

LCT

The *Linux&C. Toolkit*

<http://www.enneenne.com/projects/lct>

by

Rodolfo Giometti <giometti@oltrelinux.com>

This project is under the GNU GPL license, version 2.



2003



L'origine

Alcuni mesi fa il direttore di Linux&C. chiedeva nella maillist di redazione se qualcuno avesse avuto voglia di scrivere degli articoli in cui ci fosse qualcosa da «provare».

Io gli ho proposto il progetto LCT.



Cosa è?

Il progetto LCT ha un duplice scopo:

- spiegare come funzionano alcuni meccanismi interni di Linux (processi, pipe, schedulatore, ecc.);
- mettere in pratica quanto appreso sviluppando una estensione del linguaggio di scripting TCL tramite il linguaggio C.



A cosa si mira

Si cerca di dare ai lettori la possibilità di mettere in pratica quello che viene illustrato teoricamente.

Chiunque avrà quindi la possibilità di provare da solo quanto legge sulla rivista e di collaborare, se vuole, al miglioramento del progetto stesso.



E cosa è TCL?

TCL (pronuncia: *ticol*) è un potente linguaggio di scripting attraverso il quale è possibile implementare anche programmi complessi.

La sua estensione grafica si chiama TK (pronuncia: *tichei*) con la quale si realizzano già numerose interfacce grafiche.

TCL ha una speciale API in C per aggiungere nuovi comandi.



TCL: il linguaggio «a comandi»

TCL non possiede delle strutture come i normali linguaggi di programmazione ma utilizza solo comandi. Anche i costrutti come *while* e *if* sono in realtà comandi.

```
while test body
```

```
if expr then body1 ?else? body2
```

Questo lo rende estremamente flessibile per eventuali estensioni: basta aggiungere un nuovo comando il quale può implementare anche un nuovo costrutto!



Un comando semplice: uname

Di per se TCL non implementa il comando uname, ma una volta aggiuntolo avremo la possibilità di fare quanto segue:

```
"giometti@zaigor:~/Projects/lct$ ./lctsh
```

```
% uname
```

```
Linux
```

```
% uname -all
```

```
Linux zaigor 2.4.18 #6 SMP Mon Aug 25  
18:01:18 CEST 2003 i686
```



Es. di mplementazione di uname

```
int lct_uname(ClientData clientData, Tcl_Interp
*interp, int objc, Tcl_Obj *CONST objv[])
{
    struct utsname data;
    Tcl_Obj *res;
    CONST84 char *opts[] = {
        "-all", "-kernel-name", "-nodename",
        "-kernel-release", "-kernel-version",
        "-machine", NULL
    };
    ...
}
```



Es. di mplementazione di uname

```
...  
p = 1;    /* first non option */  
while (p < objc && *Tcl_GetString(objv[p]) ==  
        '-') {  
    if (Tcl_GetIndexFromObj(interp, objv[p],  
        opts, "option", 0, &i) != TCL_OK)  
        return TCL_ERROR;  
  
    selected |= 1<<i;  
    p++;  
}  
Tcl_FreeResult(interp);  
...
```



Es. di mplementazione di uname

...

```
ret = uname(&data);  
if (ret < 0) {  
    lct_syscall_err(objv[0]);  
    return TCL_ERROR;  
}
```

...



Es. di mplementazione di uname

...

```
res = Tcl_NewStringObj("", 0);
```

```
if (selected & (1<<KERNEL_NAME))
```

```
    Tcl_AppendStringsToObj(res, data.sysname,  
                           " ", NULL);
```

```
if (selected & (1<<NODENAME))
```

```
    Tcl_AppendStringsToObj(res, data.nodename,  
                           " ", NULL);
```

...

```
Tcl_SetObjResult(interp, res);
```

```
return TCL_OK;
```

```
}
```



Non si reinventa l'acqua calda, ma...

Qualcuno avrà da notare che esistono già alcuni progetti che, estendendo il linguaggio TCL, permettono di utilizzare delle chiamate di sistema come `fork()`, `pipe()`, ecc.: Ettl, TclX.

Lo scopo di LCT non è quello di fare loro concorrenza ma di poter essere utilizzato come strumento didattico.



Composizione degli articoli

Ogni articolo:

- affronterà teoricamente un aspetto notevole del sistema Linux;
- mostrerà come sia possibile utilizzarlo per estendere il linguaggio di scripting TCL;
- farà degli esempi di programmazione con il nuovo linguaggio esteso chiamato LCT.



Stato dell'arte

Attualmente sono già pronti due articoli e il terzo è in «lavorazione»:

1. Illustra il progetto e dà una prima idea di come sia possibile estendere TCL.
2. Parla della gestione dei processi (`fork()` e `wait()`).
3. Parlerà (forse) delle pipe (`pipe()`)
4. Parlerà (forse) dello schedulatore.



Dove prendere il codice

È a disposizione di tutti i lettori un CVS e un FTP anonimo grazie ai quali sarà possibile seguire l'evoluzione del progetto.

Se la cosa riscuoterà successo (sperabile) è in preparazione anche l'apertura di un mail list dove poter scambiare idee ed opinioni con gli altri lettori e con l'autore.

<http://www.enneenne.com/projects/lct>



Link

Home page (dall'uscita del primo articolo):

<http://www.enneenne.com/projects/1ct>

Home page di Tcl/Tk:

<http://www.tcl.tk>

Libro:

Practical programming in Tcl and Tk
(Welch)

