Spaghetti Hack

I moduli Perl sono come gli ingredienti di un buon piatto. E' possibile combinarli per cucinare portate di ogni tipo, oppure combinarli per creare ulteriori ingredienti, come il sugo di un buon piatto di spaghetti. E come per la preparazione degli spaghetti, possiamo comprare un sugo già pronto, o prepararlo da noi utilizzando tutti gli ingredienti che lo compongono.

Fare gli spaghetti con un sugo pronto o con uno preparato da noi ha rispettivi pregi e difetti: nel primo caso utilizzando il sugo già pronto ci possiamo dedicare ad altri aspetti del piatto, concentrando i nostri sforzi sull'obiettivo primario che rimane "servire un buon piatto di spaghetti". Inoltre possiamo affidare la cottura degli spaghetti anche ad altri cuochi che conoscendo il sugo ne faranno lo stesso uso che ne faremmo noi.

Quando invece decidiamo di non usare un sugo già pronto, subentrano maggiori difficoltà per le quali però occorre prendere in considerazione i seguenti aspetti:

- Il sugo che useremo se pur comprato (e quindi utilizzato da tantissime persone) sarà buono ?
- Ammettendo che il sugo sia buono, perché non provarne una variante visto che TMTOWTDI ? (ok! in cucina non si dice :)
- I sughi pronti hanno ingredienti all'interno che non possiamo togliere. Magari proprio quegli ingredienti che in genere non riusciamo a digerire e che rendono il sugo **pesante**.

Tutto questo, porta inevitabilmente a dedurre che entrambi i metodi sono validi, ma che si adattano meglio in relazione alla problematica che dobbiamo affrontare. E allora quando l'uno e quando l'altro?

Sicuramente sceglieremo di cucinare il nostro sugo quando:

- vogliamo ottenere un prodotto su misura che meglio si adatti al tipo di pasta;
- vogliamo che sia digeribile e quindi *leggero* anche al costo di una preparazione più lunga;
- vogliamo essere pronti a situazioni in cui il sugo pronto non è disponibile

Quest'ultima considerazione ci deve far riflettere su un aspetto. Se mai impariamo a cucinare il sugo, il giorno che troveremo il supermercato chiuso, potremmo trovarci veramente in mezzo ai guai!

C'è poi una terza possibilità, e cioè quella di utilizzare un sugo pronto, ma di aggiungere qualche spezia per insaporirlo a nostro gusto.

In IGSuite sono state adottate tutte le soluzioni sopra esposte, abbiamo utilizzato moduli di CPAN lasciandoli inalterati, abbiamo preso moduli da CPAN e li abbiamo leggermente modificati affinchè fossero adatti ai nostri scopi ma soprattutto più leggeri, ed infine abbiamo scritto alcuni moduli from scratch, prestando particolare importanza per ognuna delle soluzioni adottate ai seguenti obiettivi:

- Prestazioni
- Portabilità
- Facilità di installazione

Purtroppo non tutti i moduli Perl in CPAN offrono tali caratteristiche. Ci sono moduli pesantissimi che inglobano centinaia di metodi e per i quali la maggior parte delle volte ci si trova ad usare il 5% delle features. Ci sono moduli che non funzionano su tutte le piattaforme supportate da Perl, ed infine ci sono moduli scritti in C che per essere installati, necessitano di una compilazione che inevitabilmente aumenta le difficoltà di installazione da parte dell'utente.

Come fare quindi per garantire Prestazioni, Portabilità e facilità di installazione pur utilizzando i moduli.

Semplicemente facendo delle scelte caso per caso. Adottando l'una o l'altra soluzione a seconda della problematica da affrontare.

E' chiaro che nel tempo tali scelte possono variare. Potrebbero uscire moduli che risolvono meglio una problematica, ad esempio meglio di come la si era risolta in precedenza con una delle tre soluzioni. Per questo motivo, si è scelto in molti casi di wrappare metodi forniti da moduli esterni, attraverso metodi forniti da IG.pm il Core module del Framework di IGSuite.

A titolo d'esempio le chiamate a CGI.pm per leggere i parametri dai form sono Wrappate dal metodo IG::MkCgiEnv, oppure le chiamate ai moduli per interfacciarsi agli RDBM sono wrappate dal metodo IG::DbQuery. In questo modo se nel tempo viene sviluppato un nuovo modulo più veloce ad esempio di CGI.pm basterà modificare IG.pm lasciando invariate tutte le applicazioni che ne fanno uso.

Ovviamente quella esposta non è una regola generale ed è stata applicata solo per i moduli non Core del Perl ma soprattutto per moduli a carattere ed uso generale. Sono invece i moduli Core del Perl che generalmente offrono tutte le caratteristiche sopra esposte e che vengono usati in IGSuite senza limiti.

Per concludere, ci sono moduli non core scritti in Pure Perl, che si è preferito distribuire unbundle con il pacchetto di IGSuite. Oltre a semplificare la vita dell'utente che vuole installare IGSuite, si ha la certezza che la versione del modulo che sta utilizzando IGSuite in quel momento è in linea con il resto del progetto.

Per quest'ultima tipologia di moduli, non volendo comunque alterare le librerie dell'utente, si è creato un namespace apposito **IG::** (nell'omonima directory) che contiene tutti i moduli non Core scritti in Pure Perl.

Moduli utilizzati in IG

Modulo	Provenienza	Modalità d'Uso	Uso
Carp	Core	Wrappato in IG.pm	Gestione warning e die messages
Config	Core	Utilizzo diretto	Utilizzato negli script di installazione e di autoaggiornamento del sistema
DBI/DBD/Pg	Non Core	Wrappato in IG.pm	Accesso al RDBM
Digest::MD5	Core	Utilizzo diretto	IGWebMail IGFileManager
Fnctl	Core	Wrappato in IG.pm	Utilizzato per flock
IG	**IG Module**	IG Core Module	Definisce il framework di IG
IG::AlgorithmDiff	Non Core	Utilizzo diretto	IGWiki
IG::Bechmark	Non Core	Utilizzo diretto	IG.pm
IG::CGIAjax	Non Core	Wrappato in IG.pm	Implementa Ajax nel framework di IG
IG::CGISimple	Non Core	Wrappato in IG.pm	Accesso ai parametri dei form e ai Cookies
IG::ClassAccessor	Non Core	Utilizzo diretto	Utilizzato da IG::CGIAjax
IG::ConfigSimple	Non Core	Utilizzo diretto	Utilizzato per la gestione delle configurazioni
IG::Cwd	Non Core	Utilizzo diretto	Utilizzato dal framework
IG::DBStructure	**IG Module**	Wrappato in IG.pm	Definisce la struttura del database
IG::DocView	**IG Module**	Utilizzo diretto	Implementa un tiff viewer
IG::FileMMagic	Non Core	Utilizzo diretto	IGFileManager IGWebMail

Modulo	Provenienza	Modalità d'Uso	Uso
IG::GetOptLong	Non Core	Utilizzo diretto	Permette l'esecuzione di IG da riga di comando
IG::HTMLEntities	Non Core	Wrappato in IG.pm	Codifica le entità html
IG::HTTPDate	Non Core	Wrappato in IG.pm	Gestione date
IG::Hylafax	**IG Module**	Utilizzo diretto	IGFax
IG::MimeBase64	Non Core	Utilizzo diretto	IGWiki IGWebMail
IG::MimeWords	Non Core	Utilizzo diretto	IGWiki IGWebMail
IG::Mydes.pm	**IG Module**	Utilizzo diretto	IGWiki
IG::MailAddress	Non Core	Utilizzo diretto	IGWebMail
IG::MailHtmlFilter	Non Core	Utilizzo diretto	IGWebMail
IG::MailPop3Client	Non Core	Utilizzo diretto	IGWebMail
IG::MailSendmail	Non Core	Utilizzo diretto	IGWebMail
IG::Menu	**IG Module**	Utilizzo diretto	Implementa il menu delle feature
IG::MimeBase64	Non Core	Utilizzo diretto	IGWebMail
IG::MimeWords	Non Core	Utilizzo diretto	IGWebMail
IG::ParseRecDescent	Non Core	Utilizzo diretto	Utilizzato da IG::SpreadSheetWriteExcel
IG::PDFExtract.pm	Non Core	Utilizzo diretto	Conta le pagine nei documenti PDF
IG::QuotedPrint	Non Core	Utilizzo diretto	IGWebMail
IG::SpreadsheetWriteExcel	Non Core	Wrappato in IG.pm	Export di dati in file Excel
IG::Thumbnails	**IG Module**	Utilizzo diretto	IGFax IGFileManager IGArchive
IG::TextBalanced	Non Core	Utilizzo diretto	Utilizzato da IG::SpreadSheetWriteExcel
IG::TextBeautify	Non Core	Wrappato in IG.pm	Piccola feature per le textarea
IG::TextTemplate	Non Core	Utilizzo diretto	Non ancora implementato
IG::TimeZone	Non Core	Utilizzo diretto	Gestione delle date
IG::URI	Non Core	Utilizzo diretto	IGWiki
IG::WebMail	**IG Module**	Utilizzo diretto	IGWebMail
IG::WikiFormat	**IG Module**	Utilizzo diretto	IGWiki
Lwp::Simple	Non Core	Utilizzo diretto	isogestd
Text::Wrap	Core	Utilizzo diretto	IGWiki
Time::Local	Core	Utilizzo diretto	checkmsg