche toccare con mano che il futuro del sistema *TeX* è ricco di sorprese. Avremo il privilegio di essere testimoni di questa sorprendente evoluzione.

Chiudo quindi questo editoriale chiedendo venia degli errori che purtroppo commetterò, anche se cercherò di evitarli a tutti i costi; ho svolto la funzione di Scientific Committee Chairman per alcuni congressi, ma non ho mai svolto la funzione di direttore di una rivista, che io apprezzo fin dal primo suo numero; non vorrei proprio che sotto la mia direzione la rivista si impoverisca di contenuti e perda l'interesse che finora ha suscitato nei lettori.

▷ Claudio Beccari
claudio dot beccari
at gmail dot com

I comandi condizionali

Claudio Beccari

Sommario

I comandi condizionali sono raramente usati dagli utenti $\text{\LaTeX}$ e quelle poche volte che lo fanno ricorrono, per evitare alcune loro trappole, a pacchetti esterni. Tali pacchetti però non sempre sono utili allo scopo. Fronteggiare gli errori che si presentano richiede una conoscenza approfondita di ciò che sta sotto ai comandi condizionali, al fine di servirsi di quelli più adatti e più in generale dei pacchetti che li definiscono.

Abstract

Conditional commands are seldom used by $\text{\LaTeX}$ users and in those rare instances they resort to external packages that are supposed to skip the insidious traps set up by these commands. Some such packages sometimes don't succeed in this task. In order to master the unusual error situations that may arise, it is better to know what there is under the hood of the typesetting engine conditional command interpretation so as to use them or the right packages in the proper way.

1 Introduzione

I comandi condizionali sono quelli che cominciano con `\if`; ce ne sono molti, predefiniti già dagli albori di $\text{\TeX}$ e accessibili direttamente dagli utenti di $\text{\LaTeX}$; questo a sua volta ne definisce altri mediante una sintassi più robusta. Inoltre le estensioni di $\text{\TeX}$, grazie a $\varepsilon\text{-TeX}$, sono già incorporate nei motori di composizione *pdf_{te}x* dal lontano 2005 ed analogamente nei motori di composizione *xet_{ex}* e *luat_{ex}* fin dalla loro nascita. Questi nuovi comandi estendono la possibilità di verificare diversi altri eventi, cosa difficilmente controllare con i comandi originali.

I comandi condizionali normalmente richiedono la sintassi seguente, nonostante altri comandi possano venire definiti in modo diverso:

```
\if<condizionale>
  <azione se vero>
\else
  <azione se falso>
\fi
```

A loro volta le azioni *<azione se vero>* e *<azione se falso>* possono contenere altre espressioni condizionali complete. Questo fatto però, come vedremo più avanti, rende questi costrutti relativamente

fragili, in particolare quando essi sono complicati oppure quando particolari azioni sono da compiere.

1.1 I comandi di $\text{\TeX}$

I comandi oggi disponibili a livello di interprete (cioè definiti nel motore di composizione, non nel nucleo di $\text{\LaTeX}$) sono i seguenti (BREITENLOHNER, 1998; KNUTH, 1996):

`\if` confronta due token; questi possono essere dei singoli caratteri espliciti, delle macro, argomenti di macro; prima di eseguire i confronti questi token vengono eventualmente sviluppati; se lo sviluppo di una macro contiene a sua volta più token, il confronto viene fatto fra i primi due.

`\ifcat` confronta i codici di categoria di due token che rappresentano caratteri espliciti, argomenti di macro, macro, e simili; le macro vengono sviluppate prima che venga eseguito il confronto. Si noti la differenza fra `\if` e `\ifcat`; il secondo confronta i codici di categoria di due caratteri, il primo confronta i codici interni di due caratteri, quasi sempre coincidenti con i codici ASCII.

`\ifnum` esegue un confronto numerico fra due entità rappresentate da contatori, numeri espliciti, macro, caratteri impliciti, argomenti di macro, eccetera; le due quantità da confrontare sono separate da un operatore di relazione fra i seguenti: `=`, `>`, `<`. Prima di eseguire il confronto le macro vengono sviluppate e i contenuti dei contatori vengono recuperati, in modo che i confronti avvengano realmente fra numeri.

`\ifdim` simile a `\ifnum`, ma confronta due lunghezze; di nuovo gli elementi da confrontare possono venire indicati con macro, registri lunghezza, lunghezze esplicite, eccetera. I due token sono separati da due operatori di relazione come per i controlli numerici.

`\ifodd` controlla se lo sviluppo di quello che segue rappresenta un valore numerico intero dispari.

`\ifvmode` controlla se il modo di comporre corrente sia quello verticale.

`\ifhmode` controlla se il modo di comporre corrente sia quello orizzontale; in $\text{\LaTeX}$ questo modo è anche chiamato *LR mode*.

`\ifmmode` controlla se si sta componendo in modo matematico.

`\ifinner` controlla se si sta componendo in modo 'interno'; questo capita per esempio quando si sta componendo testo all'interno di una scatola; per

gli utenti di L $\text{\TeX}$ vale la pena di ricordare che quando si compone all'interno degli ambienti `figure` e `table` si è in modo interno.

`\ifvoid` controlla se un token che rappresenta il nome di una scatola o il suo numero identificativo esplicito indica una scatola completamente vuota.

`\ifhbox` controlla se una scatola contenga materiale orizzontale

`\ifvbox` controlla se una scatola contenga materiale verticale.

`\ifx` controlla se due macro hanno le stesse caratteristiche per quanto riguarda la loro natura globale e se siano entrambe 'long'¹ e se la loro definizione è coincidente; le due macro *non* vengono sviluppate.

`\ifeof` controlla se nel leggere un file si è raggiunta la sua fine (*end of file*); quando si cerca di leggere un file inesistente, la sua inesistenza equivale ad un *end of file*; viene usato per esempio in comandi L $\text{\TeX}$ del tipo `\IfFileExists` o `\InputIfFileExists` ma può venire usato da chiunque.

`\iftrue` è un test particolare perché restituisce sempre il valore 'vero'.

`\iffalse` analogamente questo è un test particolare che restituisce sempre il valore 'falso'.

`\ifcase` è un test soggetto a un contatore; verifica il contenuto del contatore, che supponiamo contenga il valore n , poi esegue fra un elenco di azioni possibili, l' n -esima azione; le azioni della lista sono separate mediante il comando `\or`; se non si vuole esaminare la lista oltre un certo punto si può usare il comando `\else` e definire un'azione di default; in ogni caso la lista viene terminata mediante il comando `\fi`.

`\ifdefined` controlla se un token è definito; tipicamente il token è una macro o una 'control sequence' a cui può essere stato assegnato un significato con `\let`.

`\ifcsname` controlla se il nome di una 'control sequence' corrisponde ad una macro definita; il nome della macro viene esplicitato senza il backslash davanti e viene terminato mediante `\endcsname`. Questo controllo è più efficace di quello con la vecchia sintassi; si noti infatti che i seguenti due costrutti (vecchio e nuovo) sono equivalenti:

```
\expandafter
  \csname_{nome}\endcsname\relax
  <azione se falso>
\else
  <azione se vero>
\fi
```

1. Una macro 'long' accetta come argomenti anche più capoversi; una macro che non abbia questa caratteristica non accetta che una frazione di capoverso.

```
\ifcsname_{nome}\endcsname
  <azione se vero>
\else
  <azione se falso>
\fi
```

`\iffontchar` viene seguito da un nome di font, oppure dal comando `\font` che rappresenta il nome del font corrente, e poi da un numero non superiore a 255, per verificare se il glifo con quell'indirizzo dentro quel font esiste davvero. L'indirizzo 255 è il massimo che utilizzabile con *pdf $\text{\TeX}$* il quale accetta solo font di tipo Type 1 o comunque senza caratteri oltre quell'indirizzo. Con i motori *xetex* e *luatex*, che gestiscono font UNICODE contenuti in polizze ricchissime di caratteri di ogni genere, il numero corrispondente all'indirizzo può essere virtualmente grande 'a piacere', senza superare la capacità di gestione dei numeri interi da parte del sistema T $\text{\TeX}$.

`\unless` questo non è un vero comando condizionale, ma quando precede un comando condizionale ne nega l'esito; quindi, se senza `\unless` un comando condizionale fornisce il risultato 'vero', allora preceduto da `\unless` fornisce il risultato 'falso'.

Altri comandi condizionali possono venire definiti mediante il comando `\newif` e si comportano praticamente come i comandi di T $\text{\TeX}$. Si tenga presente che il comando `\newif` definisce il test logico il cui nome comincia con `\if`, ma definisce anche due altri comandi per assegnare il valore vero o falso a questo comando; per esempio, se volessimo definire la variabile booleana `blu` per controllarne il valore mediante il test logico `\ifblu`, allora si scriverebbe:

```
\newif\ifblu
```

Appena viene definito, il test è 'falso' ma `\newif` ha definito anche i comandi `\blutruer` e `\blufalse` con cui assegnare il valore 'vero' o 'falso' al test logico `\ifblu`; di fatto i comandi in questione sono:

```
\def\blutruer{\let\ifblu\iftrue}
\def\blufalse{\let\ifblu\iffalse}
```

così che di volta in volta essi rendono il comando `\ifblu` equivalente ai test nativi `\iftrue` o `\iffalse` visti sopra.

1.2 I comandi di L $\text{\TeX}$

Altri comandi condizionali definiti mediante il comando `\newif` vengono di solito definiti nei file di classe; per esempio la classe `book` definisce anche i comandi condizionali relativi alle opzioni che può gestire; i significati e gli scopi traspaiono abbastanza dai loro nomi;

```
\if@restonecol      \if@titlepage
\if@openright        \if@mainmatter
\if@compatibility    \if@twoside
\if@twocolumn
```

I comandi condizionali di $\text{\LaTeX}$ sono definiti mediante macro che a loro volta si appoggiano ai condizionali nativi del linguaggio $\text{\TeX}$, ma sono costruiti in modo da essere più robusti. Tali comandi condizionali sono definiti all'interno del nucleo di $\text{\LaTeX}$ e sono ‘protetti’, nel senso che, contenendo il carattere ‘@’ che normalmente non è una lettera, evitano che l'utente possa inavvertitamente ridefinire un comando interno. Essi sono indicati nell'elenco che segue senza descriverne la sintassi, sebbene essa traspaia abbastanza dal nome.

<code>\@ifnextchar</code>	<code>\@ifstar</code>
<code>\@ifdefinable</code>	<code>\@ifundefined</code>
<code>\@ifforloop</code>	<code>\@iffileonpath</code>
<code>\@ifframebox</code>	<code>\@ifframepicbox</code>
<code>\@ifatmargin</code>	<code>\@ifpackageloaded</code>
<code>\@ifclassloaded</code>	<code>\@ifpackagelater</code>
<code>\@ifclasslater</code>	<code>\@ifl@aded</code>
<code>\@ifl@ter</code>	<code>\@ifl@t@r</code>
<code>\@ifpackagewith</code>	<code>\@ifclasswith</code>
<code>\@if@ptions</code>	<code>\@if@pti@ns</code>

In generale questi comandi interni di $\text{\LaTeX}$ richiedono due argomenti (fra graffe, se non sono costituiti da un solo token) che contengono rispettivamente l'azione se vero e l'azione se falso. Talvolta questi due argomenti possono essere preceduti da un ulteriore primo argomento; infatti all'interno del nucleo di $\text{\LaTeX}$ i comandi sono definiti in modo da accorgersi se il primo argomento è facoltativo, ovvero racchiuso fra parentesi quadre, o se gli argomenti che seguono il comando sono solo quelli obbligatori; per esempio, è nota la sintassi del comando matematico per indicare la radice:

$$\sqrt[\langle indice \rangle]{\langle radicando \rangle}$$

L'azione se vero della radice è chiaramente facoltativo, perciò la definizione del comando `\sqrt` deve verificare se il primo token che lo segue (*next character*, un singolo token) sia o meno una parentesi quadra; infatti la definizione dentro il nucleo di $\text{\LaTeX}$ (contenuto nel file `latex.ltx`) è la seguente:

```
\DeclareRobustCommand\sqrt{%
  \@ifnextchar[\@sqrt\sqrtsign
}
```

Si vede dunque che il condizionale `\@ifnextchar` è seguito da tre token; il primo è il carattere da verificare, la parentesi quadra aperta, il secondo è il comando da eseguire se il risultato del confronto è vero, mentre il terzo indica l'azione da eseguire se il confronto è falso.

2 La fragilità dei comandi

Fin dai suoi albori i comandi dell'interprete $\text{\TeX}$ e di quelli definiti nel primo insieme di macro `plain.tex` sono risultati affetti dalla fragilità; oggi il $\text{\LaTeX}$ 3 Team ha apportato moltissime correzioni

al nucleo di $\text{\LaTeX}$ in modo da eliminare quasi completamente la presenza di comandi fragili, tanto che oggi è difficile incontrarne; ma se l'utente scrive le sue macro ricorrendo ai comandi dell'interprete, allora è possibile che egli stesso crei macro fragili.

Una macro è robusta quando il suo testo sostitutivo, cioè la sua definizione, viene sempre sviluppato ed eseguito correttamente e completamente in qualsiasi circostanza; altrimenti il comando è fragile.

La guida di Leslie Lamport (LAMPORT, 1994) (come la sua traduzione commentata (BECCARI, 2014a)) riportano un elenco di comandi $\text{\LaTeX}$ fragili; non sono moltissimi, ma sono di uso frequente; tra loro il comando `\` che accetta ben due argomenti facoltativi, l'asterisco e l'ammontare della spaziatura verticale indicata fra parentesi quadre.

La fragilità è dovuta a diverse cause, la più importante delle quali è costituita dallo sviluppo incompleto delle varie macro invocate dal comando fragile; ciò può dipendere dal fatto che una parte dello sviluppo venga scritta nei file ausiliari, oppure che i vari `\if`, `\else` e `\fi` che costituiscono il costrutto “primitivo” di comandi condizionali, non vengono eseguiti al momento giusto. Alcune delle soluzioni che spesso risolvono i problemi di fragilità sono indicate nei paragrafi seguenti.

3 Cenni al linguaggio L3

Il linguaggio L3 è in fase di sviluppo da parte del $\text{\LaTeX}$ 3 Team per costituire la base per la definizione del nuovo $\text{\LaTeX}$ 3. Questo, dopo una ventina d'anni dalla decisione di realizzare la versione 3 del nuovo nucleo di $\text{\LaTeX}$, comincia ad essere disponibile in via sperimentale.

L'idea fondante del linguaggio L3 è quella di costituire un'interfaccia fra i comandi definiti dall'utente e l'interprete sottostante in modo che si possano gestire correttamente e in modo robusto tutte le possibili strutture informatiche che un programma per la composizione tipografica può richiedere, con particolare attenzione alla gestione degli argomenti facoltativi e dei comandi condizionali.

Alcune parti di questo linguaggio sono già disponibili e vengono normalmente distribuite con le installazioni moderne. Diversi pacchetti di estensione del linguaggio $\text{\LaTeX}$ sono già scritti e molti tra loro, specifici per $\text{\XeLaTeX}$ e $\text{\LuaLaTeX}$, sono già stati realizzati; in questo articolo, però, mi limiterò a descrivere i comandi ed i pacchetti che verosimilmente l'utente userà più frequentemente con $\text{\PDFLaTeX}$.

4 I pacchetti utili

Di per sé i comandi del linguaggio originale $\text{\TeX}$ non sono difficili da usare, ma hanno indubbi inconvenienti: (a) la fragilità; (b) l'impossibilità di

usare “frasi logiche”; (c) l’evidente complessità nel caso si debbano fare controlli molto complicati.

5 Le estensioni del pacchetto babel

Il pacchetto `babel` (BEZOS, 2013) mette a disposizione due comandi `\bbl@afterelse` e `\bbl@afterfi` che permettono di eseguire l’*azione se vero* o l’*azione se falso* scavalcando i comandi `\else` o `\fi`. Questi comandi usano argomenti delimitati e in un certo senso sono più efficaci ed efficienti dei comandi nativi del linguaggio `TeX`, ma sono altrettanto e forse anche più fragili di questi ultimi; in particolare richiedono la creazione di gruppi per poter costruire dei test annidati.

Dentro `babel` funzionano bene, ma ne sconsiglierei fortemente l’uso per definire macro personali.

6 Il pacchetto ifthen

Il pacchetto `ifthen` (CARLISLE, 2001) affronta questi problemi in modo più efficace; la sintassi è la seguente:

```
\ifthenelse{<test>}%
{\<azione se vero>} {\<azione se falso>}
```

dove *<test>* è un test logico che produce come risultato uno stato di ‘vero’ o ‘falso’. Il pacchetto prevede *<test>* per eseguire quasi tutti i controlli eseguibili con i comandi condizionali nativi di `TeX`; ma *<test>* può essere anche una frase logica, racchiusa fra i delimitatori `\(` e `\)`, che collega fra loro diversi test mediante i connettori logici `\NOT`, `\OR` e `\AND`. Chiunque conosca i rudimenti di logica booleana delle scuole secondarie può ragionevolmente scrivere una frase logica corretta senza avere una particolare esperienza di programmazione. Questo pacchetto è ulteriormente esteso da `xifthen` (NOIREL, 2009) che accetta una sintassi più efficace e robusta per la costruzione di frasi logiche.

Prescindiamo per ora dai *<test>* specifici che si possono eseguire e cerchiamo di capire perché la sintassi del pacchetto `ifthen` è quasi sempre robusta. Sostanzialmente la lunga e complessa definizione del comando principale `\ifthenelse` assegna il valore ‘vero’ o ‘falso’ in base al risultato di *<test>* al comando condizionale `\ifTE@val`, che è reso equivalente a `\iftrue` oppure a `\iffalse`.

Alla fine della definizione di `\ifthenelse` compare il test finale:

```
\ifTE@val
\expandafter\@firstoftwo
\else
\expandafter\@secondoftwo
\fi
```

È proprio qui che si trova il congegno che rende robusto il comando `\ifthenelse`; esso non legge

gli ultimi due argomenti della sua sintassi, si limita ad eseguire il *<test>*. La scelta se eseguire la prima o la seconda azione viene affidata a quei due comandi ‘misteriosi’ che compaiono dopo `\expandafter`: `\@firstoftwo` e `\@secondoftwo`. `LaTeX` nel suo nucleo ricorre spesso a questo genere di comandi.

Le definizioni di questi due comandi corrispondono al loro nome, accettando due argomenti ma selezionando solo il primo oppure il secondo argomento:

```
\def\@firstoftwo#1#2{#1}
\def\@secondoftwo#1#2{#2}
```

È proprio questo meccanismo che permette a `\ifthenelse` di essere praticamente robusto. Ma la sintassi di `\ifthenelse` si è mostrata fragile almeno in un caso che mostrerò più avanti nonostante ciò nulla tolga all’affidabilità del pacchetto `\ifthen`.

Vediamo un caso che riprendo dalla cosiddetta Guida `gT` (BECCARI, 2014b).

Il comando `\cleardoublepage` è definito nel nucleo di `LaTeX`² mediante l’uso dei comandi nativi di `TeX`:

```
\def\cleardoublepage{\clearpage
\if@twoside\ifodd\c@page\else
\hbox{}\newpage\if@twocolumn
\hbox{}\newpage\fi\fi\fi}
```

Sono riconoscibili i comandi condizionali `\if@twocolumn` e `\if@twoside` usati anche nella classe `book` ma definiti nel nucleo di `LaTeX`; si vede all’opera anche il comando primitivo `\ifodd`.

Usando il pacchetto `ifthen` questa definizione potrebbe essere tradotta in:

```
\newcommand*\cleardoublepage%
{\clearpage
\ifthenelse{%
\(% inizio frase logica
\boolean{@twoside}%
\AND\NOT\isodd{\value{page}}}%
\)% fine frase logica
}{%
\mbox{}\newpage
\ifthenelse{\boolean{@twocolumn}}%
{\mbox{}\newpage}{}%
}{%
}
```

Come si vede i comandi condizionali definiti con `\newif`, come `\if@twoside`, nel nucleo di `LaTeX`, vengono controllati per conoscere il loro stato di ‘vero’ o ‘falso’ mediante il comando `\boolean` il cui argomento è la variabile booleana stessa, cioè il nome del comando condizionale privato del prefisso `\if`. I connettori logici `\AND` e `\NOT` servono per definire una frase logica che connette due diversi

2. Qui se ne riporta una versione semplificata.

stati di ‘vero’ o ‘falso’; il test `\isodd` usa una sintassi leggermente diversa, ma corrisponde al comando condizionale `\ifodd`.

Il pacchetto `ifthen`, nell’esempio mostrato, con un solo comando `\ifthenelse` permette l’esecuzione della medesima operazione che richiede con i comandi nativi di T_EX tre costrutti condizionali annidati.

7 Il pacchetto `etoolbox`

Il pacchetto `etoolbox` (LEHMAN, 2011) è una preziosa “cassetta degli attrezzi” per `pdflatex`, `xelatex` e `lualatex`, tanto utile che spesso è caricato di default. Basta esaminare il file `.log` di una qualunque composizione recente, cercare la stringa `etoolbox` e constatare se il pacchetto in questione è stato caricato anche senza l’invocazione esplicita da parte dell’utente.

Questo pacchetto mette a disposizione dell’utente molte decine di nuovi comandi di cui almeno una cinquantina per gestire l’esecuzione condizionale di codice. Sarebbe troppo lungo elencarli e commentarli tutti in questo articolo, quindi rinvio il lettore alla documentazione del pacchetto. Ricordo che aprendo una finestra comandi (che prende nomi diversi con sistemi operativi diversi) e digitandovi dentro il comando

```
texdoc <pacchetto>
```

si apre il visualizzatore di file PDF o il browser di rete, che permettono di leggere direttamente la documentazione di un qualsiasi `<pacchetto>`. Talvolta il `<pacchetto>` dispone di diversi file di documentazione, per cui può essere utile aggiungere una opzione

```
texdoc -showall <pacchetto>
```

In questo modo il programma di lettura della documentazione mostra l’elenco numerato di tutti i file di documentazione relativi al `<pacchetto>` specificato; rispondendo al prompt di sistema con il numero del file che interessa, quel file viene aperto e se ne può leggere il contenuto.

La documentazione di `etoolbox` è abbastanza vasta, ma è anche molto vasto il numero di comandi messi a disposizione; il paragrafo §3.5 nella pagina 12 comincia con i comandi di definizione e di gestione delle variabili booleane; il paragrafo §3.6 che comincia nella pagina 14, descrive una moltitudine di comandi condizionali, tanto che il paragrafo finisce nella pagina 23.

Tutti i comandi condizionali accettano diversi argomenti gli ultimi due dei quali specificano l’`<azione se vero>` e l’`<azione se falso>`; in questo modo tutti i comandi sono piuttosto robusti e non ho mai riscontrato un caso in cui si “rompessero”. Più avanti mostrerò un esempio in cui gli altri

comandi condizionali o nativi di T_EX o definiti dagli altri pacchetti si mostrano fragili, ma quelli di `etoolbox` sono robusti.

In un comando condizionale gli argomenti precedenti gli ultimi due hanno funzioni speciali che differiscono talmente da comando a comando che è necessario esaminare la documentazione per comprenderne il significato e la sintassi.

Quello che balza agli occhi è la varietà di test che si possono fare per controllare lo stato dei *comandi* o dei *nomi dei comandi* (i comandi a cui è stato tolto il backslash che li introduce); per esempio:

```
\ifdef{<comando>}{<azione se vero>}%
  {<azione se falso>}
\ifcsdef{<nome del comando>}%
  {<azione se vero>}{<azione se falso>}
```

controllano entrambi se `<comando>` è definito, ma il primo fa riferimento al comando vero e proprio mentre il secondo controlla solo se esiste un comando con quel nome.

Le espressioni logiche con i vari connettori logici sono certamente eseguibili, ma forse la sintassi è più complessa di quella del pacchetto `ifthen`; secondo me è più chiara sebbene sia un poco più prolissa.

L’esempio di `\cleardoublepage` visto per il pacchetto `ifthen` diventa:

```
\newcommand*\cleardoublepage{\clearpage
  \ifboolexpr{bool{&twoside}
    and not
    test{\ifnumodd{\value{page}}}}}%
  {\mbox{} \newpage
  \ifboolexpr{bool{&twocolumn}}
    {\mbox{} \newpage}{}}
}
```

Nonostante il ripiegamento delle righe di codice si vede chiaramente che la sintassi del comando principale, che può tranquillamente venire annidato, è quella seguente:

```
\ifboolexpr{<espressione booleana>}%
  {<azione se vera>}{<azione se falsa>}
```

Si riconosce che i test definiti con `\newif` vengono usati come variabili booleane togliendo loro il prefisso `\if` ma vengono preceduti dal descrittore `bool`. Invece i test che usano argomenti vengono eseguiti con i comandi di `etoolbox` che prevedono l’inserimento dei due esiti negli ultimi due argomenti, saltando la specificazione di tali argomenti, ma premettendo il qualificatore `test` al comando condizionale con i suoi primi argomenti. I connettori logici `or`, `and` e `not` sono semplici ‘parole’, non comandi, e sono separate l’una dall’altra (e da `bool` e `test`) da semplici spazi; il comando complessivo non accetta righe vuote, ma permette di andare a capo rendendo più leggibile il codice.

Come si vede la sintassi è più articolata rispetto a quella del pacchetto `ifthen` ma è il prezzo pagato per ottenere una maggiore robustezza.

8 Confronto fra i vari comandi condizionali

Supponiamo di voler creare un pacchetto che fra le altre cose usi un test che per semplicità pensiamo creato con il comando `\newif`:

```
\newif\iftest
```

Nel definire un nuovo comando senza sovrapporlo a definizioni già esistenti, non possiamo ricorrere a un comando `\csprovideif` analogo al comando `\providecommand`. Dobbiamo quindi controllare se è già definito il test `\iftest`.

8.1 Comandi nativi di TEX

Creiamo il semplice file di prova seguente:

```
\documentclass{article}
\newif\iftest
\testfalse
% \testtrue
\nofiles
\begin{document}
\ifdefined\iftest
\typeout{definito}
\else
\typeout{non definito}
\fi\fi
\end{document}
```

La documentazione di eTEX dice che `\ifdefined` non sviluppa il comando su cui esegue il test ma questo è solo parzialmente vero. Compilando il piccolo file di prova diverse volte, commentando via via sia la definizione di `\iftest` e i suoi comandi d'impostazione, sia solo l'uno o l'altro comando d'impostazione, se ne possono osservare gli effetti.

Nel caso in cui `\iftest` è definito e a seconda del suo stato, si scopre che ci vogliono uno o due comandi di fine del costrutto condizionale; quando ce n'è uno di troppo il programma di compilazione si arresta con l'errore: `extra \fi...`

Il comando condizionale quindi è fragile e si è "rotto".

8.2 Comandi di ifthen

Si crei ora un altro file di prova come il seguente:

```
\documentclass{article}
\newif\iftest
\testfalse
% \testtrue
\usepackage{ifthen}
\nofiles
\begin{document}
\ifthenelse{\isundefined{\iftest}}
{\typeout{non definito}}
```

```
{\typeout{definito}}
\end{document}
```

Si ripetano gli esperimenti svolti nel paragrafo precedente, commentando via via i comandi che si riferiscono al test `\iftest` nelle varie combinazioni; nuovamente si scoprirà che esiste almeno una combinazione in cui lo sviluppo del comando `\ifthenelse` si "rompe" e dà luogo ad un messaggio d'errore. Di nuovo questo comando, nonostante sia più robusto dei comandi nativi di TEX, non è robusto abbastanza.

8.3 Comandi di etoolbox

Si predisponga nuovamente di un piccolo file di prova come il seguente:

```
\documentclass{article}
\newif\iftest
\testfalse
% \testtrue
\usepackage{etoolbox}
\nofiles
\begin{document}
\ifdef{\iftest}% oppure \ifcsdef{\iftest}
{\typeout{definito}}
{\typeout{non definito}}
\end{document}
```

Per vederne gli effetti, si ripetano ora i soliti test con le varie combinazioni di righe commentate in merito al test `\iftest`; si cambi anche l'enunciato che contiene `\ifdef` con l'enunciato che contiene `\ifcsdef`; si scoprirà che in nessun caso il costrutto logico si "rompe"; non sono riuscito a creare altri esempi nei quali i comandi di `etoolbox` si dimostrino fragili.

8.4 Ancora sulla fragilità dei comandi condizionali

I comandi condizionali di `etoolbox` sono molto robusti perché chi ha scritto le macro ha sfruttato a fondo le estensioni di eTEX. In tal modo è riuscito a "detokenizzare" i comandi che a loro volta fossero argomenti di altre macro, facendoli così diventare delle stringhe di caratteri ed evitando che venissero eseguiti al momento sbagliato.

Evidentemente i comandi di `ifthen`, scritti nel 2001, assai prima di quando non siano state aggiunte le modifiche di ϵ -TEX nei vari motori di composizione, funzionano molto bene, ma non sono così robusti.

I comandi condizionali nativi di TEX si rivelano fragili perché vengono eseguiti troppo presto; secondo la metafora del tubo digerente di Knuth, essi vengono elaborati nella bocca, non nello stomaco, in modo che nello stomaco arrivi *solo* quella parte del ramo *⟨azione se vero⟩* o *⟨azione se falso⟩* da eseguire a seconda dell'esito del test. In questo modo, se l'argomento del comando primitivo `\ifdefined`

è un comando condizionale, questo viene eseguito prima di vedere il resto della frase logica.

Quando l'esecuzione di uno dei tre programmi di composizione *pdf_latex*, *xel_latex* o *lua_latex* si blocca per un errore di difficile interpretazione, molto spesso la causa è un comando condizionale primitivo eseguito al momento sbagliato. Il L^AT_EX 3 Team sta facendo di tutto per eliminare tutti i comandi fragili, ma ancora il nucleo di L^AT_EX, così come i molti pacchetti delle distribuzioni del sistema T_EX che non siano stati aggiornati di recente, sono soggetti alla fragilità.

9 Definizione di alcuni comandi utili

9.1 Un salto pagina condizionale

Spesso componendo con L^AT_EX si ottengono delle pagine con gli spazi verticali esageratamente allargati; quasi sempre tale effetto sarebbe causato da un comando di sezionamento che richiede almeno 4 o 5 righe di spazio verticale e che in mancanza di quest'ultimo fa proseguire la composizione alla pagina successiva. L'effetto è sgradevole e quindi sarebbe utile prima del comando di sezionamento poter dare un comando del tipo `\goodpagebreak` che esegua i controlli al nostro posto. Basta che tale comando confronti quanto spazio esiste ancora sulla pagina con un numero specificato di righe:

```
\newcommand*\goodpagebreak[1][4]{\par
\ifdimgreater
  {\dimexpr\pagegoal-\pagetotal}%
  {\#1\baselineskip}%
  {\relax}{\newpage}}
```

Se sulla pagina è disponibile un numero di righe superiore al valore preimpostato o a quello specificato, allora il comando non esegue alcuna azione di sezionamento altrimenti esegue un fine pagina. Non voglio affermare che con i comandi nativi di T_EX o i comandi del pacchetto *ifthen* non si sarebbe potuto ottenere un risultato robusto, ma passando attraverso i comandi di *etoolbox* siamo sicuri che questo costruito è robusto.

9.2 Scalare un oggetto solo se eccede la giustezza

In questo esempio non faremo acrobazie di programmazione; cercheremo invece di confezionare un ambiente `resizedisplay` che scali alla giustezza corrente una figura o una tabella nel caso in cui le loro dimensioni naturali la eccedano.

Carichiamo il pacchetto *graphicx* e definiamo una nuova lunghezza e una nuova scatola in modo tale che il preambolo diventi:

```
\usepackage{graphicx}

\newlength\mydisplaywidth
\newbox{\mydisplay}
```

Poi definiamo il nuovo ambiente cominciando ad impostare la nostra nuova lunghezza al valore di `\columnwidth` o di `\textwidth` a seconda che si stia componendo a due o ad una colonna; non basta verificare la variabile booleana `@twocolumn`, perché la composizione a due colonne potrebbe essere eseguita dentro l'ambiente *multicols* senza che sia stata impostata la composizione a due colonne per l'intero documento. Ricordando che `\linewidth` rappresenta la giustezza corrente, cominciamo quindi a definire:

```
\newenvironment{resizedisplay}{%
\ifdimless{\linewidth}{\textwidth}%
{\setlength{\mydisplaywidth}{\columnwidth}}%
{\setlength{\mydisplaywidth}{\textwidth}}%}
```

Siccome dobbiamo misurare la larghezza dell'oggetto da scalare, lo componiamo dentro l'ambiente `lrbox`, che metterà l'oggetto dentro la scatola `\mydisplay`; continuiamo quindi la definizione dei comandi di apertura del nostro ambiente con il codice:

```
\begin{lrbox}{\mydisplay}
```

Ora l'ambiente è aperto e possiamo inserire la tabella (ambiente *tabular*) o la figura (`\includegraphics[...]{...}`). Inserito nell'ambiente quello che occorre, dobbiamo chiudere la scatola; in vista del fatto che l'ambiente potrebbe essere usato quando si è in modo verticale, dobbiamo eliminare l'eventuale rientro del capoverso. Perciò inseriamo subito:

```
{\end{lrbox}}\noindent
```

Poi dobbiamo provvedere ai controlli sulle larghezze e provvedere eventualmente a scalare l'oggetto. L'oggetto però ora è inscatolato dentro la scatola `\mydisplay` perciò dobbiamo controllare la larghezza di questa scatola; senza ricorrere i comandi L^AT_EX per misurare la larghezza degli oggetti (che verrebbero comunque messi dentro un'altra scatola per poter misurare le dimensioni di quest'ultima) usiamo i comandi nativi di T_EX ed eseguiamo il controllo con i seguenti comandi:

```
\ifdimless{\wd\mydisplay}{\mydisplaywidth}%
{\usebox{\mydisplay}}%
{\resizebox{\mydisplaywidth}{!}%
{\usebox{\mydisplay}}}%
\ignorespaces
}
```

Il comando `\wd`, seguito dal nome della scatola, ne restituisce la larghezza (*width*). Il comando `\usebox` è il comando L^AT_EX per usare il contenuto della scatola il cui nome compare nel suo argomento³.

Ora la sintassi per l'uso dell'ambiente è semplice:

3. Per approfondire le informazioni relative a questi comandi si può consultare la guida tematica (BECCARI, 2014a).

```
\begin{resizedisplay}
<oggetto da scalare>
\end{resizedisplay}
```

ma non bisogna abusarne. Se l'oggetto è decisamente troppo grande, il rimpicciolimento potrebbe essere eccessivo, tale da renderne illeggibile il contenuto. Quando poi l'oggetto è costituito da una o più equazioni in display, l'uso di questo ambiente richiede alcune cautele, come per esempio quella di racchiudere l'espressione matematica dentro un `minipage` di cui bisogna determinare la larghezza "sperimentalmente" altrimenti si rischia di rimpicciolire troppo il contenuto matematico. La `minipage` consente di comporre equazioni in display, cosa che l'ambiente `lrbox` non consente, ma richiede anche una certa disponibilità a procedere per tentativi visto che essa richiede comunque che le si specifichi la giustezza.

L'esempio dimostra che definire nuovi comandi e nuovi ambienti è sempre utile per risparmiare la scrittura di codice relativamente complesso ma del quale è richiesta la stesura senza il minimo errore e nella giusta sequenza.

L'esempio non è particolarmente complesso e, invece dei comandi di `etoolbox`, si sarebbero potuti usare anche i comandi del pacchetto `ifthen` o i comandi condizionali nativi di $\TeX$. Ma così facendo abbiamo la certezza della loro robustezza.

10 Conclusioni

Sono convinto che uno dei punti più delicati della programmazione di macro nei linguaggi che ci vengono offerti da $\TeX$, $\LaTeX$ e dai vari pacchetti di estensione sia proprio costituito dai comandi condizionali.

I modi per scrivere comandi condizionali, da usare sia all'interno di file di classe o di pacchetti personali, ma anche dentro macro personali definite nel preambolo dei propri documenti, sono diversi e diversamente utili; più che altro essi differiscono per la fragilità o robustezza che questi comandi presentano nelle varie circostanze.

La fragilità è commentata e descritta anche nel documento di GREGORIO (2009) di cui consiglio la lettura; capire la fragilità dei comandi condizionali è importante; sapere come fare per poter usare comandi meno fragili o, meglio ancora, decisamente robusti è ancora più importante.

Riferimenti bibliografici

BECCARI, C. (2014a). *Il $\LaTeX$ Reference Manual commentato*. GJr. Scaricabile dal sito www.guitex.org dalla sezione Documentazione, sottosezione Guide Tematiche.

— (2014b). *Introduzione all'arte della composizione tipografica con $\LaTeX$* . Gruppo utilizzatori

Italiani di $\TeX$ e $\LaTeX$. Scaricabile dal sito www.guitex.org, sezione Documentazione.

BEZOS, J. (2013). *Babel*. $\TeX$ Users Group. La versione 3.9h è disponibile con ogni distribuzione aggiornata del sistema $\TeX$ completa e leggibile con `texdoc babel`.

BREITENLOHNER, P. (1998). *The ϵ - $\TeX$ Manual*. The NTS Team, $\TeX$ Users Group. Disponibile con ogni distribuzione del sistema $\TeX$ completa e leggibile con `texdoc etex`.

CARLISLE, D. (2001). *The ifthen package*. $\TeX$ Users Group.

GREGORIO, E. (2009). «Appunti di programmazione in $\TeX$ e $\LaTeX$ ».

KNUTH, D. E. (1996). *The $\TeX$ book*. Addison Wesley, Reading, Mass., 16^a edizione.

LAMPORT, L. (1994). *$\LaTeX$: A document preparation system – User guide and reference manual*. Addison Wesley, 2^a edizione.

LEHMAN, P. (2011). *The etoolbox package*. $\TeX$ Users Group. Disponibile con ogni distribuzione del sistema $\TeX$ completa e leggibile con `texdoc etoolbox`.

NOIREL, J. (2009). *The xifthen package*. Disponibile con ogni distribuzione del sistema $\TeX$ completa e leggibile con `texdoc xifthen`.

▷ Claudio Beccari
Villarbasce
`claudio dot beccari at gmail.com`

4^{rs}TEXnica si rinnova

Claudio Beccari, Massimiliano Dominici, Gianluca Pignalberi

Sommario

Il rinnovo della direzione della nostra rivista è l'occasione per rivedere le scelte fatte in passato a proposito di formato e veste tipografica, ma anche per ciò che riguarda il processo di produzione del PDF per la stampa e per la versione online.

Abstract

With the new staff in charge of our magazine, it is a good moment to discuss and review the past technical choices concerning the format and the typographical design; it is even more important to review the production process of the final PDF documents for printing and the online distribution.

1 Lo stato attuale della rivista

4^{rs}TEXnica è appena entrata nel suo nono anno di età. Ideata durante il meeting del 2005, infatti, ha visto il suo primo numero uscire nell'aprile del 2006. I sei mesi intercorsi tra queste due date furono dedicati all'organizzazione della prima redazione, al disegno della veste tipografica e alla stesura del codice della classe. Veste tipografica e codice, rimasti sostanzialmente immutati nel corso di questi anni, eccezion fatta per alcune modifiche di scarso rilievo, scontano la relativa inesperienza, all'epoca, dell'autore, M. Dominici.

La gestione dei primi numeri avveniva in maniera totalmente manuale. Il redattore doveva creare due file master, uno per la versione a stampa e uno per la versione online, ciascuno con le corrette impostazioni; riportare nel preambolo i pacchetti caricati e i comandi definiti dagli autori dei singoli articoli risolvendo gli eventuali conflitti; assemblare infine gli articoli, a cui era stato preventivamente commentato il preambolo, tramite una serie di comandi `\include`. La compilazione delle due versioni era condotta anch'essa manualmente e, alla fine del processo, erano disponibili solo i due PDF contenenti l'intero numero. Anche gli abstract da inviare alla rivista gemella TUGboat erano estratti a mano (e per i primi due numeri addirittura tradotti sul momento, perché non era previsto un abstract in inglese).

In corrispondenza della quarta uscita, con l'esperienza dei primi tre numeri alle spalle, comincio a farsi sentire l'esigenza di rendere il processo di produzione della rivista più automatizzato. In particolare si erano presentati diversi casi di conflitti

tra le impostazioni dei diversi articoli, a volte di difficile risoluzione. Prendendo spunto da un articolo di Thierry Bouche apparso su TUGboat (BOUCHE, 2006), l'allora direttore Dominici tentò di mettere in funzione un meccanismo basato sul pacchetto `pdfpages` (MATTHIAS, 2013) e sull'uso di `--shell-escape` per compilare separatamente i singoli articoli e includere automaticamente i PDF risultanti. Fu subito evidente un problema: `pdfpages` non conserva gli elementi ipertestuali del formato PDF; ciò fa perdere irrimediabilmente tutti i collegamenti interni o esterni nella versione online. Questo problema venne in parte aggirato usando la libreria `iText`¹ per concatenare i vari elementi della rivista (articoli, indice, pagine di copertina, annunci, pagine riempitive, ecc.) nella sua versione online. `iText` infatti preserva tutti i collegamenti interni. I collegamenti dall'indice ai singoli articoli erano estratti dalla versione a stampa e reinseriti in quella online, usando dei piccoli programmi *ad hoc* basati sulla stessa libreria `iText`.

Il tutto risultava molto macchinoso, anche perché i programmi *ad hoc* erano fin troppo *ad hoc*: andavano riscritti numero per numero. Così, nel passaggio di gestione dalla direzione Dominici a quella Pignalberi, fu deciso di abbandonare completamente questa via: perché eseguire a mano ogni volta tutte quelle operazioni quando un computer può farle più velocemente e sempre allo stesso modo? Quanto segue è stato ispirato dalla precedente esperienza di Pignalberi a Free Software Magazine. Tony Mobily, il direttore, fece le stesse cose per automatizzare la produzione del PDF della rivista. Dunque il nuovo direttore di 4^{rs}TEXnica intraprese un processo di razionalizzazione della produzione della rivista, a partire dalla separazione del preambolo dell'articolo dal suo contenuto. Si ottenne così l'attuale *kit* di 4^{rs}TEXnica, in cui l'originale template è stato diviso in quattro sezioni, una contenente il solo preambolo comune a tutti gli articoli, un'altra i pacchetti caricati dall'autore, un'altra ancora i comandi definiti dallo stesso, e infine l'ultima contenente il testo vero e proprio dell'articolo.

In parallelo venne sviluppato uno script per *bash* che gestisse l'intera catena di compilazione delle versioni a stampa e online della rivista. Lo script in questione, ancora in uso, non si limita a invocare nel giusto ordine e numero i programmi richiesti per ottenere un PDF corretto (`latex`, `bibtex`, ecc.), ma si incarica anche di inserire ISSN e relativo

1. <http://itextpdf.com/>.

codice a barre (cfr. l'appendice A), data e numero corretti e di modificare alcuni elementi in base al tipo di uscita (regolare o meeting). Inoltre estrae gli abstract e li concatena in un unico file, e, per ogni articolo crea una voce bibliografica da inserire nel database BibTeX che contiene tutti gli articoli usciti su *ArsTeXnica*.

Pur trattandosi di un sistema che permette di non preoccuparsi di numerosi aspetti nella produzione della rivista, ora trattati automaticamente, l'attuale processo presenta limiti e difetti venuti a galla nel corso dell'ultimo passaggio di testimone alla direzione della rivista da Pignalberi a Beccari. I principali punti irrisolti sono tre:

1. Lo script funziona solo su sistemi operativi dotati di un interprete di comandi *bash*. Nessun problema, quindi, su Linux (sul quale è stato sviluppato), su Mac OS X e altri derivati di Unix. Per Windows, invece, è necessario aver installato l'ambiente Cygwin e in ogni caso nessun test è stato fatto per controllarne il funzionamento.
2. La compilazione è inefficiente. Ad ogni invocazione dello script tutti i passi vengono di nuovo compiuti, tranne poche eccezioni, anche nel caso in cui ciò non fosse necessario.
3. La gestione di eventuali conflitti tra le impostazioni dei singoli articoli è demandata al redattore. Un tipico esempio di questo problema è l'uso di differenti impostazioni per listings (HEINZ *et al.*, 2013) che rimangono in funzione anche dopo la fine dell'articolo cui si riferiscono. Per ovviare a questo inconveniente, Pignalberi aveva implementato il comando `\lststoredefault` che, come recita il nome stesso, ripristinava le impostazioni di base del pacchetto. Contattò anche l'autore di listings per suggerire l'inclusione del comando nel codice del pacchetto, ma senza successo.

2 La redazione

2.1 Versante umano

La redazione, intesa come insieme di persone, della rivista ha avuto finora una gestione "artigianale": il redattore capo (o direttore) riceve i contributi per email, li passa sempre per email a un revisore o al comitato scientifico, riceve per email le osservazioni, invia per email le osservazioni all'autore che le integra e reinvia l'articolo al direttore (lo avreste mai detto? per email). Quest'ultimo passa l'articolo a un correttore di bozze che compila una lista di correzioni (refusi e, a volte, forma) da ripassare al direttore. A seconda dei casi, e della fretta, tali correzioni vengono fatte d'ufficio o dall'autore. Insomma, avrete capito che in termini di tempo costa più la comunicazione tra autore e redazione (col direttore nel ruolo di hub — concentratore)

che realizzare l'articolo. Non parliamo della sanità mentale del concentratore...

2.2 Versante tecnologico

Durante il suo secondo mandato alla direzione di *ArsTeXnica*, Pignalberi aveva iniziato a considerare l'uso di Open Journal System,² software open source per la gestione di pubblicazioni accademiche con revisione tra pari (peer-reviewed) creato dal Public Knowledge Project. Tale software prevede l'uso di un database a scelta tra MySQL e PostgreSQL. L'adozione di questo programma avrebbe giovato: il concentratore sarebbe stato ora un programma e il direttore avrebbe riacquisito il suo ruolo di organizzatore di alto livello; non ci sarebbe stato più quello scambio disorganizzato di email né la necessità di usare un foglio elettronico per tenere aggiornato il calendario degli scambi di file e lo stato di avanzamento dei singoli articoli. Purtroppo l'allora direttore è stato vinto dalla sua indolenza e non è andato oltre la fase preliminare (cioè quella che precede l'installazione del programma) e ha continuato a lavorare forsennatamente "a mano", lasciando *ArsTeXnica* nel suo stato "artigianale".

2.3 Riorganizzazione dei compiti

"Storicamente", sul direttore gravavano tutti i compiti di gestione della rivista, dalla raccolta degli articoli all'"indirizzo" da imprimere alla singola uscita, dalla produzione del PDF ai contatti con gli stampatori. Questa situazione rischiava di rendere usurante l'incarico di direttore. Si è deciso allora di delegare una parte di questi compiti, anche approfittando della presenza in redazione dei due ex direttori, Dominici e Pignalberi, forti del loro bagaglio di esperienze pregresse. Il direttore è adesso libero da compiti tecnici e amministrativi e può concentrarsi sulla raccolta degli articoli e sulle decisioni che riguardano la linea editoriale vera e propria. Della produzione dei PDF per la stampa e per la versione online si occuperà Dominici, mentre a Pignalberi sono demandati i contatti con la tipografia. Delle spedizioni torna ad occuparsi Silvia Maschio.

3 Automatizziamo tutto

Gli autori dell'articolo si sono proposti di risolvere i tre punti elencati nel paragrafo 1 e dalla discussione sono emerse alcune valutazioni condivise:

1. Per superare il primo inconveniente è possibile ricorrere a un linguaggio di scripting disponibile su tutti i principali sistemi operativi. I primi linguaggi con queste caratteristiche che vengono in mente sono Python, Ruby, Lua. Un vantaggio rispetto ad un semplice script per *bash* o per il linguaggio batch di Windows
2. <http://pkp.sfu.ca/ojs/>.

è che quelli elencati sono linguaggi di programmazione completi e contengono già le funzioni adatte ad eseguire i compiti che nello script attuale sono demandati a programmi esterni,³ rendendo così il sistema più compatto e con meno dipendenze.

2. Per quanto riguarda l'efficienza della compilazione è bene affidarsi ad un sistema dedicato per l'automazione dello sviluppo (*build system*). `make`⁴ è sicuramente una scelta valida ma, dal momento che per ciascuno dei linguaggi citati in precedenza esiste l'implementazione di un *build system* (Python/SCons⁵, Ruby/rake⁶, Lua/lake⁷), è preferibile usare uno di questi in modo da rendere più stretta l'integrazione dei vari componenti. Tuttavia, per quanto riguarda la compilazione `latex` in senso stretto, data la sua complessità, è probabilmente meglio affidarsi a strumenti dedicati, come `latexmk` che si integra piuttosto bene con *build system* generici.
3. La soluzione ideale per evitare conflitti di impostazioni tra diversi articoli sarebbe la compilazione separata di ciascuno di essi e l'assemblaggio finale dei PDF così ottenuti con il pacchetto `pdfpages`. Questo pone alcuni problemi, il principale dei quali, come abbiamo visto nel paragrafo 1, è che non è possibile preservare i collegamenti ipertestuali. La soluzione scelta in passato di appoggiarsi a iText per l'assemblaggio è poco convincente anche perché introduce un'ulteriore dipendenza esterna. Fortunatamente le ultime versioni di $\TeX$ Live contengono il pacchetto `pax` che permette di estrarre e reintrodurre collegamenti ipertestuali.⁸ Da un primo test sembra che il pacchetto funzioni piuttosto bene, per cui la via che passa per `pdfpages` torna ad essere praticabile.

Oltre a questi punti principali che devono essere necessariamente risolti per avere un sistema efficiente e portatile, vi sono altre funzionalità che sarebbe utile aggiungere in quanto andrebbero a coprire aspetti della produzione della rivista attualmente scoperti o coperti solo in parte. Ad esempio, con lo script attuale è possibile estrarre

3. Naturalmente ci riferiamo alle operazioni aritmetiche e ad alcune manipolazioni di stringhe e di data; le esecuzioni di $\LaTeX$ e degli altri programmi vanno sempre effettuate tramite chiamate di sistema.

4. <https://www.gnu.org/software/make/>.

5. <http://www.scons.org/>.

6. <http://rake.rubyforge.org/>.

7. <http://stevedonovan.github.io/lake/topics/index.md.html>.

8. In realtà anche `pax` impone alcune dipendenze, in particolar modo Java e la libreria PDFBox (e Perl nel caso si voglia usare lo script `pdfannotextractor`). Tuttavia riteniamo che, essendo un componente di $\TeX$ Live, richieda un impegno minimo per ciò che riguarda installazione e uso.

e concatenare i sommari dei singoli articoli (e le loro traduzioni obbligatorie e facoltative fornite dall'autore) per essere trasmessi alla redazione di TUGboat. Tuttavia sarebbe molto conveniente ottenere automaticamente anche una loro versione HTML da inserire sulle pagine web della rivista, compito che viene, al momento presente, svolto a mano.

È poi ancora da valutare la possibilità di mettere i sorgenti degli articoli sotto controllo di versione usando un servizio web di hosting per lo sviluppo di progetti software. Questo permetterebbe di avere un archivio completo di tutti i numeri e degli strumenti per la loro compilazione. GIT possiede attualmente un account su Github, ma quest'ultimo concede repository privati solo a pagamento. Bitbucket, invece, un servizio simile a Github, prevede repository privati gratuiti con limitazioni al numero dei collaboratori che possono accedervi. Oltre agli evidenti vantaggi derivanti dall'uso di un repository online vi sono anche aspetti più controversi: la dipendenza da un nuovo programma (il *version control system* che si è scelto di usare) e il ricorso ad una forma di storage online, con tutte le note implicazioni in termini di privacy e di possibile discontinuità del servizio.

4 Una nuova veste tipografica?

Nelle discussioni svolte nella mailing list dello staff di Ar $\TeX$ nica negli anni passati si è discusso molto sul formato e la veste grafica della rivista. Mentre da un lato la veste grafica, i font, le immagini, gli articoli come gli editoriali, gli indici, eccetera sembrano raccogliere il consenso generale, il formato della pagina è stato discusso con più partecipazione.

Il dilemma è sostanzialmente se continuare a pubblicare la rivista in formato A4 oppure se passare al formato B5, o a formati di simili dimensioni.⁹

Ricordiamo che il rapporto altezza/larghezza dei fogli delle serie ISO (ISO216) è sempre pari a $\sqrt{2}$; ogni foglio di ciascuna serie è caratterizzato da un numero ed ha la superficie dimezzata rispetto al foglio caratterizzato dal numero precedente.

Per la serie A — non si parla di calcio perché la chimica esula dalle competenze della rivista —, il foglio A0 ha una superficie di 1 m^2 , perciò le sue dimensioni sono di 840 mm per 1190 mm; dividendo quattro volte per 2 il lato lungo dei fogli si arriva al comune foglio A4 di 210 mm per 297 mm. Questa particolare serie ha la proprietà che se n è il numero che caratterizza il formato, allora un pacco di 2^n fogli ha un peso uguale alla grammatura della carta, perché la superficie globalmente totalizzata dai fogli di quel pacco ammonta a un metro quadro.

Per la serie B, invece il formato B0 ha la base di 1000 mm e l'altezza di 1414 mm; dividendo il lato

9. Giova ricordare che le riviste scientifiche, sia quelle dell'IEEE che di altre organizzazioni, sono da anni in formato letter?

lungo per 2 cinque volte si arriva al formato B5 che ha le dimensioni di 176 mm per 250 mm.

Ecco quindi le differenze sostanziali. I fascicoli in formato A4 sono “troppo” alti per metterli in verticale negli scaffali di una libreria se i ripiani non sono particolarmente distanziati; perciò in generale è necessario metterli coricati. Invece i fascicoli in formato B5 si possono mettere in verticale nella maggior parte degli scaffali sia in casa sia in ufficio.

La pagina A4 contiene più materiale di quella B5, tanto che è opportuna la composizione su due colonne, con gli inconvenienti che questo tipo di composizione si porta dietro. Il formato B5 non richiede la composizione su due colonne, ma essendo di superficie minore il numero di pagine di un fascicolo aumenta; stimiamo che l'aumento possa essere dell'ordine del 40%.

È ovvio anche che le figure, le tabelle, e ogni altro oggetto fuori testo deve essere pensato per la composizione a una o due colonne per il formato A4, mentre non ci sono diversità per il formato B5.

Con il formato A4 si possono avere oggetti fuori testo sia a una sia a due colonne, ma quelli ad una colonna non possono venire collocati se si usa l'ambiente multicols (MITTELBAUGH, 2011), il quale però consente di pareggiare le colonne a fine articolo. Invece l'opzione di composizione twocolumns oppure il comando \twocolumns permettono di inserire oggetti mobili in colonna, ma non pareggiano le colonne dell'ultima pagina di ogni articolo. Il pacchetto balance (DALY, 1999), usato fino ad oggi, riesce talvolta a pareggiare le colonne, ma è necessario intervenire a mano con comandi appositi nei casi in cui ciò non avvenga, come per esempio quando nell'ultima pagina ci sia un oggetto mobile. Inoltre, se un articolo contiene sia immagini a una colonna sia immagini a giustezza piena, bisogna ricorrere al pacchetto fixltx2e (MITTELBAUGH *et al.*, 2006), perché l'attuale versione di L^AT_EX 2_ε si porta dietro un baco fin dai tempi della vecchia versione L^AT_EX 2.09.

Insomma, niente di terribile, ma con il formato B5 non si avrebbe nessuno di questi inconvenienti.

L'inconveniente maggiore rimane quello del formato della carta B5 (o B4) che si trova in commercio con grande difficoltà, e diventa necessario ricorrere alla carta in formato A4 (o A3) ritagliando l'eccesso e producendo un grosso sfrido; il maggior numero di pagine fa evidentemente lievitare il costo della rivista. Al momento di scrivere queste note si stanno richiedendo dei preventivi per valutare l'aumento dei costi. È possibile che presso certe tipografie la stampa venga eseguita su fogli molto più grandi da piegare quanto basta per produrre segnature in ottavo o in sedicesimo e in questo modo ci sarebbe molto meno sfrido e la carta costerebbe di meno rispetto all'equivalente quantità di carta già tagliata in formato A4 o B5.

Resta anche il problema del colore; stampare a

colori è particolarmente costoso, se non altro perché bisogna lavorare in quadricromia. Noi prevediamo di continuare a fare come ora, cioè di stampare la rivista per i soci, e di mettere a disposizione i file dei singoli articoli e dell'intera rivista sul sito www.guitex.org; la versione on line potrebbe essere tranquillamente a colori. Presumibilmente la versione online verrebbe letta a schermo e in questo caso la versione in formato B5 sarebbe anche preferibile, sia per la lettura sullo schermo di un PC sia sullo schermo di un dispositivo e-reader sufficientemente grande. Probabilmente la riduzione di scala su uno schermo verticale di 11 pollici di diagonale potrebbe addirittura essere più confortevole della lettura su uno schermo orizzontale tradizionale (dai 13,3 pollici in su).

Visto che molti frequentatori del Forum [guitex](http://www.guitex.org) già richiedono di poter leggere sui loro dispositivi hand-held (ormai conosciuti come tablet dopo l'obsolescenza del nome PDA¹⁰ che anche Steve Jobs diede al Newton di Apple¹¹), le varie guide e gli altri testi pubblicati dal nostro Gruppo, la soluzione B5 potrebbe accontentare molti lettori. Ma non possiamo certo richiedere agli autori di predisporre i loro articoli in modo che possano venire composti su due colonne in formato A4 e stampati in bianco e nero¹² per la stampa, e su una colonna a colori per la distribuzione online in formato B5.

Pensiamo che questi interrogativi svaniranno completamente quando disporremo dei preventivi; questo non vuol dire che non si realizzi la versione B5 se il costo è solo leggermente maggiore della versione A4; tuttavia sono decisioni che la redazione di *ArXiv* e il consiglio direttivo del [guitex](http://www.guitex.org) dovranno prendere quando avranno gli elementi a loro disposizione.

5 L'orizzonte temporale

La revisione del processo di produzione e della veste grafica della rivista è un compito complesso che richiede una pianificazione attenta e un periodo di tempo in cui sperimentare le varie soluzioni. Per quanto i due aspetti possano in linea teorica essere mantenuti separati, bisognerà tener conto di quanto del processo di automazione può essere delegato a L^AT_EX stesso nel riscrivere il codice della classe. Un esempio di ciò che si può ottenere senza ricorrere a strumenti esterni è, per esempio, l'esportazione degli abstract o, in generale, la scrittura su file esterni di metadati da riutilizzare in seguito, sia nella composizione vera e propria della rivista, sia per compiti accessori (per esempio, la

10. http://it.wikipedia.org/wiki/Computer_palmare.

11. http://it.wikipedia.org/wiki/Apple_Newton.

12. Un file PDF a colori stampato in tonalità di grigio perde buona parte della sua qualità perché le tonalità di grigio con cui sono rappresentati i colori sono difficilmente distinguibili e talvolta sono troppo chiare.

costruzione automatica di voci bibliografiche per i singoli articoli).

Tutto ciò richiederà naturalmente tempo, ma l'importante è che intanto le cose si siano messe in moto. È intenzione degli autori di questo articolo arrivare ad avere un sistema funzionante entro la data del prossimo meeting, in modo da poter presentare al pubblico del convegno e ai lettori del numero autunnale di *ArsTeXnica* le soluzioni effettivamente adottate, anche in termini di codice.

L'effettiva entrata in produzione è prevista, anche per una questione di coerenza all'interno del ciclo di uscite annuale, per il numero primaverile del prossimo anno, cioè il 19. Ci sarà quindi il tempo per aggiustamenti dell'ultima ora, anche sulla base delle reazioni e dei suggerimenti che raccoglieremo al meeting.

6 Conclusioni

Questo articolo rappresenta al momento più una dichiarazione di intenti che l'esposizione delle soluzioni ai problemi che ci siamo trovati davanti, in questi anni, nella gestione della rivista. Abbiamo deciso di rendere pubbliche queste intenzioni anche per coinvolgere i soci nella discussione e cominciare a raccogliere suggerimenti per migliorare ciò che siamo andati presentando in queste pagine. Potete contribuire, contattando gli autori dell'articolo o discutendone sulla mailing list del gruppo.

Un primo contributo in questo senso ci è già giunto da uno dei revisori dell'articolo, che ha suggerito la possibilità di gestire la raccolta e la pubblicazione degli articoli tramite un database (SQLite3, per esempio) in cui immagazzinare contenuto e metadati. Si tratterebbe di un ulteriore passo nella direzione di un'organizzazione più "industriale" della rivista, superando l'attuale fase "artigianale" e in tal senso la redazione è intenzionata a esplorarne la fattibilità, una volta raggiunti gli obiettivi più immediati sopra elencati.

A Dietro le barre

Sebbene *ArsTeXnica* abbia avuto il regolamentare numero di codice ISSN (International Standard Serial Number¹³), cioè il numero univoco che identifica i periodici (giornali e riviste), fin dalla nascita, l'adozione del codice a barre e la sua stampa sulla copertina sono arrivate solo col numero 6. Un codice a barre è la rappresentazione grafica di una sequenza di numeri e lettere; tale rappresentazione consta di barre verticali affiancate, trasparenti o colorate, di uguale altezza e spessore variabile. Con la locuzione "codice a barre" indichiamo la rappresentazione grafica ma non lo standard usato per ottenerla. I numeri ISSN, come pure

13. <http://www.issn.org>. L'organismo che assegna i codici ISSN in Italia è il Centro Nazionale ISSN, <http://www.bice.rm.cnr.it/Iissn.html>.

gli ISBN che identificano i libri, sono resi graficamente secondo lo standard EAN-13 (International Article Number; era European Article Number). Ci sono diversi standard di rappresentazione differenti principalmente per la quantità di informazione rappresentabile e per gli eventuali vincoli di rappresentazione.

La differenza sostanziale tra un codice ISBN e un ISSN sta in una piccola informazione aggiuntiva.¹⁴ Andiamo con ordine. Un ISSN è un numero di otto cifre (sette di identificativo e una di controllo) che identifica un periodico, dunque una *testata*. Allo stesso modo un ISBN (che di cifre ne ha dieci¹⁵) identifica un libro, cioè una specifica edizione, per esempio *Rumore bianco* di Tullio Pironti Editore nell'edizione del 1987 (e successive ristampe). È ovvio che una diversa edizione sarà identificata da un diverso ISBN. Il relativo codice a barre è formato da un blocco di informazioni ottenute dal numero descritto. Il codice a barre generato da un ISSN è diviso in due blocchi: il primo è ottenuto all'incirca come quello dell'ISBN, pur con una certa differenza di informazioni veicolate, il secondo è un blocco più corto che rappresenta due o cinque cifre ed è denominato *add-on*. Capire il perché dell'*add-on* è immediato: un periodico esce a intervalli regolari di tempo. Nominarne il titolo equivale a identificare l'entità "periodico", non una sua particolare istanza identificata dal numero. Non ci dilungheremo ulteriormente sulle regole di generazione del codice a barre a partire dall'ISSN.¹⁶

Il codice a barre stampato su *ArsTeXnica* viene generato da GNU barcode di Alessandro Rubini.¹⁷ Questo è un programma a riga di comando molto versatile che emette un output PostScript. Visto che la compilazione di *ArsTeXnica* viene eseguita da PDFIAT_{EX}, l'output PostScript viene convertito in PDF prima di essere inserito nel suo *scavo* (spazio predisposto). Tutti i lettori avranno altresì notato che il font senza grazie del codice a barre stampato su *ArsTeXnica* è diverso dal corrispondente usato nella rivista. Il primo è, infatti, Helvetica. Non ci siamo mai premurati di cambiarlo ma si può fare semplicemente importando il codice a barre in un programma di grafica vettoriale, per esempio Inkscape, selezionare la scritta e cambiare font (nel nostro caso bisogna avere installato CMSS in versione TTF/OTF sul proprio sistema).

14. Il codice ISBN identifica anche l'editore mentre l'ISSN no, ma non è questa la differenza che ci riguarda in questo ambito.

15. Il lettore attento avrà notato che gli ISBN hanno invece 13 cifre. In realtà le prime tre sono sempre 978 e identificano la merce "libro".

16. Una buona fonte di informazioni è il sito <http://www.bl.uk/bibliographic/issn.html>. Un'altra è <http://wwwcdf.pd.infn.it/AppuntiLinux/a2850.htm>. Entrambe sono comunque incomplete mancando di alcune nozioni sugli standard de facto degli add-on.

17. [http://www.gnu.org/software/barcode/..](http://www.gnu.org/software/barcode/)

Ora che abbiamo visto il come, spendiamo due secondi a capire il perché. È evidente che tutti sanno per esperienza (casse del supermercato, ufficio postale) che il codice a barre (e la sua evoluzione QR code) serve a far leggere velocemente a un computer la sequenza numerica tramite uno scanner. Qualcuno ancora pensa che le altre informazioni ottenute (prezzo e nome della merce) siano contenute nel codice stesso mentre invece questo è usato come chiave univoca di interrogazione di un database.

Riferimenti bibliografici

- BOUCHE, T. (2006). «A pdfL^AT_EX-based automated journal production system». *TUGboat*, **27** (1), pp. 45–50. Proceedings of EuroT_EX 2006.
- DALY, P. W. (1999). *Balancing the Two Columns of Text on the Last Page*. TUG. Leggibile con `texdoc balance` nella distribuzione TeX Live.
- HEINZ, C., MOSES, B. e HOFFMANN, J. (2013). *The Listings Package*. TUG. Leggibile con `texdoc listings` nella distribuzione TeX Live. L'attuale versione è affidata alla manutenzione di Jobst Hoffmann.
- ISO216 (2014). «ISO 216». http://it.wikipedia.org/wiki/ISO_216. Voce dell'enciclopedia libera online Wikipedia. Consultato il 2014/03/05.
- MATTHIAS, A. (2013). *The pdfpages Package*. TUG. Leggibile con `texdoc pdfpages` nella distribuzione TeX Live.
- MITTELBACH, F. (2011). *An environment for multicolumn output*. TUG. Leggibile con `texdoc multicol` nella distribuzione TeX Live.
- MITTELBACH, F., CARLISLE, D., ROWLEY, C. e SCHMIDT, W. (2006). *The fixltx2e and fix-cm packages*. TUG. Leggibile con `texdoc fixltx2e` nella distribuzione TeX Live.
- ▷ Claudio Beccari
claudio dot beccari at gmail dot com
 - ▷ Massimiliano Dominici
mlgdominici at gmail dot com
 - ▷ Gianluca Pignalberi
g dot pignalberi at alice dot it

Realizzare semplici animazioni in figure: come usare TikZ in Beamer

Claudio Fiandrino

Sommario

L'articolo illustra alcuni metodi per realizzare semplici animazioni per presentazioni in Beamer. In particolare, si focalizzerà l'attenzione su figure realizzate con TikZ sfruttando la compatibilità esistente fra la classe Beamer e l'omonimo pacchetto grafico.

Abstract

The aim of the article is to present methodologies to create simple animations for Beamer presentations. Specifically, TikZ-based pictures will be considered thanks to the interoperability between the Beamer class and the package itself.

1 Introduzione

La classe Beamer si pone oggi come prima scelta per la creazione di presentazioni con L^AT_EX (TANTAU *et al.*, 2013). Leggendone la documentazione, si può trovare un gran numero di consigli per strutturare un'ottima presentazione scientifica. L'autore, Till Tantau, ha ideato la classe in vista della sua dissertazione di dottorato e ha raccolto nel corso degli anni una serie di linee guida riassunte nel capitolo 5 *Guidelines for Creating Presentations* della documentazione. In merito alle animazioni, tali linee guida sono piuttosto categoriche e ne consigliano un utilizzo limitato e mirato, ad esempio la spiegazione progressiva dei vari passi di un algoritmo o di un sistema.

Quando le figure da realizzare sono semplici, conviene utilizzare direttamente L^AT_EX che offre numerosi pacchetti grafici. Uno di questi è TikZ, sviluppato dallo stesso autore della classe Beamer (TANTAU, 2013). TikZ è l'interfaccia di alto livello di PGF (Portable Graphic Format); quest'ultimo viene utilizzato in Beamer per la creazione degli elementi grafici come sidebar e blocchi. TikZ è sviluppato in modo modulare attraverso le librerie: alcune sono fondamentali in quanto costituiscono il nucleo stesso del pacchetto, altre possono essere utilizzate dall'utente all'occorrenza. Nel secondo caso, infatti, le funzionalità introdotte non influiscono sulla operatività del pacchetto. Le librerie ausiliarie devono essere caricate nel preambolo di un documento attraverso `\usetikzlibrary{lista delle librerie}` dove *lista delle librerie* è una lista separata da virgole.

Questo articolo presenta le diverse metodologie che permettono di creare semplici animazioni in figure sfruttando la compatibilità fra la classe Beamer e TikZ. Si illustreranno esempi molto semplici, con un massimo di due fotogrammi. Il lettore non percepisca questa scelta espositiva come limitazione intrinseca dei metodi presentati: utenti esperti possono anche realizzare animazioni relativamente complesse con gli strumenti che prenderemo in analisi.

È opportuno evidenziare che non si prenderanno in esame altri metodi per realizzare animazioni come quelli forniti dal pacchetto `animate`. Tale pacchetto crea animazioni basate su Javascript pertanto la soluzione non è supportata da tutti i visualizzatori di PDF. Le animazioni si definiscono attraverso il comando `\animategraphics` o l'ambiente `animateinline` e possono essere inserite in qualsiasi tipo di documento, non solo in presentazioni.

Il resto dell'articolo è così strutturato:

- la sezione 2 illustrerà brevemente come la classe Beamer gestisce le animazioni;
- la sezione 3 mostrerà in dettaglio gli strumenti standard di TikZ che “animano” le figure;
- la sezione 4 introdurrà il lettore a strumenti ausiliari, come lo stile `visible on` e il pacchetto `aobs-tikz` che estende gli strumenti forniti da TikZ (FIANDRINO, 2014);
- la sezione 5 conclude il lavoro.

2 Animazioni in Beamer

Un'ottima introduzione che spiega come realizzare animazioni con Beamer è contenuta nell'articolo di PIGNALBERI (2010). In questa sezione, tuttavia, si concentra e focalizza l'attenzione sugli aspetti tecnici delle animazioni.

Beamer permette di creare animazioni con numerosi comandi: `\pause`, `\onslide`, `\only`, `\uncover`, `\visible`, `\invisible`, `\alt` e `\temporal`. Ogni comando svolge naturalmente una funzione diversa, sebbene la caratteristica che tutti condividono è quella di determinare in quali fotogrammi appare o non appare un certo contenuto. Ogni quadro (*frame* in inglese) è caratterizzato da uno o più fotogrammi, anche chiamati diapositive (*slide* in inglese). La progressiva esposizione di un certo numero di fotogrammi

in sequenza è un'animazione. In Beamer si può creare e controllare un'animazione agendo su numero (chiamato *overlay specification*) e ordine dei fotogrammi. In termini di codice, ciò si traduce dichiarando il numero del fotogramma come primo argomento nei comandi che *supportano le animazioni*; tale argomento viene delimitato dai caratteri < e > per evitare confusioni con i classici delimitatori {, }, [e].

Si elencano di seguito alcune regole per definire *overlay specifications*. Al solito, si prendono in esame quelle rilevanti ai fini dell'articolo:

- se le parentesi uncinate contengono un solo numero, <4> ad esempio, l'azione del comando si realizza *soltanto* nel quarto fotogramma;
- se il numero è seguito da un “-”, come <3->, l'azione del comando è valida dal terzo fotogramma in avanti;
- se il carattere “-” separa due numeri, <1-3> ad esempio, l'azione del comando si verifica nel primo, secondo e terzo fotogramma;
- se due o più numeri sono separati da una virgola, come <1,3,4>, l'azione del comando ha effetto nel primo, terzo e quarto fotogramma (è conveniente proteggere in un gruppo la dichiarazione, ossia <{1,3,4}>);
- è possibile dichiarare espressioni complesse partendo dai costrutti sin qui elencati: <{1-3,5,7}> attiva l'azione del comando dal primo al terzo fotogramma e successivamente nel quinto e settimo.

I comandi precedentemente elencati non sono gli unici che *supportano le animazioni*, ma sono i comandi “nativi” di Beamer. Queste *macro base* permettono di definire altre macro in grado di gestire l'ordine di apparizione nei fotogrammi. Ad esempio, celebri comandi LATEX come \item e \includegraphics sono stati modificati in modo da poter controllare in quali fotogrammi appariranno le descrizioni degli elenchi o le figure. Ad esempio:

```
\includegraphics<2>{figura}
```

mostra la **figura** solo nel secondo fotogramma. Come è possibile tale magia? Il trucco è molto semplice. Occorre definire il nuovo comando tenendo conto di un argomento aggiuntivo che controlli i fotogrammi, in gergo “rendere il comando *overlay-aware*”. In genere si procede così:

```
\newcommand<>{\nuovo}[1]{%
  \comando#2{\funzione{#1}}%
}
```

Attraverso \newcommand<> si impone che il \nuovo comando in fase di definizione possa controllare in

quale fotogramma (argomento #2) svolgere la propria \funzione. \comando è una delle *macro base* riportate nell'elenco iniziale. Vediamo un esempio concreto. Con il seguente codice, si definisce un comando \rosso per fare comparire del testo con il colore rosso:

```
\newcommand<>{\rosso}[1]{%
  \only#2{\textcolor{red}{#1}}%
}
```

Successivamente, grazie a \rosso<2>{ciao} la parola *ciao* comparirà in rosso solo nel secondo fotogramma del quadro.

3 Il supporto nativo di TikZ

In TikZ il supporto per il controllo dei fotogrammi è garantito grazie alla seguente definizione:¹

```
\def\tikz@path@overlay#1{%
  \let\tikz@signal@path=\tikz@signal@path%
  % for detection at begin of matrix cell
  \pgfutil@ifnextchar<{%
    \tikz@path@overlayed{#1}}{\path #1}%
  }
\def\tikz@path@overlayed#1<#2>{%
  \path<#2> #1%
}
```

Agendo su \path, il comando base con cui si possono costruire segmenti e nodi, automaticamente anche \draw, \fill, \filldraw, \shade, \shadedraw e \node ereditano la caratteristica *overlay-aware*.

Prendiamo in esame un esempio pratico. Disegniamo con TikZ un cerchio visibile solo nel secondo fotogramma del quadro. Con il codice seguente si ottiene il risultato mostrato nelle figure 1a e 1b.

```
\documentclass{beamer}
\usepackage{lmodern}
\usepackage{tikz}
\setbeamertemplate{navigation symbols}{}

\begin{document}

\begin{frame}{Titolo}
Un elenco:
\begin{itemize}
\item a
\item b
\end{itemize}

\begin{tikzpicture}
\draw<2>[radius=2cm] (0,0)circle ;
\end{tikzpicture}
\end{frame}

\end{document}
```

1. Ringrazio Ilhan Polat per avere trovato la definizione nel file omnicomprendivo `tikz.code.tex`.

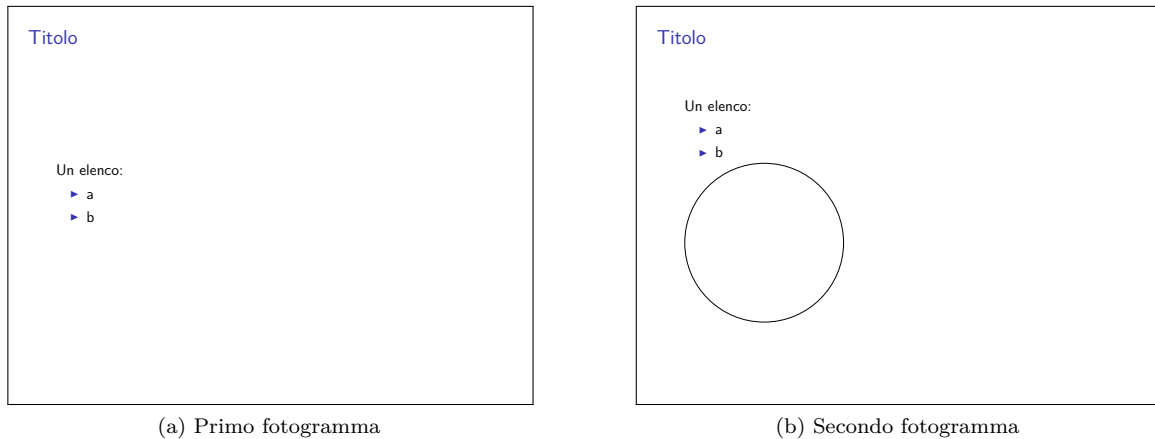


FIGURA 1: Un esempio di animazione con TikZ

Abbiamo quindi capito che i comandi nativi di TikZ sono in grado di eseguire la propria funzione, in determinati fotogrammi. Ma TikZ offre molto di più. La tecnica illustrata nell'esempio precedente è molto semplice e permette soltanto di far comparire degli elementi in determinati fotogrammi. In questo secondo esempio, vediamo come cambiare le proprietà di un elemento sempre visibile. TikZ permette di definire le proprietà di un elemento attraverso stili (PANTIERI e GORDINI, 2014): si tratta quindi di far assumere allo stesso stile proprietà diverse in fotogrammi diversi. Tutto ciò è possibile grazie al comando `\only`. Nelle figure 2a e 2b il cerchio presenta, partendo dall'alto verso il basso, una sfumatura dal rosso al bianco e dal bianco al rosso rispettivamente nel primo e secondo fotogramma.

```
\documentclass{beamer}
\usepackage{lmodern}
\usepackage{tikz}
\setbeamertemplate{navigation symbols}{}

\begin{document}
\begin{frame}{Titolo}

\begin{tikzpicture}
\only<1>{%
\tikzset{colora/.style={%
top color=red,
bottom color=white}}
}
\only<2>{%
\tikzset{colora/.style={%
top color=white,
bottom color=red}}
}
\draw[radius=2cm,colora] (0,0) circle ;
\end{tikzpicture}
\end{frame}

\end{document}
```

A questo punto, è chiaro come creare animazioni più complesse: combinando le due tecniche si possono far comparire gli elementi del disegno in determinati fotogrammi e alterarne le proprietà (naturalmente quando sono visibili).

4 Altri strumenti

I due metodi presentati nella sezione 3 rappresentano il supporto che intercorre automaticamente fra TikZ e Beamer. L'utente può contare su queste tecniche in ogni momento. Tuttavia, con l'utilizzo del primo metodo, può insorgere un problema chiamato *jumping effect*. I fotogrammi riportati nelle figure 1a e 1b, se visualizzati in sequenza, ben evidenziano l'inconveniente. Il cerchio, comparso nel secondo fotogramma, *sposta* la posizione che il testo occupava nella prima diapositiva: l'effetto percepito dal pubblico è quello di un salto, da qui il termine *jumping effect*. Tale fenomeno è dovuto al meccanismo interno con cui Beamer elabora il contenuto del quadro: ogni fotogramma è considerato indipendente dagli altri. Beamer, non ha nozione che lo stesso elenco è ripetuto e che il cerchio compare soltanto nella seconda diapositiva. Quindi, nel primo fotogramma non si tiene conto dello spazio che il cerchio andrà ad occupare.

Per ovviare a questo inconveniente esistono due alternative. La prima consiste nell'indicare espressamente a Beamer di tenere conto del contenuto di ogni fotogramma. A tal proposito è necessario utilizzare gli ambienti `overlayarea` e `overprint`. Il secondo approccio consiste nel disegnare *sempre* il cerchio e renderlo visibile solo negli opportuni fotogrammi. Questo metodo è stato implementato sotto forma di *stile* TikZ perciò può essere utilizzato direttamente nella figura. In termini di codice, si traduce nell'utilizzare l'opzione `opacity` con valore 0 per mantenere invisibili determinati elementi della figura. Ecco la definizione dello stile:²

2. L'autore ha sviluppato questo stile su TeX.SX: <http://tex.stackexchange.com/q/55806/13304>

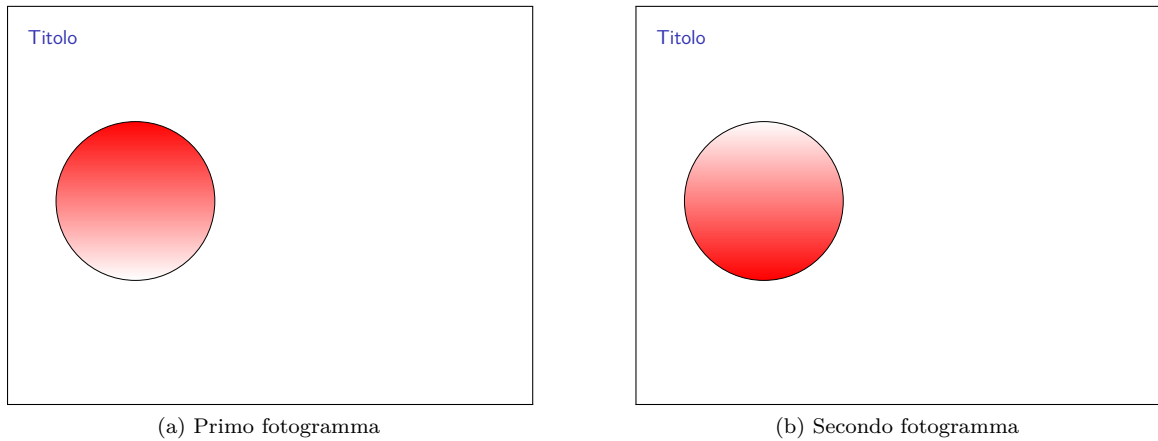


FIGURA 2: Animazione con alterazione delle proprietà di un elemento

```
\tikzset{
invisible/.style={opacity=0},
visible on/.style={%
alt={#1}{invisible}}
},
alt/.code args={<#1>#2#3}{%
\alt<#1>{%
\pgfkeysalso{#2}
}%
\pgfkeysalso{#3}}
},
}
```

Cambiando il codice riportato nel precedente documento:

```
\draw<2>[radius=2cm] (0,0)circle ;
```

con:

```
\draw[radius=2cm,
visible on=<2>] (0,0)circle ;
```

si ottiene il risultato mostrato in figura 3 senza il problema evidenziato.

In maniera del tutto analoga alla prima tecnica spiegata nella sezione 3, lo stile `visible on` permette soltanto di rendere visibili gli elementi della figura in alcuni fotogrammi. Recentemente, su CTAN è apparso un pacchetto che, mantenendo la stessa filosofia dello stile `visible on`, permette di alterare le proprietà di elementi nella figura in diversi fotogrammi (FIANDRINO, 2014). Il pacchetto, `aobs-tikz`, definisce la libreria `overlay-beamer-styles` con questo scopo. Le proprietà che possono venire alterate riguardano il colore del bordo, del riempimento e del testo, le sfumature e l'aspetto del segmento. Per creare un'animazione è necessario specificare:

- le proprietà di default dell'elemento;
- le modifiche alle proprietà di default;

- i fotogrammi in cui le proprietà modificate vengono utilizzate; automaticamente, negli altri fotogrammi si applicano le proprietà di default.

Nell'esempio seguente illustriamo l'utilizzo della libreria mettendone in luce uno dei suoi punti di forza. L'opzione `double` di TikZ permette di disegnare un segmento con linee parallele; dietro le quinte, l'opzione attiva una modalità

```
\tikz@addmode{\tikz@mode@doubletrue}%
```

che viene applicata globalmente al segmento. Una volta applicata, la modalità non può essere rimossa: in un'animazione, tuttavia, è necessario poterla disabilitare in determinati fotogrammi. `aobs-tikz` risolve questo problema introducendo l'opzione `double disabled`.

È importante notare che tutte le regole elencate nella sezione 2 per la dichiarazione delle *overlay specifications* sono valide sia per lo stile `visible on` sia per gli stili definiti da `aobs-tikz`. Con questi metodi, tuttavia, è *obbligatorio* proteggere in un gruppo tutte quelle definizioni che contengono delle virgole. Infatti, senza tale precauzione, il documento non compilerebbe in quanto la virgola è utilizzata dal parser di TikZ per separare opzioni diverse.

Le figure 4a e 4b illustrano un esempio in cui avviene uno scambio di informazione fra due nodi. Il nodo sorgente sarà contraddistinto dal colore azzurro mentre il nodo destinazione dal colore arancione e dal bordo con opzione `double` attiva. Grazie alla libreria, vediamo come realizzare una semplice animazione in cui nel primo fotogramma il nodo a sinistra è il mittente mentre nel secondo è il destinatario. Inoltre, rendiamo esplicito il senso della comunicazione indicando il verso con una freccia e denotando con un tratteggio la risposta al primo messaggio.

Il codice seguente rappresenta il documento completo dell'animazione in figura 4.

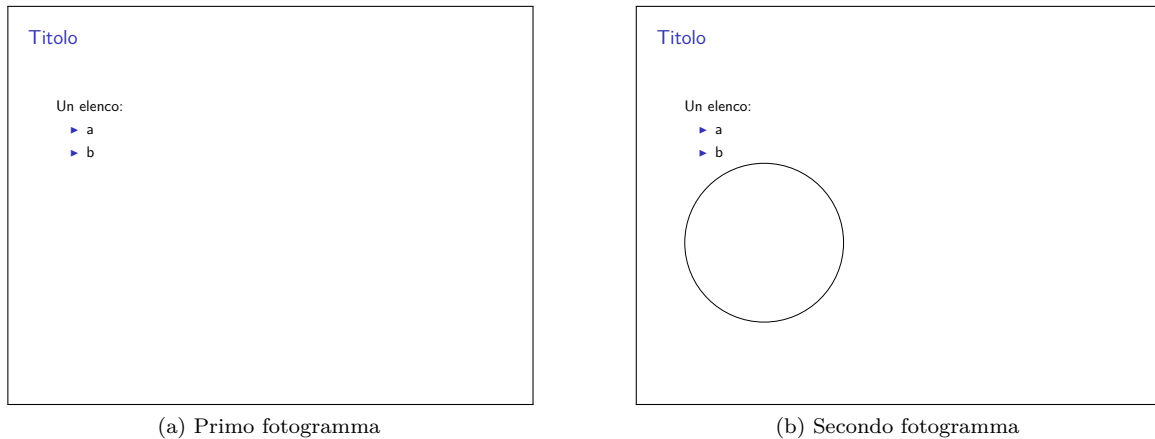


FIGURA 3: Animazione priva del problema jumping effect

```

\documentclass{beamer}
\usepackage{lmodern}
\usepackage{tikz}
\usetikzlibrary{overlay-beamer-styles}
\setbeamertemplate{navigation symbols}{}

\begin{document}

\begin{frame}{Titolo}
\begin{tikzpicture}[
elemento/.style={%
circle,
minimum size=2cm,
draw,
background default shade={%
top color=white,
bottom color=cyan!80!blue
},
background shade={
top color=white,
bottom color=orange
},
background default aspect={
double disabled
},
background aspect={
double distance=2pt
}
},
freccia/.style={
background default aspect={
-stealth, solid,
},
background aspect={
stealth-, dashed,
}
}
]
\node[elemento,
shade on=<2>,
aspect on=<2>
]

```

```
(A) at (0,0) {};
```

```

\node[elemento,
shade on=<1>,
aspect on=<1>,
]

```

```
(B) at (5,0) {};
```

```

\draw[ultra thick,
freccia,
aspect on=<2>,
] (A)--(B);
\end{tikzpicture}
\end{frame}

```

```
\end{document}
```

Analizziamolo passo a passo. Il codice principalmente consiste in due fasi:

1. la definizione degli stili `elemento` e `freccia`;
2. l'utilizzo degli stili nella figura.

Lo stile `elemento` è un cerchio con una dimensione prestabilita che viene colorato normalmente in azzurro (`background default shade`) e solo in alcuni fotogrammi in arancione (`background shade`). Inoltre, l'opzione `double` è normalmente disabilitata (`background default aspect`). Pertanto con:

```

\node[elemento,
shade on=<2>,
aspect on=<2>
]
(A) at (0,0) {};
```

stiamo definendo il nodo A come la sorgente poiché solo nel secondo fotogramma (`shade on=<2>` e `aspect on=<2>`) andrà ad assumere il colore arancione e il doppio tratto. Per il nodo B, vale il contrario: infatti le modifiche alle proprietà di default vengono attivate nel primo fotogramma (`shade on=<1>` e `aspect on=<1>`).

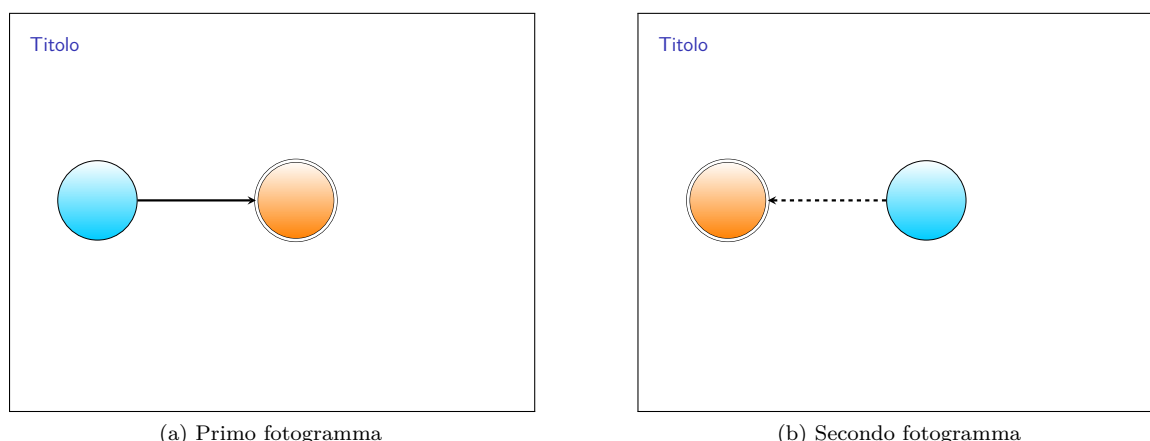


FIGURA 4: Animazione con la libreria `overlay-beamer-styles`

In conclusione, prendiamo in analisi lo stile **freccia**. Normalmente, il segmento non è tratteggiato e il verso della freccia punta verso la fine del segmento stesso. Il comportamento modificato, invece, inverte il verso della freccia e rende il segmento tratteggiato. Per questo motivo, la modifica è attivata nel secondo fotogramma con `aspect on=<2>`.

5 Conclusioni

L'articolo ha illustrato in modo semplice come utilizzare TikZ per realizzare animazioni con la classe Beamer sfruttando la compatibilità fra il pacchetto e la classe. Altre tecniche che permettono di realizzare animazioni con Beamer non sono state considerate. Sono stati presi in esame diversi metodi che hanno lo scopo di rendere visibili/invisibili alcuni elementi delle figure e di modificarne le proprietà.

Gli strumenti esistenti non permettono certo di realizzare animazioni mirabolanti. Tuttavia, sono più che sufficienti nel contesto di presentazioni scientifiche, nel corso delle quali le animazioni devono essere di supporto per chi espone e non uno strumento utilizzato per stupire il pubblico.

Riferimenti bibliografici

- FIANDRINO, C. (2014). *aobs-TikZ*. URL <http://www.ctan.org/pkg/aobs-tikz>. Consultabile con: `texdoc aobs-tikz`
- PANTIERI, L. e GORDINI, T. (2014). *L'arte di disegnare con L^AT_EX*. URL http://www.lorenzopantieri.net/LaTeX_files/ArteTikZ.pdf.
- PIGNALBERI, G. (2010). «Presentazioni animate in L^AT_EX». *ArsT_EXnica*, (10), pp. 33–40. URL <http://www.guit.sssup.it/arstexnica/>.
- TANTAU, T. (2013). *The TikZ and PGF Packages*. URL <http://www.ctan.org/pkg/pgf>. Consultabile con: `texdoc tikz`
- TANTAU, T., WRIGHT, J. e MILETIĆ, V. (2013). *The Beamer Class*. URL <http://www.ctan.org/pkg/beamer>. Consultabile con: `texdoc beamer`

▷ Claudio Fiandrino
claudio dot fiandrino at gmail dot com

Testatine e piè di pagina laterali

Gianluca Pignalberi

Sommario

Alcuni editori di libri hanno iniziato a pubblicare collane dall'aspetto inconsueto: i dati normalmente posti nelle testatine e/o nei piè di pagina sono ora posti lateralmente e ruotati. Vediamo un esempio di come ottenere lo stesso effetto con `LATEX`, `fancyhdr` e `TikZ`.

Abstract

Some books publishers started publishing books series that show an original look: those data normally positioned in the header or in the footer are now located on a side and rotated. We will show how to get the same result with `LATEX`, `fancyhdr` and `TikZ`.

1 Introduzione

Aprite un libro, uno qualsiasi. Molto probabilmente ritroverete quanto descritto nei prossimi due capoversi e non ne sarete affatto sorpresi, tant'è banale ciò che indicherò.

Fin da quando ho iniziato a leggere i primi libri di letteratura per ragazzi, ma ancor prima coi fumetti, ho sempre cercato la posizione dei numeri di pagina perché spesso non avevo un segnalibro e mi serviva sapere il punto di interruzione della lettura per riprenderla da lì (anche se da piccolo ricorrevo alle ora odiate “orecchie”, dato che non mi infastidiva sguaiare i libri). Ogni volta notavo che i suddetti numeri erano collocati in una delle tre posizioni seguenti: nell'angolo alto esterno sopra al testo, nell'angolo basso esterno sotto al testo oppure in basso al centro, sempre sotto al testo. Se ho incontrato posizioni diverse, non ne ho mantenuto il ricordo. Allora non sapevo la nomenclatura ufficiale delle posizioni e la mia ignoranza non ne risentiva.

Man mano che progredivo con le letture (libri di testo, dizionari, manuali, saggi) ho iniziato a notare che gli spazi prima e dopo il testo di ogni pagina includevano altre informazioni: numeri di capitolo, titoli, parole. Insomma, quegli spazi prima quasi muti diventavano improvvisamente parlanti, quasi logorroici. Passare dagli editor spartani (chi ricorda `edlin` o il meno ostico `edit` di MS-DOS?) ai primi elaboratori di testo mi permise di conoscere i termini *testatina* e *piè di pagina* (o *piédino*), e quindi di nominarli con cognizione di causa per la gioia di padre Paulus che cerca di insegnare a Shay i termini delle parti di una scarpa (DE LILLO, 1999, pagg. 572–580).

La familiarità con questi elementi e posizioni si è protratta per lungo, lunghissimo tempo, finché un giorno non mi è capitato tra le mani un libro pubblicato da Codice Edizioni¹ in cui l'informazione relativa al capitolo corrente non trovava più l'usuale collocazione nella testatina bensì al margine, con la scritta ruotata per farla “scorrere” parallelamente al lato lungo della pagina, così come mostrato nella nota al margine di questo capoverso.

Pur essendo già da qualche anno utente `LATEX` di medio livello, non riuscivo a figurarmi come ottenere lo stesso effetto col mio compositore preferito; o meglio, non riuscivo a capire come farlo automaticamente per tutte le pagine con lo stesso meccanismo usato da `LATEX` oppure dal pacchetto `fancyhdr` (VAN OOSTRUM, 2004).

Ho avuto lo stesso pensiero, questa volta più preoccupante, quando ho dovuto implementare un plugin per disegnare una nuova collana pubblicata dall'editore EDUSC (Edizioni Santa Croce):² il direttore voleva il numero di pagina posizionato dentro un cerchio al piè di pagina, esternamente alla gabbia del testo, con un filetto verticale che dipartiva dalla circonferenza e una serie di titoli paralleli al filetto stesso. Il prototipo fornitomi è visibile nella figura 1.

Il caso ha voluto che proprio nel periodo in cui cercavo una soluzione al design desiderato, al quale avrei aggiunto volentieri qualche altro elemento testuale e grafico, mi imbattessi nell'articolo di Claudio Beccari relativo alle filigrane (BECCARI, 2013). Questo articolo ha ispirato una soluzione, forse non economica o elegante ma sicuramente funzionale, alla mia necessità.

Nel seguito dell'articolo descriverò quanto fatto per ottenere le pagine dall'aspetto mostrato nella figura 2.

2 TikZ e le filigrane

Usare `TikZ` per fare il lavoro descritto nel presente articolo è senz'altro antieconomico: un pacchetto enorme e potentissimo per tirare due linee, centrarle in un cerchio e poco altro è uno spreco di risorse. Sicuramente avrei potuto ottenere dei risultati paragonabili usando `eso-pic` (NIEPRASCHK, 2010) oppure `watermark` (ROZHENKO, 2004), o addirittura l'ambiente `picture`. Verosimilmente avrei potuto ricorrere al meccanismo per il posizionamento dell'unghia nei dizionari descritto in BECCARI e

Nota al margine

1. <http://www.codiceedizioni.it/>.

2. <http://www.edizionisantacroce.it/>.

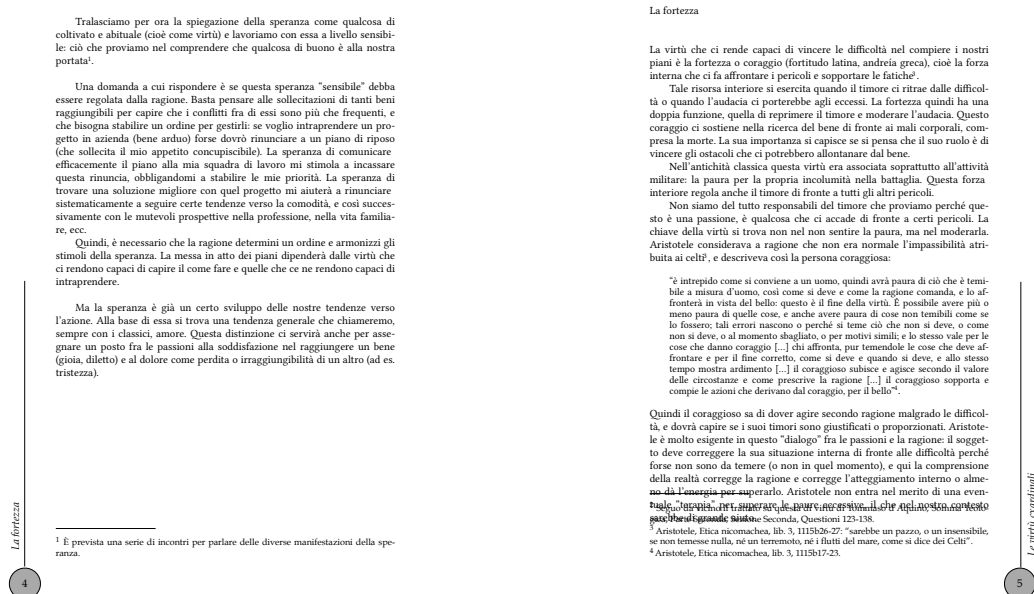


FIGURA 1: Prototipo dell'aspetto di una nuova collana di libri pubblicata dall'Editrice Santa Croce. La sovrapposizione del testo alle note nella pagina 5 è dovuta a un errore di conversione di Quartz per Mac OS X o, più probabilmente, a un'incompatibilità di questo col programma originale di creazione del prototipo.

FLECK (2012) o anche quello dei *thumb-indexes* del pacchetto fancyhdr (VAN OOSTRUM, 2004, pag. 20).

Il problema della produzione è annoso, dibattuto e apparentemente insolubile: «serve una soluzione e serve per ieri». Spesso è impossibile trovare il tempo necessario a individuare la soluzione migliore, quindi è più rassicurante ricorrere a uno strumento familiare. Per molto tempo ho disegnato schemi e altro con TikZ; tuttavia, quando ho visto come aggiungere delle filigrane senza difficoltà con lo stesso pacchetto ho afferrato la possibilità di non perdere il poco tempo a mia disposizione per terminare quella parte di lavoro. Ho ottenuto un risultato funzionale e bello per quanto riguarda l'aspetto, anche se sicuramente non ottimale dal punto di vista dell'impiego delle risorse di elaborazione.

Stabilito lo strumento che avrei usato, ho definito il tipo di elementi grafici, che d'ora in avanti chiamerò *decorazioni*, da realizzare. La lista completa comprende:

- una decorazione vuota, senza alcun elemento, per le pagine interamente bianche, i frontespizi e il colophon;
- una decorazione senza numero di pagina (sostanzialmente due filetti perpendicolari che

si incrociano nel punto dove sarà centrato il cerchio contenente il numero di pagina) per le pagine iniziali dei capitoli e per le pagine dispari dei capitoli non numerati.³ Tale decorazione è solo destra;⁴

- una decorazione come la precedente ma col numero di pagina bianco posto dentro un cerchio grigio. Tale decorazione sarà sia destra che sinistra e conterrà del testo di natura variabile:

- titolo del capitolo non numerato a sinistra;
- titolo del capitolo numerato, con relativo numero, a sinistra;
- autore del capitolo a destra.

I comandi definiti all'uopo sono, rispettivamente:

- `\noinfo`

3. I capitoli non numerati dei libri della collana in oggetto non hanno autore.

4. Con le abbreviazioni *destra* e *sinistra* indico rispettivamente le pagine dispari e pari. Poiché la decorazione è sempre posta all'esterno, si troverà esattamente nelle posizioni citate rispetto al testo.

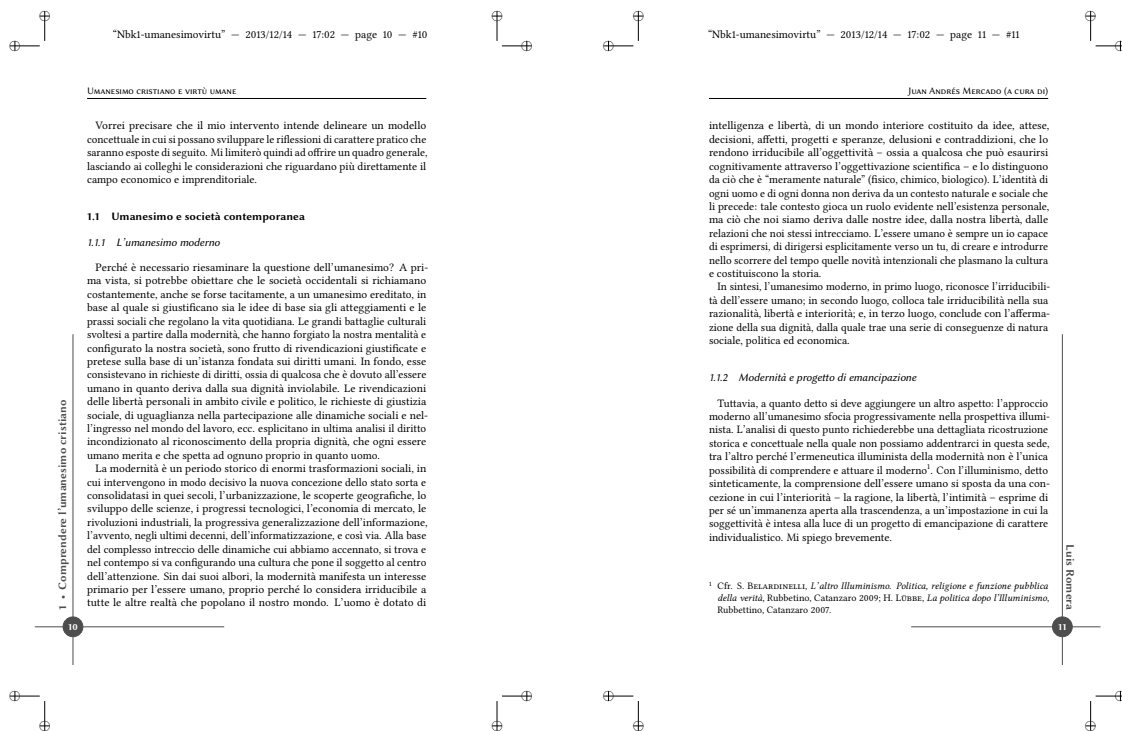


FIGURA 2: Aspetto finale del primo libro di una collana dell'Editrice Santa Croce. La presenza dei crocini di taglio è voluta e serve a rimarcare che il filetto orizzontale è *al vivo*, cioè dopo il taglio delle pagine termina esattamente sul margine della pagina.

- \whiteoddinfo⁵
- ...info
 - \evenfp...
 - \even...
 - \odd...

Vediamo dunque il codice usato per definire le decorazioni citate:

```
1 \RequirePackage{tikzpagenodes}
2 \RequirePackage[contents={}]{background}
```

configura l'ambiente da usare.

```
3 \newcommand{\noinfo}[1][{}]{%
4 \tikz[remember picture, overlay]{
5 }
6 }
```

definisce la decorazione vuota.⁶ Notate alla riga 4 l'opzione `remember picture` passata a `\tikz`. Questo richiede una doppia compilazione affinché l'effetto voluto possa essere ottenuto.

5. Un'analoga `\whiteeveninfo` è stata prevista per classi diverse da `book`. Qui non è riportata perché esaminiamo la composizione di un libro.

6. In mancanza di una decorazione vuota ritroveremmo l'ultima decorazione usata anche su una pagina vuota.

```
7 \newcommand{\whiteoddinfo}[1][{}]{%
8 \tikz[remember picture, overlay]{
9 \coordinate [xshift=9.5cm,yshift=-5.7
10 cm] (centro) at (current page.
11 center);
12 \draw[color=black!70] (centro) --
13 +(-5.1,0);
14 \draw[color=black!70] (centro) --
15 +(0,10.1);
16 \draw[color=black!70] (centro) --
17 +(1.1,0);
18 \draw[color=black!70] (centro) --
19 +(0,-1.1);
20 }
```

realizza la decorazione delle pagine iniziali dei capitoli e delle pagine destre dei capitoli non numerati. Notate che il comando non disegna due linee ma quattro. È stato un modo comodo per allungare le linee che nelle decorazioni coi numeri di pagina partono dalla circonferenza e non dal centro del cerchio per non trasparire sotto i numeri bianchi. Sia in questo che negli altri comandi il posizionamento della decorazione avviene in coordinate relative al centro della pagina.

```
16 \newcommand{\oddinfo}[1][{}]{%
17 \tikz[remember picture, overlay]{%
18 \coordinate [xshift=9.5cm,yshift=-5.7
19 cm] (centro) at (current page.
```

```

        center);
19 \coordinate [xshift=-3.5mm] (a) at (
    centro);
20 \coordinate [yshift=3.5mm] (b) at (
    centro);
21 \coordinate [xshift=3.5mm] (c) at (
    centro);
22 \coordinate [yshift=-3.5mm] (d) at (
    centro);
23 \filldraw [color=black!70] (centro)
    circle (3.5mm);
24 \node [font=\bfseries\footnotesize,
    color=white,#1] at (centro) {\
    textsf{\thepage}};
25 \draw [color=black!70] (a) -- +(-5,0);
26 \draw [color=black!70] (b) -- +(0,10);
27 \draw [color=black!70] (c) -- +(1,0);
28 \draw [color=black!70] (d) -- +(0,-1);
29 \node [color=black!70,above left,
    yshift=1mm,rotate=270] at (b) {\
    textbf{\textsf{\autname}}};
30 }%
31 }
32 \newcommand{\eveninfo}[1][{}]{%
33 \tikz[remember picture, overlay]{%
34 \coordinate [xshift=-4.5cm,yshift
    =-5.7cm] (centro) at (current
    page.center);
35 \coordinate [xshift=3.5mm] (a) at (
    centro);
36 \coordinate [xshift=-3.5mm] (c) at (
    centro);
37 \coordinate [yshift=-3.5mm] (d) at (
    centro);
38 \filldraw [color=black!70] (centro)
    circle (3.5mm);
39 \node [font=\bfseries\footnotesize,
    color=white,#1] at (centro) {\
    textsf{\thepage}};
40 \draw [color=black!70] (a) -- +(5,0);
41 \draw [color=black!70] (b) -- +(0,10);
42 \draw [color=black!70] (c) -- +(-1,0);
43 \draw [color=black!70] (d) -- +(0,-1);
44 \node [color=black!70,above right,
    yshift=1mm,rotate=90] at (b) {\
    textbf{\textsf{\thechapter}\
    textbullet~\chname}}};
45 }%
46 }

I due precedenti comandi sono quelli relativi alle
decorazioni dei capitoli numerati. Notiamo la pre-
senza delle istruzioni per disegnare i cerchi grigi
contenenti il numero di pagina bianco e per posizio-
nare le scritte coi contenuti corretti parallelamente
ai filetti verticali.

47 \newcommand{\evenfpinfo}[1][{}]{%
48 \tikz[remember picture, overlay]{%
49 \coordinate [xshift=-4.5cm,yshift
    =-5.7cm] (centro) at (current
    page.center);
50 \coordinate [xshift=3.5mm] (a) at (
    centro);
51 \coordinate [yshift=3.5mm] (b) at (
    centro);
52 \coordinate [xshift=-3.5mm] (c) at (
    centro);
53 \coordinate [yshift=-3.5mm] (d) at (
    centro);
54 \filldraw [color=black!70] (centro)
    circle (3.5mm);
55 \node [font=\bfseries\footnotesize,
    color=white,#1] at (centro) {\
    textsf{\thepage}};

```

```

56 \draw [color=black!70] (a) -- +(5,0);
57 \draw [color=black!70] (b) -- +(0,10);
58 \draw [color=black!70] (c) -- +(-1,0);
59 \draw [color=black!70] (d) -- +(0,-1);
60 \node [color=black!70,above right,
    yshift=1mm,rotate=90] at (b) {\
    textbf{\textsf{\chname}}};
61 }%
62 }

```

Questo comando varia rispetto al precedente `\eveninfo` solo per il contenuto della stringa parallela al filetto: non c'è il numero del capitolo perché questa decorazione è quella relativa ai capitoli non numerati.

```

63 \def\pagetype{}
64 \def\frontpages{frontpages}
65 \def\mainpages{mainpages}
66 \def\emptypages{emptypages}

```

Queste ultime definizioni servono per i controlli sul tipo di pagina corrente necessari a selezionare la decorazione corretta. Ne discuteremo nella prossima sezione.

3 Posizionamento condizionale con `fancyhdr`

Come stabiliamo quale delle decorazioni usare nella pagina corrente? Il meccanismo più semplice che ho individuato è stato “agganciarmi” alle funzioni di `fancyhdr` che uso per le testatine. Queste ultime contengono, a seconda delle pagine, il titolo del testo, il curatore del testo oppure niente. Aggiungere una definizione sul tipo di pagina proprio nelle definizioni degli stili di `fancyhdr` è stato immediato (righe 2, 9, 16 e 21 del listato seguente).

```

1 \fancypagestyle{MCEnb}{%
2 \def\pagetype{mainpages}
3 \fancyhf{}
4 \fancyhead [RE]{\hspace{-2.5cm}\begin{
    tabular}{@{}p{12cm}@{}}\footnotesize
    {\textsc{\textsf{\@title}}}\hline\
    end{tabular}}
5 \fancyhead [L0]{\begin{tabular}{@{}p{12cm}
    @{} }\hfill\footnotesize{\textsc{\
    textsf{\@author}}}\hline\end{
    tabular}}
6 }
7
8 \fancypagestyle{front}{%
9 \def\pagetype{frontpages}
10 \fancyhf{}
11 \fancyhead [RE]{\hspace{-2.5cm}\begin{
    tabular}{@{}p{12cm}@{}}\footnotesize
    {\textsc{\textsf{\@title}}}\hline\
    end{tabular}}
12 \fancyhead [L0]{\begin{tabular}{@{}p{12cm}
    @{} }\hfill\footnotesize{\textsc{\
    textsf{\@author}}}\hline\end{
    tabular}}
13 }
14
15 \fancypagestyle{plain}{%
16 \def\pagetype{frontpages}
17 \fancyhf{}
18 }
19
20 \fancypagestyle{empty}{%

```

```

21 \def\pagetype{emptypages}
22 \fancyhf{}
23 }

```

Manca ora il meccanismo per capire, per ogni nuova pagina, di che tipo di pagina si tratti. Le poche righe seguenti ci mostrano che sfruttare il comando `\AddEverypageHook{}` fornito dal pacchetto `everypage` (CALLEGARI, 2007) rende facile selezionare la giusta decorazione in base al controllo sul tipo e sulla parità/disparità della pagina. Il codice seguente traduce i casi elencati alle pagine 25–26.

```

1 \AddEverypageHook{%
2 \ifx\pagetype\frontpages
3 \ifodd\value{page}\whiteoddinfo\
  BgMaterial\else%
4 \evenfinfo\BgMaterial\fi\else%
5 \ifx\pagetype\emptypages\noinfo\
  BgMaterial\else%
6 \ifodd\value{page}\odddinfo\
  BgMaterial\else\eveninfo\
  BgMaterial\fi\fi\fi}

```

Sono convinto che sia possibile migliorare lo stile e l'efficienza del codice del plugin, ma nel poco tempo a mia disposizione mi sono accontentato della versione descritta che svolge adeguatamente il compito prefissato.

4 Note di progettazione

Ora che abbiamo visto il codice, osserviamo brevemente quali considerazioni mi hanno portato a ottenere l'aspetto finale della collana.

Ho avuto dal direttore di collana le misure della pagina e della gabbia di testo e le indicazioni sul font e i corpi da usare. I rientri, le misure dei filetti e l'aspetto dell'indice derivano dalla collana madre, di cui quella mostrata è uno *spin-off*.

I miei gradi di libertà sono stati la definizione dell'aspetto dei titoli dei capitoli con la relativa colorazione in sostituzione del corsivo — argomento non toccato nell'articolo — e la definizione delle decorazioni a partire dal prototipo.

In base alle misure della pagina e della gabbia, e di conseguenza dei margini, ho stimato che i filetti sarebbero stati distanti dal testo mezzo centimetro, quello verticale sarebbe stato lungo 11,5 cm e quello orizzontale 6,5 cm; le parti corte invece sarebbero terminate a 1,3 cm dal punto di incrocio (spazio filetto verticale-bordo pagina), punto dov'è centrato il cerchio col numero di pagina. Tale cerchio, del diametro di 7 mm, è dimensionato per contenere numeri di tre cifre, anche se la collana difficilmente prevederà libri più lunghi di cento pagine. Il font delle scritte è Linux Biolinum (cioè Linux Libertine Sans Serif); i numeri sono bianchi mentre gli altri elementi delle decorazioni sono grigi (nero al 70%). Infine le scritte distano un millimetro dai filetti (lo spazio è misurato dal filetto al discendente e non alla linea di base, ma non sono convinto dell'eleganza di questa soluzione).

Rimane da segnalare un bug sulle misure riportate nel comando `\whiteoddinfo`: i filetti risultano leggermente più corti del dovuto e dovrò rivedere le misure.

5 Conclusioni

Alcuni testi di aspetto moderno iniziano a porre lateralmente quegli elementi extratestuali normalmente posti nelle testatine o nei piè di pagina: titoli, numeri di pagina, autori. Fare la stessa cosa con L^AT_EX è un compito entusiasmante perché non c'è un'unica soluzione da poter adottare. In questo articolo abbiamo discusso di una tra le tante possibili: usare `fancyhdr` per selezionare il tipo di aspetto e `TikZ` per la sua possibilità di posizionare delle filigrane, oltre che per la sua potenza grafica in generale. La combinazione di questi due pacchetti permette di dare un aspetto più moderno ai propri documenti con un minimo sforzo programmatico. In termini di riusabilità del codice, invece, sappiamo di poter elaborare gli elementi delle decorazioni all'infinito e con relativa semplicità, pertanto l'investimento iniziale, irrisorio in termini di tempo, sarà fruttuosissimo quando bisognerà adottare la stessa soluzione in altri contesti (per esempio un testo con decorazioni graficamente più elaborate).

Riferimenti bibliografici

- BECCARI, C. (2013). «Le filigrane e le figure di sfondo». *ArsTeXnica*, (15), pp. 46–58.
- BECCARI, C. e FLECK, H. (2012). «Una classe per comporre un dizionario». *ArsTeXnica*, (14), pp. 31–40.
- CALLEGARI, S. (2007). «The `everypage` package». URL <http://www.ctan.org>. Lo leggete dando `texdoc everypage` al prompt del terminale o di DOS.
- DELILLO, D. (1999). *Underworld*. Einaudi.
- NIEPRASCHK, R. (2010). «The `eso-pic` package». URL <http://www.ctan.org>. Lo leggete dando `texdoc eso-pic` al prompt del terminale o di DOS.
- ROZHENKO, A. I. (2004). «The `watermark` package». URL <http://www.ctan.org>. Lo leggete dando `texdoc watermark` al prompt del terminale o di DOS.
- VAN OOSTRUM, P. (2004). «Page layout in L^AT_EX». URL <http://www.ctan.org>. Lo leggete dando `texdoc fancyhdr` al prompt del terminale o di DOS.

▷ Gianluca Pignalberi
g dot pignalberi at alice dot
it

Un primo sguardo al modulo publications

Luigi Scarso

Sommario

Presentiamo il modulo ConTeXt publications dedicato alla gestione della bibliografia. Sono illustrate le componenti interne e viene descritto un metodo per gestire il formato dei file utilizzato da Reference Manager® e EndNote® della Thomson Reuters.

Abstract

We introduce the ConTeXt module publications for the management of the bibliography. We describe the components and how to implement a support for the format of the files used by Reference Manager® and EndNote® from Thomson Reuters.

1 Introduzione

Nel Novembre del 2013 l'Institut für Klassische und Romanische Philologie - Abteilung für Griechische und Lateinische Philologie dell'Università di Bonn (<http://www.philologie.uni-bonn.de/>) ha lanciato il progetto di edizioni critiche con ConTeXt-MKIV. Diretto dal Prof. Dr. Thomas A. Schmitz, direttore tecnico dell'Istituto, il progetto vede Hans Hagen come capo sviluppatore e l'autore come secondo sviluppatore ed intende avvalersi delle caratteristiche di L^AT_EX, attualmente uno dei motori tipografici T_EX tra i più avanzati, sviluppando e modificando se necessario alcune parti di ConTeXt-MKIV per soddisfare i requisiti richiesti. È probabile che il progetto porti ad un modulo incluso nella distribuzione *standalone* (http://wiki.contextgarden.net/ConTeXt_Standalone) ma non nel formato, quindi da caricare a richiesta. È altresì possibile, dato che H. Hagen e l'autore appartengono al team di sviluppo di L^AT_EX, che alcuni requisiti del progetto influenzino lo sviluppo del motore di impaginazione qualora si rivelassero colli di bottiglia, ma bisogna tenere presente che nel caso di L^AT_EX, contrariamente a ConTeXt, la tendenza attuale è quella di delegare quanto più possibile al formato esterno concentrandosi ad ottimizzare il codice C per migliorare le prestazioni, soprattutto la riduzione del tempo di esecuzione.

Sebbene le edizioni critiche siano un argomento specialistico, in ambito T_EX è ben affrontato ed esiste una buona letteratura in merito. Senza la pretesa di voler essere esaurienti e limitandoci solo ai contributi italiani, ricordiamo il "Il progetto Maurolico" (<http://www.maurolico.unipi.it/mtex/mtex.htm>), il cui inizio è data-

to Dicembre 2000 (cfr. <http://www.maurolico.unipi.it/comite.htm>), il CeT_EX, "Corso di Edizione Critica al computer con T_EX/LaT_EX" (<http://www.didaskalikos.org/CeTeX/>) del Prof. J. Leal della Pontificia Università della Santa Croce, il testo "Edizioni Critiche" (LEAL e PIGNALBERI, 2012) e, più recentemente, i due articoli (FADINI, 2012) e (JERÓNIMO LEAL AND GIANLUCA PIGNALBERI, 2012) presentati al GUIT meeting 2012. Oggi si tende a utilizzare X_YL^AT_EX con il pacchetto ledmac oppure il più recente eledmac (cfr. <http://geekographie.maieul.net/Ledmac-est-mort-vive-Eledmac>), mentre i package tikz e xypic sono frequentemente usati per i stemma codicum.

La bibliografia è un aspetto fondamentale di una edizione critica, ma si tratta di un argomento meno settoriale — praticamente ogni articolo, report o libro sia in ambito scientifico che in ambito umanistico riporta una bibliografia minima. Anche in questo caso esiste un package di riferimento, biblatex, spesso associato al motore "bibliografico" biber (cfr. <http://biblatex-biber.sourceforge.net/>) che offre completo supporto a UNICODE ed una gestione della memoria molto migliore rispetto al tradizionale bibtex; inoltre è in grado di mostrare graficamente le relazioni tra le voci di una bibliografia (*cross-references*) mediante il programma GRAPHVIZ.

È innegabile che in questo ambito ConTeXt-MKIV non gode attualmente della stessa popolarità di L^AT_EX e non vi sono quindi moduli specifici per le edizioni critiche, anche se in passato alcuni tentativi sono stati fatti in tal senso.¹ ConTeXt ha anche un proprio modulo per le bibliografie, ma è piuttosto essenziale. Il progetto vuole quindi "colmare" questa lacuna, tenendo certamente ben presente le funzionalità offerte dai packages di L^AT_EX ma con alcune decisioni precise:

- per le edizioni critiche, il formato XML-TEI come base di partenza per il testo;
- il formato bib di BIBT_EX per le bibliografie, ma non il formato bbl — questo implica che le macro L^AT_EX non sono riconosciute;
- al fine di ridurre le dipendenze, l'utilizzo esclusivo dei programmi

1. *Steps needed to expand ConTeXt to the demands of Critical Edition typesetting*, I.S. Hamid and T.A. Schmitz I° ConTeXt meeting 2007, <http://meeting.contextgarden.net/2007/share/idris/cr-apparatus.pdf>

LUA_{TEX}/LUAJIT_{TEX},²METAPOST e Con_{TEX}t.

L'ultimo punto probabilmente merita ulteriori spiegazioni, in quanto discende direttamente da una sostanziale differenza tra i due formati L_AT_EX e Con_{TEX}t: il primo ha un kernel relativamente semplice e delega le funzionalità particolari a packages esterni; il secondo tende viceversa ad inglobare le funzionalità nel formato, o comunque in moduli caricali a tempo di esecuzione ma presenti nella distribuzione standalone, ed utilizza esclusivamente LUA come linguaggio di programmazione general-purpose. Ad esempio, un programma come BIBER verrebbe realizzato in LUA con il duplice vantaggio sia di renderlo immediatamente disponibile per tutte le piattaforme che supportano L_AT_EX, sia di riusare il codice LUA già presente in Con_{TEX}t. Inoltre biber è, sostanzialmente, un interprete PERL staticamente linkato con il sorgente PERL di BIBER ed i relativi moduli per un totale di circa 19 MiB, ai quali vanno sommati circa 19 MiB di X_YL_AT_EX; per contro la dimensione di L_AT_EX è di circa 8 MiB.³

Allo stato attuale la parte di edizioni critiche è nella sua fase iniziale, mentre il modulo relativo alla bibliografia, scritto ex-novo, è disponibile nella distribuzione standalone — e lo sarà anche nella prossima T_EXLive 2014, a meno di imprevisti: si tratta infatti di un modulo ancora in fase di sviluppo, quindi non “production-ready”, e questo implica che è ancora possibile valutare richieste e/o suggerimenti da parte della comunità.

Nelle prossime sezioni presenteremo brevemente il modulo e una estensione proposta dall'autore per gestire il formato RIS utilizzato dal sistema Reference Manager di Thomson Reuters.

2 Il modulo publications

2.1 Datasets

In Con_{TEX}t il modulo relativo alla bibliografia è chiamato `publications` e si basa attualmente su B_IB_TE_X (per questo motivo le macro utilizzano lo spazio dei nomi `btx`). Il file viene caricato interamente in memoria e convertito in una `table` di LUA, utilizzata come base per le successive elaborazioni. La prima osservazione riguarda il fatto che in un file `bib` normalmente sono presenti macro di L_AT_EX non implementate in Con_{TEX}t od implementate in modo diverso. In questo caso l'elaborazione *non* si ferma, ma viene emesso su log un messaggio opportuno:

```
publications > start used btx commands
publications > standard CONTEXT 1 known
publications > standard ConTeXt 4 known
publications > standard TeXLive 3 KNOWN
publications > standard eTeX 1 known
```

2. LUAJIT_{TEX} dovrebbe essere distribuito con la T_EXLive 2014

3. LUAJIT_{TEX} ha una dimensione di circa 8.5 MiB. Le dimensioni si riferiscono ad eseguibili Linux.

```
publications > standard hbox 6 known
publications > standard sltt 1 unknown
```

La macro non implementata viene inoltre stampata nel documento con un carattere a spaziatura fissa, per essere più facilmente rintracciabile durante la correzione delle bozze. Una macro può essere (ri)definita con il comando `\definebtxcommand`, come in

```
\definebtxcommand\TUB {TUGboat}
\definebtxcommand\sltt{\tt}
```

Ad esempio, il seguente file `bib`

```
@Article{sometag,
  author = "An Author and Another One",
  title = "A hopefully
\sltt{meaningful} title",
  journal = maps,
  volume = "25",
  number = "2",
  pages = "5--9",
  month = mar,
  year = "2013",
  ISSN = "1234-5678",
}
```

viene convertito internamente nella `table`

```
{
  ["author"]="An Author and Another One",
  ["category"]="article",
  ["index"]=1,
  ["issn"]="1234-5678",
  ["journal"]="maps",
  ["month"]="3",
  ["number"]="2",
  ["pages"]="5--9",
  ["tag"]="sometag",
  ["title"]="A hopefully
\\btxcmd{sltt}{meaningful} title",
  ["volume"]="25",
  ["year"]="2013",
}
```

Sorge spontaneo immaginare un analogo formato XML ed infatti la macro

`\convertbtxdatasettoxml[<dataset>]` converte un file `bib` (o più correttamente un *dataset*, cfr. oltre) in modo diretto:

```
<?xml version='1.0' standalone='yes'?>
<bibtex>
  <entry tag="sometag" category="article"
        index="1">
    <field name="author">An Author
        and Another One</field>
    <field name="issn">1234-5678</field>
    <field name="journal">maps</field>
    <field name="month">3</field>
    <field name="number">2</field>
    <field name="pages">5--9</field>
    <field name="title">A hopefully
\\btxcmd{sltt}{meaningful} title</field>
```

```
<field name="volume">25</field>
<field name="year">2013</field>
</entry>
</bibtex>
```

Il formato XML può anche essere letto in alternativa al file bib. Infine, è possibile utilizzare file in stile TEX:

```
\startpublication
  \category{article}
  \tag{sometag}
  \author{An Author and Another One}
  \category{article}
  \index{1}
  \issn{1234-5678}
  \journal{maps}
  \month{3}
  \number{2}
  \pages{5--9}
  \tag{sometag}
  \title{A hopefully
\btxcmd{sltt}{meaningful} title}
  \volume{25}
  \year{2013}
\stoppublication
```

È quindi possibile avere diversi file che contengono riferimenti bibliografici: i file costituiscono un *dataset* associato ad un identificativo (standard nell'esempio) nel modo seguente:

```
\definebtxdataset[standard]
\usebtxdataset[standard][bibl.bib]
\usebtxdataset[standard][bibl.lua]
\usebtxdataset[standard][bibl.xml]
\usebtxdataset[standard][bibl.tex]
```

Un dataset è un elemento della Lua table `publications.datasets` ed è a sua volta una table, ad esempio i campi `luadata` e `xmldata` contengono rispettivamente il formato LUA ed XML⁴ del dataset:

```
\startluacode
local datasets= publications.datasets
for _,dataset in pairs(datasets) do
  table.print(dataset.luadata)
  print(dataset.xmldata)
end
\stopluacode
```

È buona norma specificare sempre l'identificativo: allo stato attuale esiste un solo dataset, `standard`, ma è previsto il supporto per più dataset.

2.2 Presentazione di un dataset

Come visto precedentemente, la struttura dati di base di `publications` è una table LUA e quindi necessita di una “presentazione” o *rendering* — in breve, lo stile della bibliografia. La macro `\definebtxrendering` associa un dataset ad un rendering:

4. La conversione deve essere attivata dall'utente con `\convertbtxdatasettoxml[standard]`.

```
\definebtxrendering
[standard:apa]
[dataset=standard,
alternative=apa]
```

In questo esempio il dataset `standard` è associato all'alternativa `apa`⁵ (i.e. lo stile) ed è definito nel file `publ-imp-apa.mkiv`. Il “nome” (o chiave) di questo rendering è `standard:apa` ed è definito dall'utente. Per il momento `apa` è l'unica alternativa ad essere implementata, ma è sufficientemente ricca da poter essere usata come base di partenza per altri stili: non ci addentreremo quindi in tutti i dettagli, limitandoci ai punti salienti. Ogni campo (*field*) di una voce (*entry*) della bibliografia è gestito tramite un *setup* definito tramite la macro `\startsetups btx:apa<field> ... \stopsetups` e richiamato con `\btxsetup{<field>}` come nel caso seguente, dove `<field>` è `article`:

```
\startsetups btx:apa:article
  \btxsetup{btx:apa:common:author-or-key-and-
year}
  \btxdoif {title} {
    \btxflush{title}\btxperiod
  }
  \btxdoifelse {journal} {
    \bgroup\it\btxflush{journal}\egroup
  } {
    \btxdoif {crossref} {
      In\btxspace\btxflush{crossref}
    }
  }
  \btxdoifelse {volume} {
    \btxcomma
  }
  \bgroup\it\btxflush{volume}\egroup
  \btxdoif {issue} {
    \btxlparent\btxflush{issue}\btxrparent
  }
  \btxdoif {pages} {
    \btxcomma\btxflush{pages}
  }
  \btxperiod
  } {
    \btxsetup{btx:apa:common:pages:pp}
  }
  \btxsetup{btx:apa:common:note}
  \btxsetup{btx:apa:common:comment}
\stopsetups
```

Il file è quindi caricato con il comando `\loadbtxdefinitionfile[apa]` dove, come si nota, il nome dello stile `apa` funge da chiave. All'interno del documento il riferimento alla voce della bibliografia si ottiene con la macro `\cite` (i.e. `\cite{sometag}`) e l'elenco delle pubblicazioni tramite `\placelistofpublications[standard:apa]` (letteralmente: “mettere la lista delle pubblicazioni `standard` con stile `apa`”):

5. American Psychological Association cfr. ANONYMOUS (2009)

```

\definebtxdataset[standard]
\usebtxdataset[standard][bibl.bib]
\usebtxdataset[standard][bibl.lua]
\usebtxdataset[standard][bibl.xml]
\usebtxdataset[standard][bibl.tex]
\starttext
A citation from the dataset\cite[sometag]
\placelistofpublications[standard:apa]
\stoptext

```

Le proprietà di un rendering si possono modificare con `\setuprendering` (i valori di default sono tra parentesi):

```

\setupbtxrendering [<name>][.=.]
alternative = <TEXT> ("apa")
dataset     = <TEXT> ("standard")
setups      = <TEXT> ("btx:rendering:apa")
method      = "local"|"global"|"none"|"force"
              ("global")
sorttype    = "short"|"reference"|"dataset"|"
              "default"
criterium   = "here"|"cite"|"all" ("here")
refcommand  = <TEXT> ("authoryears")
numbering   = "yes"|"cite" ("yes")
width       = <DIMENSION>|"auto" ("auto")
distance    = <DIMENSION> ("1.5\emwidth")

```

Menzioniamo solo due chiavi, `method` e `criterium`. La chiave `method` seleziona come mostrare la lista: `local` è usato per le liste locali ad una sezione (e.g. alla fine di ogni capitolo); `global` quando la lista viene mostrata una sola volta nel documento (e.g. alla fine di un libro); `none` non mostra la lista (ma continua a gestirla); `force` mostra tutti gli elementi della lista, anche eventuali duplicati (utile per un controllo). La chiave `criterium` mostra tutti gli elementi della bibliografia anche se non citati (`all`), oppure solo quelli citati (`here` o `cite`).

È possibile definire il dataset di default con `\setupbtxrendering[dataset=<TEXT>]`: in questo modo il nome è sottinteso e non è necessario specificarlo ulteriormente (ad esempio con `\setupbtxrendering[dataset=standard]` e `\setupbtxrendering[method=local]` si modifica `method` del dataset `standard`).

È previsto anche un meccanismo per gestire le *varianti* di lista, ovvero come mostrare i nomi, cognomi, mesi e l'ordinamento:

```

\setupbtxlistvariant [<name>][.=.]
namesep      = <TEXT> (" ")
lastnamesep  = <TEXT> ("and")
finalnamesep = <TEXT> ("and")
firstnamesep = <TEXT>
juniorsep    = <TEXT>
vonsep       = <TEXT>
surnamesep   = <TEXT>
surnamejuniorsep = <TEXT> (" ")
juniorjuniorsep = <TEXT>
surnamefirstnamesep = <TEXT> (" ")
surnameinitialsep = <TEXT> (" ")
etallimit    = <TEXT> ("5")

```

```

etaldisplay  = <TEXT> ("5")
etaltext     = <TEXT> ("et al."=
monthconversion = "number" "month"
              "month:mnem"
              ("number")
authorconversion = "normal" "inverted"
              "normalshort"
              "invertedshort"
              ("normal")

```

Attualmente le varianti sono `"author"`, `"editor"`, `"artauthor"` e *non* sono legati alle alternative, ma condivise.

Infine `\setupbtxlist` imposta i parametri generali (colore, stile, allineamento...) della lista:

```

\setupbtxlist [<name>][.=.]
alternative = <TEXT> ("left")
style       = <TEXT>
color       = <TEXT>
headstyle   = <TEXT>
headcolor   = <TEXT>
width       = <DIMENSION> ("4\emwidth")
distance    = <DIMENSION> ("1\emwidth")
hang        = <NUMBER>
align       =
headalign   =
margin      = "cd:yes" "cd:no" ("cd:no")
before      = <COMMAND> ("blank")
inbetween   = <COMMAND> ("blank")
after       = <COMMAND> ("blank")
display     = "cd:yes" "cd:no" ("yes")
command     = <COMMAND>

```

I campi attualmente riconosciuti sono:

abstract, address, annotate, assignee, author, bibnumber, booktitle, chapter, comment, country, day, dayfiled, doi, edition, editor, eprint, howpublished, institution, isbn, issn, journal, key, keyword, keywords, language, lastchecked, month, monthfiled, names, nationality, note, notes, number, organization, pages, publisher, revision, school, series, size, title, type, url, volume, year, yearfiled.

2.3 Citazioni

L'ultima componente del modulo è costituita dalle citazioni, indipendenti dalle alternative e varianti di lista. Il comando è `\cite[<nome>][<tag>]`, dove `<nome>` è opzionale (default: `num`) e fa riferimento anche in questo caso ad una variante, mentre `<tag>` è obbligatorio e fa riferimento ad una voce del dataset con il corrispondente `tag`. Una variante di citazione è prima definita con `\definebtxcitevariant`:

```

\definebtxcitevariant [<name1>][<name2>]

```

dove `<name2>` è opzionale, ma se presente implica che `name1` è una copia di `name2`; i suoi parametri si possono poi modificare con `\setupbtxcitevariant`:

```
\setupbtxcitevariant [<name>][.=.]
alternative = <TEXT> ("num")
setups      = <TEXT> ("btx:cite:num")
interaction = <TEXT> ("start")
andtext     = <TEXT> ("and")
otherstext  = <TEXT> ("et al.")
compress    = <TEXT> ("no")
putsep      = TEXT
lastputsep  = TEXT
inbetween   = TEXT
right       = TEXT
middle      = TEXT
left        = TEXT
```

Le alternative attualmente sono:

"author", "authornum", "authoryear",
"authoryears", "doi", "key", "num", "page",
"page", "serial", "short", "type", "url",
"year".

Anche in questo caso è possibile definire la variante di default:

```
\setupbtxcitevariant[alternative=<TEXT>]
```

Inoltre, dato che in un documento è possibile definire diversi dataset, `\cite` permette di specificare la coppia `<dataset>::<tag>`, i.e. `\cite[standard::sometag]` per individuare la voce della bibliografia nel dataset `<dataset>` (cfr. figura 1).

3 Il formato RIS in publications

RIS è il formato bibliografico usato dai programmi Reference Manager® e EndNote® della Thomson Reuters. È analogo al formato bib (ma non gestisce elementi tipografici) ed è descritto in http://www.refman.com/support/risformat_fields_01.asp e link seguenti; un esempio è riportato nella figura 2. Un file RIS è una sequenza di record (i.e. una voce della bibliografia) suddiviso in campi (*fields*); ogni campo inizia con una etichetta (*tag*) ed ha un contenuto. Il formato utilizza la IBM Extended Character Set (più precisamente un suo sottoinsieme), quindi *non* è UNICODE®. Un *tag* occupa 6 byte ed è del tipo `<tagname>_ _ _ _ _ <content>`, dove i *tagname* sono descritti nelle specifiche e `_` è il carattere ASCII `0x2D` ("segno meno"). Il contenuto dipende dal *tag*: in alcuni casi è libero, in altri è limitato in lunghezza, in altri ancora assume valori da un insieme predefinito. Ogni record *deve* iniziare con `TY_ _ _ _ _ <reference type>` (dove *<reference type>* sono 4 byte che identificano il tipo di voce come abstract, report etc.) e *deve* terminare con `ER_ _ _ _ _`; tra questi due vi *possono essere* i rimanenti campi il cui ordine non è rilevante.

Per il parsing di un file RIS è naturale usare *LPeg*, il modulo Lua per le *Parsing Expression Grammar*. Nella prossima sezione daremo una panoramica del modulo, con l'avvertenza che si tratta di materiale la cui comprensione può non risultare immediata ad una prima lettura e comunque limi-

tato alle funzioni usate per il formato RIS. Le Peg sono diffusamente usate in ConTEXt e il modulo *publications* non fa eccezione.

3.1 LPeg: Patterns

Senza entrare in dettagli tecnici (per i quali rimandiamo a GRUNE e JACOBS (2007), sez. 15.7 per un inquadramento generale ed a IERUSALIM-SCHY (2009) e <http://www.inf.puc-rio.br/~roberto/lpeg> per l'implementazione in LUA), una Peg è un insieme di *pattern* combinati tramite le operazioni `+`, `*`, `-` e `^`. Un pattern *terminale* è la stringa vuota `ε` oppure un intero d con $0 \leq d \leq 255$ espresso tramite una stringa, in modo tale che ad esempio `" "` è la stringa vuota `ε` (come anche `' '` e `[=[]=]`, visto che entrambe sono stringhe valide in LUA) mentre `"a"` corrisponde — se si usa una codifica compatibile con ASCII — all'intero 97 (in alternativa `"\97"` o `"\097"` sono entrambi stringhe LUA equivalenti a 97). A parte `ε`, si tratta comunque di valori rappresentati semplicemente con un singolo byte, per cui LPeg offre le seguenti funzioni per specificare pattern più complessi:

- `lpeg.P(<str>)`: è il pattern della stringa *str*, ovvero la sequenza (da sinistra verso destra) degli interi d che, nella codifica in uso, corrisponde alla stringa *str*. Così `lpeg.P("")` è `ε`, e, se si usa la codifica UTF-8, `lpeg.P("ac")` è la sequenza 97, 99, `lpeg.P("ð")` è la sequenza 195, 179. *Nota bene*: se si usa una codifica diversa da UTF-8, i valori della sequenze cambiano di conseguenza. Ad esempio, con IBM Extended Character Set, `lpeg.P("ð")` è la sequenza 149 — del tutto diversa dal caso precedente;
- `lpeg.P(<n>)`: se $n = 0$ è la stringa vuota, se $n > 0$ corrisponde ad un pattern di esattamente n terminali (i.e. una stringa di n byte) e se $n < 0$ è un pattern di *al massimo* n byte. I valori dei byte non sono importanti;
- `lpeg.P(<pattern>)`: se *P1* è un pattern `lpeg.P(P1)` è *P1* (i.e. in questo caso `lpeg.P` è la funzione identità);
- `lpeg.P(<boolean>)`: se *boolean* è `true` il pattern riconosce qualsiasi input (pattern true T), se è `false` il pattern rigetta qualsiasi input (pattern false F);
- `lpeg.R(<range>)`: se *range* è una stringa di due byte $d_1 d_2$ e $d_1 \leq d_2$, il pattern riconosce un carattere $c \in \{d_1, d_1 + 1 \dots d_2\}$; se $d_1 > d_2$ è il pattern false F. La funzione accetta anche un array di *range* ed in questo caso il pattern è l'unione dei pattern dei singoli elementi; `lpeg.R({"az", "AZ"})` è il pattern delle lettere minuscola o maiuscola (in codifica compatibile ASCII). Chiamata senza argomento `lpeg.R()` ritorna il pattern F;

- `lpeg.S(<str>)`: è il pattern unione dei terminali in *str*. Ad esempio `lpeg.S("+*/*")` è il pattern che riconosce un carattere $c \in \{ "+", "-", "*", "/" \}$; `lpeg.S("")` è il pattern `F`.

3.2 LPeg: Operazioni & Captures

Come accennato precedentemente, sui pattern sono definite quattro operazioni, $+$, $*$, $-$, $\wedge$, con il seguente significato:

- $P3=P1+P2$: il pattern $P3$ tenta di riconoscere $P1$, ed in caso di successo procede oltre tralasciando $P2$. In caso di fallimento, torna al punto di partenza (*backtracking*) e tenta $P2$. È una scelta *ordinata* e quindi *non commutativa*: $P("")+P("a")$, $P("a")+P("")$ e $P("a")$ sono 3 pattern diversi, — questo fatto renda la somma di due pattern piuttosto inusuale. Solo se i $P1$ e $P2$ sono insiemati vale $P1+P2 = P2+P1$; se $P1$ un qualsiasi pattern allora $P1+F = F+P1 = P1$ (i.e. F è l'identità dell'operazione $+$);
- $P3=P1*P2$: il pattern $P3$ tenta di riconoscere $P1$, ed in caso di successo procede oltre tentando $P2$. È commutativa, con identità T : $P1*T = T*P1 = P1$;
- $P3=-P1$: il pattern $P3$ riconosce tutto tranne $P1$. Ad esempio `-lpeg.P(1)` riconosce qualsiasi pattern che non sia un carattere — i.e. la stringa vuota;
- $P3=P1-P2$: il pattern $P3$ riconosce $P1$ *se non riconosce* $P2$. Detto in altro modo, $P1-P2=(-P2)*P1$;
- $P3=P1\wedge n$: riconosce *almeno* n copie del pattern $P1$, inclusa ϵ se $n = 0$. Se $n < 0$ riconosce *al massimo* $|n|$ copie di $P1$.

La funzione `lpeg.match(P,s,[j])` verifica se il pattern P è nella stringa s *partendo dall'inizio di s* se j è omesso; se la verifica è positiva ritorna la posizione del primo byte all'interno di s *dopo* il match, altrimenti `nil`. Ad esempio `lpeg.match(lpeg.P('a'),'baaaa')` ritorna `nil`, `lpeg.match(lpeg.P('a'),'aaaab')` ritorna `2`, `lpeg.match(lpeg.P(''),'aaaab')` ritorna `1`. Il terzo parametro opzionale j fa partire il matching dalla posizione j . Marcando in modo opportuno un pattern P , la funzione `match` può anche ritornare la stringa del pattern “catturata” (*capture*) all'interno della stringa s . La marcatura è semplice: la funzione `lpeg.C(P)` ritorna un pattern uguale a P , ma `lpeg.match(lpeg.C(P))` ritorna la stringa invece di una posizione. Quindi:

```
match = lpeg.match
P,C    = lpeg.P, lpeg.C
print("1:>"..match(C(P('a')^0),'aaaab')..'<')
```

```
print("2:>"..match(C(P('a')^0),'baaaa')..'<')
print("3:", match(C(P('a')^1),'baaaa'))
```

produce

```
1:>aaaa<
2:><<
3: nil
```

(la seconda capture è la stringa vuota ϵ). Infine, le capture sono ricorsive — ad esempio `C(P1+C(P2)*P3)` è una capture valida — e `lpeg.Ct(P)` ritorna una *table* con tutte le capture del pattern P .

3.3 Una Peg per il formato RIS

ConTeXt mette a disposizione pattern di uso comune che risultano comodi da utilizzare:

```
local patterns = lpeg.patterns
local uppercase = patterns.uppercase
local digit = patterns.digit
local whitespace = patterns.whitespace
local spacer = patterns.spacer
local newline = patterns.newline
local anything = patterns.anything
```

È inoltre conveniente utilizzare gli alias

```
local P, R, S, V, Ct, C, Cs, Cc, Cp, Cmt =
  lpeg.P, lpeg.R, lpeg.S, lpeg.V, lpeg.Ct,
  lpeg.C, lpeg.Cs, lpeg.Cc, lpeg.Cp, lpeg.Cmt
```

Entriamo ora nel vivo delle specifiche. Ricordiamo che un file è formato da uno o più record, un record ha uno o più field, un field ha un tag ed un contenuto. Un tag ha un tagname di due caratteri che sono elencati nelle specifiche; l'insieme dei tagname è quindi la somma dei singoli tagname:

```
-- Not ER
local tagname = P('A1')+P('A2')+P('A3')+
P('AB')+P('AD')+P('AU')+P('AV')+P('BT')+
P('CT')+P('CY')+P('ED')+P('EP')+P('ID')+
P('IS')+P('J1')+P('J2')+P('JA')+P('JF')+
P('JO')+P('KW')+P('L1')+P('L2')+P('L3')+
P('L4')+P('M1')+P('M2')+P('M3')+P('N1')+
P('N2')+P('PB')+P('PY')+P('RP')+P('SN')+
P('SP')+P('T1')+P('T2')+P('T3')+P('TI')+
P('TY')+P('UY')+P('U1')+P('U2')+P('U3')+
P('U4')+P('U5')+P('UR')+P('VL')+P('Y1')+
P('Y2')
local alltagname = tagname + P('ER')
```

Il pattern di un tag è altrettanto immediato: dato che il tag che chiude un record è costante ($ER_{\cup\cup\cup}$), non serve memorizzarlo — deve però essere riconosciuto:

```
local dash=P("-")
local tag=C(tagname)*spacer*spacer*dash
*spacer+(P('ER')*spacer*spacer*dash
*spacer^0)
```

L'ultimo pattern dovrebbe essere `spacer`, ma tolieremo un numero arbitrario di spazi. È conveniente definire i pattern dei field di apertura `TY` e chiusura `ER`:

```
local valid_typerref_values = P('ABST')
+P('ADVS')+P('ART')+P('BILL')+P('BOOK')
+P('CASE')+P('CHAP')+P('COMP')+P('CONF')
+P('CTLG')+P('DATA')+P('ELEC')+P('GEN')
+P('HEAR')+P('ICOMM')+P('INPR')+P('JFULL')
+P('JOUR')+P('MAP')+P('MGZN')+P('MPCT')
+P('MUSIC')+P('NEWS')+P('PAMP')+P('PAT')
+P('PCOMM')+P('RPRT')+P('SER')+P('SLIDE')
+P('SOUND')+P('STAT')+P('THES')+P('UNBIL')
+P('UNPB')+P('VIDEO')
local typerref_tagname = P("TY - ")
local typerref_field = typerref_tagname
    *C(valid_typerref_values)*newline
```

```
local endref_tagname = P("ER -")*spacer^0
local endref_field = endref_tagname*newline^0
```

Il pattern `C(valid_typerref_values)` cattura il tipo di voce della bibliografia (book, article, report...) e quindi deve essere salvato.

Per i rimanenti field seguiamo la seguente strategia: definiamo prima di tutto i pattern per i contenuti validi

```
local valid_id_characters = R("09")+R("AZ")
local valid_characters = R("\32\255")
local valid_restricted_characters=
    valid_characters-P("\42")
```

Poi definiamo i field tipo `author`, `date`, `keyword`, `id`, `periodical` e `reprint` perché hanno un contenuto diverso dagli altri field (per brevità riportiamo solo il field `id`; per il programma completo cfr. figure 3-6):

```
local id_tagname =
    C(P("ID"))*P(" - ")
local id_content =
    C((valid_id_characters-(newline*tag))
    *(valid_id_characters-(newline*tag))~19)
local id_field =
    Ct(id_tagname*id_content)*newline
```

Il field `id` ha come unico tagname `ID` ed un contenuto fatto di caratteri alfanumerici — numeri e lettere solo maiuscole — lungo minimo un carattere e massimo venti (pattern `valid_id_characters`); come tutti i field termina con un pattern `newline`. Una volta definiti i field particolari, vediamo il pattern che cattura i field rimanenti:

```
local other_tagname = tag
-(typerref_tagname + endref_tagname
+ author_tagname + date_tagname
+ keyword_tagname + id_tagname
+ periodical_tagname + reprint_tagname)
local other_content = C((anything
- (newline*tag))~0)
local other_field = Ct(other_tagname
    * other_content)* newline
```

Il tagname di un field rimanente è semplice: *non* è né un tagname particolare né di inizio o fine record. Il contenuto del field è costituito da tutti i caratteri fino al successivo tag, `newline` esclusa: nessuna limitazione sul valore dei caratteri e sulla lunghezza.

A questo punto è immediato definire il pattern di un field generale compreso quello iniziale e quello finale:

```
local ris_field = author_field + date_field
+ keyword_field + id_field
+ periodical_field
+ reprint_field + other_field
```

Un record è un field iniziale seguito da almeno un field generale e terminante con il field finale:

```
local ris_record = Ct(typerref_field
    * Ct(ris_field^1)
    * endref_field)
```

Infine, un file contiene almeno un record e noi ignoriamo eventuali spazi iniziali:

```
local ris_file = (whitespace)^0
    *Ct((ris_record*newline^0)^1)
```

La parte finale è semplice: con `lpeg.match(ris_lpeg,ris_data)` si ottiene un iteratore su tutti i record, ed ogni record viene trasformato in una `table` con `transcode_table`:

```
-- Translate a RIS file into
-- a publications Lua file
local function ris_to_lua_table(ris_lpeg,
    ris_data,prefix)

    local lua_table= {}
    local prefix = prefix or
        publications.format.ris.prefix
    for i,record in
        pairs(lpeg.match(ris_lpeg,ris_data)) do
        local TY = record[1]
        local rec_table =
            transcode_table(record[2])
        local idx = tostring(prefix)..tostring(i)
        lua_table[idx]={}
        lua_table[idx].abstract =
            merge_fields(rec_table," ","N2")
        lua_table[idx].address =
            merge_fields(rec_table," ","AD")
        lua_table[idx].author =
            merge_fields(rec_table,"","A1","AU",
                "A2","ED","A3")
        lua_table[idx].category =
            categories[TY]
        lua_table[idx].issn =
            merge_fields(rec_table," ","SN")
        lua_table[idx].keywords =
            merge_fields(rec_table," ","KW")
        lua_table[idx].note =
            merge_fields(rec_table,"; ","
                "N1","AB","AV",
                "M1","M2","M3",
```

```

        "U1","U2","U3","U4","U5")
lua_table[idx].publisher=
    merge_fields(rec_table," ", "PB")
lua_table[idx].title =
    merge_fields(rec_table," -- ",
        "T1","TI","CT",
        "BT","T2","BT","T3")
lua_table[idx].url =
    merge_fields(rec_table," ", "UR")
lua_table[idx].year =
    merge_fields(rec_table,"; ",
        "Y1","P1","Y2")
end
return lua_table
end

```

È importante notare che il riconoscimento è fatto sull'intero file precedentemente caricato in memoria e questo può essere problematico in dispositivi con memoria limitata.

La funzione `ris_to_lua_table` è privata: il modulo `publications` ha una interfaccia pubblica data dalla funzione

`publications.<formato>_to_btx(data)`
dove `<formato>` è il suffisso del file, nel nostro caso `ris`:

```

function publications.ris_to_btx(ris_data)
    return ris_to_lua_table(ris_file,
        ris_data,
        publications.format.ris.prefix)
end

```

Infine, `loaders.<formato>(dataset,filename)` è la funzione pubblica per caricare il file nel dataset:

```

function loaders.ris(dataset,filename)
    loaders.lua(dataset,
        publications.
        ris_to_btx(io.loaddata(filename)
            or ""))
end

```

In un file `TeX` la macro `\usebtxdatase` carica un file `RIS` in modo del tutto analogo ad un file `bib` o `xml` visto in precedenza, utilizzando però `loader.ris` (la chiave è quindi la desinenza `ris`): i.e. `\usebtxdatase[standard][test.ris]` aggiunge il file `test.ris` al dataset `standard`.

4 Conclusioni

Il modulo `publications` di `ConTeXt` ambisce ad emulare `biblatex` e `biber` senza dipendere da programmi esterni, ma sfruttando le librerie già presenti in

`LuaTeX`. In particolare riteniamo che `LUA` e le `Parsing Expression Grammar` siano un valido sostituto del `PERL` e questo, a parità di motore di impaginazione, porta ad un dimezzamento della memoria occupata. La possibilità di gestire formati diversi, ed in particolare un formato `XML`, aggiunge un'elasticità probabilmente non ottenibile attualmente con `biblatex`.

Il modulo attualmente è ancora in fase di sviluppo e non è stata esplorata la possibilità di produrre diagrammi delle relazioni delle voci di bibliografia, funzionalità offerta da `biber` mediante `METAPOST` (il numero di voci del formato `bibtex` è inferiore a quello di `biblatex` e probabilmente anche l'unico stile `APA` deve essere perfezionato. La documentazione è ancora incompleta.

Il team si propone di completare la prima versione per la fine del 2014.

Riferimenti bibliografici

ANONYMOUS (2009). *Publication Manual of the American Psychological Association*. American Psychological Association, Washington, DC, USA, 3ª edizione.

FADINI, M. (2012). «Stemma codicum con `LATEX`». *ArsTeXnica*, (14).

GRUNE, D. e JACOBS, C. (2007). *Parsing Techniques: A Practical Guide*. Monographs in Computer Science. Springer. URL http://books.google.it/books?id=05xA_d5dSwAC.

IERUSALIMSKY, R. (2009). «A text pattern-matching tool based on parsing expression grammars». *Softw. Pract. Exper.*, **39** (3), pp. 221–258. URL <http://dx.doi.org/10.1002/spe.v39:3>.

JERÓNIMO LEAL AND GIANLUCA PIGNALBERI (2012). «Composizione di uno Stemma codicum con `TikZ`». *ArsTeXnica*, (14).

LEAL, J. e PIGNALBERI, G. (2012). *Edizioni Critiche*. Compomat, Rome.

▷ Luigi Scarso
luigi dot scarso at gmail dot com

1

Citations from the dataset: num:[1],[2],[3]
author:(An Author and Another One)
author:(An Author and Another One, An Author and Another One)
authoryear:(An Author and Another One (2013), An Author and Another One (2013))

- 1 Author, A. and One, A. (2013). A hopefully slttmeaningful title. *maps*, 25, 5–9.
- 2 Author, A. and One, A. (2013). LUA1 A hopefully slttmeaningful title. *maps*, 25, 5–9.
- 3 Author, A. and One, A. (2013). XML1 A hopefully slttmeaningful title. *maps*, 25, 5–9.

```
\definebtxdataset[standard]
\usebtxdataset[standard][test.bib]
\usebtxdataset[standard][test.lua]
\usebtxdataset[standard][test.xml]
\definebtxrendering
[standard.apa]
[dataset=standard,method=local,
alternative=apa]

\starttext
Citations from the dataset:
num:\cite[sometag],\cite[luasometag],\cite[xmlsometag]\\
author:\cite[author][sometag]\\
author:\cite[author][xmlsometag,luasometag]\\
authoryear:\cite[authoryear][standard:xmlsometag,sometag]\\
\placelistofpublications[standard.apa][criterium=cite]
\stoptext
```

FIGURA 1: Alcune combinazioni di `\cite` ed il relativo codice.

```

TY__JOUR
A1__Baldwin,S.A.
A1__Fugaccia,I.
A1__Brown,D.R.
A1__Brown,L.V.
A1__Scheff,S.W.
T1__Blood-brain barrier breach following cortical contusion in the rat
J0__J.Neurosurg.
Y1__1996
VL__85
SP__476
EP__481
RP__Not In File
KW__cortical contusion
KW__blood-brain barrier
KW__horseradish peroxidase
KW__head trauma
KW__hippocampus
KW__rat
N2__Adult Fisher 344 rats were subjected to a unilateral impact to the dorsal
cortex above the hippocampus at 3.5 m/sec with a 2 mm cortical depression. This
caused severe cortical damage and neuronal loss in hippocampus subfields CA1, CA3
and hilus. Breakdown of the blood-brain barrier (BBB) was assessed by injecting the
protein horseradish peroxidase (HRP) 5 minutes prior to or at various times following
injury (5 minutes, 1, 2, 6, 12 hours, 1, 2, 5, and 10 days). Animals were killed 1
hour after HRP injection and brain sections were reacted with diaminobenzidine to
visualize extravascular accumulation of the protein. Maximum staining occurred in
animals injected with HRP 5 minutes prior to or 5 minutes after cortical contusion.
Staining at these time points was observed in the ipsilateral hippocampus. Some
modest staining occurred in the dorsal contralateral cortex near the superior
sagittal sinus. Cortical HRP stain gradually decreased at increasing time intervals
postinjury. By 10 days, no HRP stain was observed in any area of the brain. In the
ipsilateral hippocampus, HRP stain was absent by 3 hours postinjury and remained so
at the 6- and 12- hour time points. Surprisingly, HRP stain was again observed in
the ipsilateral hippocampus 1 and 2 days following cortical contusion, indicating
a biphasic opening of the BBB following head trauma and a possible second wave
of secondary brain damage days after the contusion injury. These data indicate
regions not initially destroyed by cortical impact, but evidencing BBB breach, may be
accessible to neurotrophic factors administered intravenously both immediately and
days after brain trauma.
ER__

```

FIGURA 2: Un esempio del formato RIS. In evidenza gli spazi che fanno parte del tag.

```

1
2 local patterns = lpeg.patterns
3 local uppercase = patterns.uppercase
4 local digit = patterns.digit
5 local whitespace = patterns.whitespace
6 local spacer = patterns.spacer
7 local newline = patterns.newline
8 local anything = patterns.anything
9 local P, R, S, V, Ct, C, Cs, Cc, Cp, Cmt = lpeg.P, lpeg.R,
10                                           lpeg.S, lpeg.V, lpeg.Ct,
11                                           lpeg.C, lpeg.Cs, lpeg.Cc,
12                                           lpeg.Cp, lpeg.Cmt
13 publications = publications or {}
14 loaders      = publications.loaders or {}
15 publications.format = publications.format or {}
16 publications.format.ris = publications.format.ris or {}
17 local dash = P("-")
18 -- Not ER
19 local tagname = P('A1')+P('A2')+P('A3')
20               +P('AB')+P('AD')+P('AU')+P('AV')+P('BT')
21               +P('CT')+P('CY')+P('ED')+P('EP')+P('ID')+P('IS')+P('J1')+P('J2')
22               +P('JA')+P('JF')+P('JO')+P('KW')+P('L1')+P('L2')+P('L3')+P('L4')
23               +P('M1')+P('M2')+P('M3')+P('N1')+P('N2')+P('PB')+P('PY')+P('RP')
24               +P('SN')+P('SP')+P('T1')+P('T2')+P('T3')+P('TI')+P('TY')+P('TY')
25               +P('U1')+P('U2')+P('U3')+P('U4')+P('U5')+P('UR')+P('VL')+P('Y1')
26               +P('Y2')
27 local alltagname = tagname + P('ER')
28 local valid_id_characters = R("09")+R("AZ")
29 local valid_characters = R("\32\255")
30 local valid_restricted_characters = valid_characters-P("\42")
31 local tag = C(tagname) * spacer * spacer * dash * spacer
32           + (P('ER') * spacer * spacer * dash * spacer^0)
33 local valid_typeref_values = P('ABST')+P('ADVS')
34               +P('ART')+P('BILL')+P('BOOK')+P('CASE')+P('CHAP')
35               +P('COMP')+P('CONF')+P('CTLG')+P('DATA')+P('ELEC')+P('GEN')+P('HEAR')
36               +P('ICOMM')+P('INPR')+P('JFULL')+P('JOUR')+P('MAP')+P('MGZN')
37               +P('MPCT')+P('MUSIC')+P('NEWS')+P('PAMP')+P('PAT')+P('PCOMM')+P('RPRT')
38               +P('SER')+P('SLIDE')+P('SOUND')+P('STAT')+P('THES')+P('UNBIL')
39               +P('UNPB')+P('VIDEO')
40 local typeref_tagname = P("TY_ _- _")
41 local typeref_field = typeref_tagname * C(valid_typeref_values) * newline
42 local endref_tagname = P("ER_ _- _") * spacer^0
43 local endref_field = endref_tagname * newline^0
44 local tagname_field = tag - (typeref_tagname + endref_tagname)
45 local author_tagname = C(P("A1")+P("U1")+P("A2")+P("ED")+P("A3")) * P("_ _- _")
46 local author_content = C((valid_restricted_characters - (newline*tag))^~255)
47 local author_field = Ct(author_tagname * author_content) * newline
48 local yyyy_mm_dd = (digit*digit*digit*digit)^~1
49                   *P("/")^~1*(digit*digit)^~1*P("/")^~1*(digit*digit)^~1
50 local date_tagname = C(P("Y1")+P("PY")+P("Y2")) * P("_ _- _")
51 local date_content = C(yyyy_mm_dd
52                       *(valid_restricted_characters-(newline*tag))^~255)
53 local date_field = Ct(date_tagname * date_content) * newline
54 local keyword_tagname = C(P("KW")) * P("_ _- _")
55 local keyword_content = C((valid_restricted_characters - (newline*tag))
56                           ^~255)
57 local keyword_field = Ct(keyword_tagname * keyword_content) * newline
58 local id_tagname = C(P("ID")) * P("_ _- _")
59 local id_content = C((valid_id_characters - (newline*tag))
60                     *(valid_id_characters - (newline*tag))^~19 )
61 local id_field = Ct(id_tagname * id_content) * newline

```

FIGURA 3: Una Peg per il formato RIS (1/4).

```

1 local periodical_tagname = C(P("JF")+P("J0")
2                               +P("JA")+P("J1")+P("J2"))*P("_-")
3 local periodical_content = C((valid_restricted_characters
4                               - (newline*tag))^~255)
5 local periodical_field    = Ct(periodical_tagname * periodical_content)*
6                               newline
7 local yy_mm_dd            = P("(")*digit*digit*P("/")*digit*digit
8                               *P("/")*digit*digit*P(")")
9 local in_file             = S("Ii")*S("Nn")*spacer^0*S("Ff")
10                              *S("Ii")*S("Ll")*S("Ee")
11 local not_in_file        = S("Nn")*S("Oo")*S("Tt")*spacer^0*S("Ii")*S("Nn")
12                              *spacer^0*S("Ff")*S("Ii")*S("Ll")*S("Ee")
13 local on_request         = S("Oo")*S("Nn")*spacer^0*S("Rr")
14                              *S("Ee")*S("Qq")*S("Uu")*S("Ee")*S("Ss")*S("Tt")
15 local reprint_tagname    = C(P("RP"))*P("_-")
16 local reprint_content    = C((spacer^0 * in_file          * spacer^0
17                               + (spacer^0 * not_in_file    * spacer^0
18                               + (spacer^0 * on_request      * spacer^0 * yy_mm_dd)
19                               ) -(newline*tag))
20 local reprint_field      = Ct(reprint_tagname * reprint_content) * newline
21 local other_tagname      = tag - (typeref_tagname + endref_tagname
22                               + author_tagname + date_tagname + keyword_tagname
23                               + id_tagname + periodical_tagname + reprint_tagname )
24 local other_content      = C((anything - (newline*tag))^0)
25 local other_field        = Ct(other_tagname * other_content) * newline
26 local ris_field          = author_field + date_field + keyword_field + id_field
27                               + periodical_field + reprint_field + other_field
28 local ris_record         = Ct(typeref_field * Ct(ris_field^1) * endref_field)
29 local ris_file           = (whitespace)^0*Ct((ris_record*newline^0)^1)
30 local categories= [{"ABST"] = "article", -- "Abstract",
31 ["ADVS"] = "misc", -- "Audiovisual material",
32 ["ART"] = "article", -- "Art Work",
33 ["BILL"] = "article", -- "Bill/Resolution",
34 ["BOOK"] = "book", -- "Book, Whole",
35 ["CASE"] = "book", -- "Case",
36 ["CHAP"] = "book", -- "Book chapter",
37 ["COMP"] = "techreport", -- "Computer program",
38 ["CONF"] = "proceedings", -- "Conference proceeding",
39 ["CTLG"] = "book", -- "Catalog",
40 ["DATA"] = "misc", -- "Data file",
41 ["ELEC"] = "article", -- "Electronic Citation",
42 ["GEN"] = "book", -- "Generic",
43 ["HEAR"] = "article", -- "Hearing",
44 ["ICOMM"] = "article", -- "Internet Communication",
45 ["INPR"] = "book", -- "In Press",
46 ["JFULL"] = "article", -- "Journal (full)",
47 ["JOUR"] = "article", -- "Journal",
48 ["MAP"] = "article", -- "Map",
49 ["MGZN"] = "article", -- "Magazine article",
50 ["MPCT"] = "misc", -- "Motion picture",
51 ["MUSIC"] = "article", -- "Music score",
52 ["NEWS"] = "article", -- "Newspaper",
53 ["PAMP"] = "book", -- "Pamphlet",
54 ["PAT"] = "article", -- "Patent",
55 ["PCOMM"] = "article", -- "Personal communication",
56 ["RPRT"] = "article", -- "Report",
57 ["SER"] = "book", -- "Serial (Book, Monograph)",
58 ["SLIDE"] = "article", -- "Slide",
59 ["SOUND"] = "article", -- "Sound recording",
60 ["STAT"] = "article", -- "Statute",
61 ["THES"] = "book", -- "Thesis/Dissertation",
62 ["UNBIL"] = "article", -- "Unenacted bill/resolution",
63 ["UNPB"] = "unpublished", -- "Unpublished work",
64 ["VIDEO"] = "misc", -- "Video recording"}

```

FIGURA 4: Una Peg per il formato RIS (2/4).

```

1  publications.format.ris.prefix = "RIS"
2  -- From { "KY" , "Value"}
3  -- to   ["KY"] = "Value"
4  local function transcode_table(t)
5      local tt = {}
6      for _, v in pairs(t) do
7          tt[v[1]] = v[2]
8      end
9      return tt
10 end
11 -- Merge the fields of t
12 -- excluding nil, with separation sep
13 local function merge_fields(t,sep,...)
14     local f_list={ ... }
15     local tt = {}
16     for _, f in ipairs(f_list) do
17         if t[f] then
18             tt[#tt+1] = t[f]
19         end
20     end
21     return table.concat(tt,sep)
22 end
23
24 -- Translate a RIS file into
25 -- a publications Lua file
26 local function ris_to_lua_table(ris_lpeg,ris_data,prefix)
27     local lua_table= {}
28     local prefix = prefix or publications.format.ris.prefix
29     for i,record in pairs( lpeg.match(ris_lpeg,ris_data) ) do
30         local TY      = record[1]
31         local rec_table = transcode_table(record[2])
32         local idx = tostring(prefix)..tostring(i)
33         lua_table[idx]={}
34         lua_table[idx].abstract = merge_fields(rec_table,"_",
35                                             "N2")
36         lua_table[idx].address  = merge_fields(rec_table,"_",
37                                             "AD")
38         lua_table[idx].author   = merge_fields(rec_table,"",
39                                             "A1","AU","A2",
40                                             "ED","A3")
41         lua_table[idx].category = categories[TY]
42         lua_table[idx].issn      = merge_fields(rec_table,"_",
43                                             "SN")
44         lua_table[idx].keywords = merge_fields(rec_table,"_",
45                                             "KW")
46         lua_table[idx].note      = merge_fields(rec_table,";_",
47                                             "N1","AB","AV",
48                                             "M1","M2","M3",
49                                             "U1","U2","U3","U4","U5")
50         lua_table[idx].publisher= merge_fields(rec_table,"_",
51                                             "PB")
52         lua_table[idx].title     = merge_fields(rec_table,"_--_",
53                                             "T1","TI","CT",
54                                             "BT","T2","BT","T3")
55         --[=[ lua_table[idx].title = string.gsub(lua_table[idx].title,"._*$","")
56             ]==]
56         lua_table[idx].url       = merge_fields(rec_table,"_",
57                                             "UR")
58         lua_table[idx].year      = merge_fields(rec_table,";_",
59                                             "Y1","P1","Y2")
60     end
61     return lua_table
62 end

```

FIGURA 5: Una Peg per il formato RIS (3/4).

```

1  function publications.ris_to_btx(ris_data)
2      return ris_to_lua_table(ris_file,ris_data,publications.format.ris.prefix)
3  end
4
5  function loaders.ris(dataset,filename)
6      loaders.lua(dataset,publications.ris_to_btx(io.loadaddata(filename) or ""))
7  end
8
9  --[==[ Example
10 Run with
11 mtxrun --script publ-format-ris.lua
12 or
13 mtxrun.exe --script publ-format-ris.lua
14 s2="sample02.txt"
15 s3="sample03.txt"
16 s4="sample04.txt"
17 s5="sample05.txt"
18
19 f_s = io.open(s5,'r')
20 ris_s= f_s:read("*a")
21 f_s:close()
22
23 local lua_table = ris_to_lua_table(ris_file,ris_s,
24                                     publications.format.ris.prefix)
25 table.print(lua_table)
26 ]==]

```

FIGURA 6: Una Peg per il formato RIS (4/4).

Eventi e novità

DANTE e. V. compie 25 anni

Si è svolto dall'11 al 14 aprile, presso il *Psychologisches Institut* dell'università di Heidelberg il 50° congresso di DANTE e. V., che celebra quest'anno i venticinque anni di attività. Il programma dell'evento è consultabile (per chi conosce il tedesco) sul sito web dell'associazione: <http://www.dante.de/events/dante2014/Programm.html>.



BachoTeX 2014

Anche quest'anno, come di prammatica, il convegno organizzato dal GUST, che raccoglie gli appassionati di T_EX provenienti dalla Polonia, si svolgerà nella località di Bachotek, dal 30 aprile al 4 maggio. Gli interventi presentati durante l'incontro saranno dedicati al tema: "What can typography gain from electronic media?". È inoltre prevista la presenza di alcuni workshop e tutorial su singoli argomenti di carattere pratico.

Sul sito dell'associazione è possibile reperire informazioni sul programma e sulla logistica (<http://www.gust.org.pl/bachotex/2014>).

BachoT_EX 2014



TUG 2014

È giunto alla 35^a edizione l'annuale meeting del TUG, che quest'anno si svolgerà negli Stati Uniti, a

Portland, Oregon, dal 28 al 30 luglio. Gli interventi potranno spaziare su tutto l'arco delle tematiche collegate all'uso di "T_EX and friends", con una particolare attenzione al versante pratico, come specificato nel Call for papers. Contestualmente alla prima sessione del convegno verrà organizzato un workshop rivolto a utenti principianti o intermedi.

Tutti gli indirizzi utili per presentare un articolo o per registrarsi si trovano sulla pagina dedicata al convegno: <http://www.tug.org/tug2014/>



8th International ConT_EXt Meeting

L'ottavo meeting internazionale di ConT_EXt si terrà dall'8 al 13 settembre a Bassenge, in Belgio. Al meeting saranno presenti sviluppatori di ConT_EXt e LuaT_EX che presenteranno le ultime novità e le idee che daranno una direzione agli sviluppi futuri di questi due programmi. Gli interventi saranno accompagnati da tutorial incentrati sulle diverse tecniche di utilizzo in vari contesti pratici.

Per informazioni consultare la pagina web <http://meeting.contextgarden.net/>



Questa rivista è stata stampata
presso Centro Stampa e Riproduzione S.r.l.,
Via di Pietralata, 157 – 00158 Roma,
su carta vellum white 80 g.
Copertina: vellum white 260 g.



Undicesimo convegno nazionale su $\text{T}_{\text{E}}\text{X}$, $\text{\LaTeX}$ e tipografia digitale

Verona, 18 ottobre 2014

Tema principale:

“ TeX come strumento per la scrittura accademica”

Call for Papers

Sabato 18 ottobre a Verona si terrà l'undicesimo Convegno annuale su $\text{T}_{\text{E}}\text{X}$, $\text{\LaTeX}$ e tipografia digitale organizzato dal Gruppo Utilizzatori Italiani di $\text{T}_{\text{E}}\text{X}$. Il Convegno sarà un momento di ritrovo e di confronto per la comunità $\text{\LaTeX}$ italiana, tramite una serie di interventi atti sia a contribuire all'arricchimento sia a supportarne lo sviluppo.

Maggiori informazioni sul Convegno e sulle modalità di presentazione degli interventi saranno disponibili all'indirizzo:

<http://www.guitex.org/guitmeeting/2014/>

ArsT_EXnica

Rivista italiana di T_EX e L^AT_EX

Numero 17, Aprile 2014

- 3 Editoriale
Claudio Beccari
- 4 I comandi condizionali
Claudio Beccari
- 12 ArsT_EXnica si rinnova
Claudio Beccari, Massimiliano Dominici, Gianluca Pignalberi
- 18 Realizzare semplici animazioni in figure:
come usare TikZ in Beamer
Claudio Fiandrino
- 24 Testatine e piè di pagina laterali
Gianluca Pignalberi
- 29 Un primo sguardo al modulo publications
Luigi Scarso

