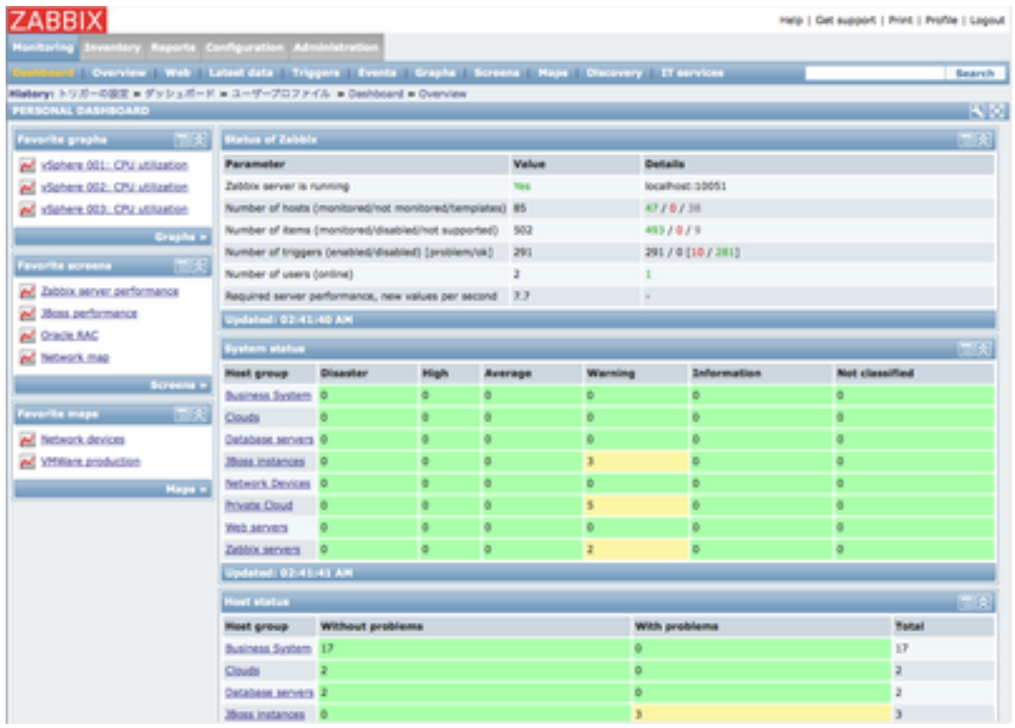
The Zen of ZQL

or how to query Zabbix and keep your cool



Sometimes, you can not get what you want through the Zabbix Frontend. Sometimes, Administrator shall make a sophisticated queries for the History or the Configuration. Sometimes, you shall deploy the configuration “as a code”

In that case, Zabbix API, that what is recommended for the Administrator, and that what most people are using. But …

```
# Create a time range
time_till = time.mktime(datetime.now().timetuple())
time_from = time_till - 60 * 60 * 4  # 4 hours

# Query item's history (integer) data
history = zapi.history.get(itemids=[item_id],
                           time_from=time_from,
                           time_till=time_till,
                           output='extend',
                           limit='5000',
                           )
```

Yes, sometimes you do not have a time to write a code. Sometimes you can not afford to write a code. Sometimes, you do not want to own and support a code. And sometimes… you do not have to write a code, because all you want, is a sending a query. Just a query, nothing more than that. Because you do not cook 5-course meal if all you want is a sandwich.

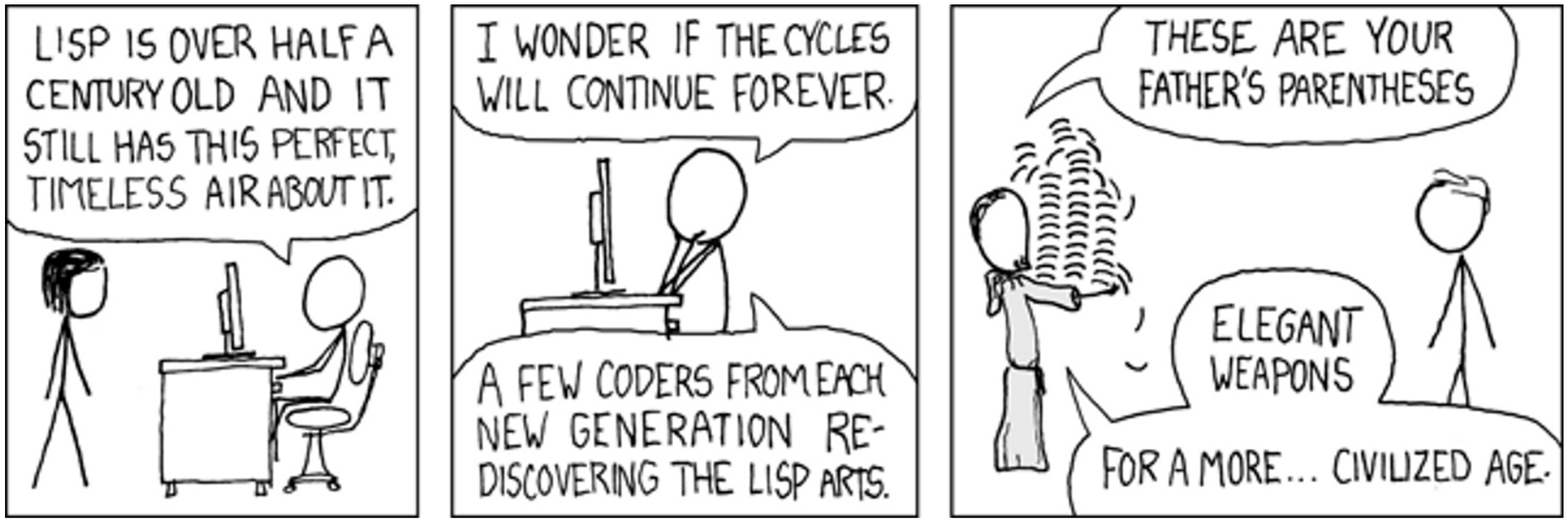
But, Zabbix doesn't have a query language ...

It does now !

and it is called ZQL - #Zabbix Query Language

(ZQL is_a ((LISP-based), Domain specific language))

ZQL is based on the idea of “pipelining”, means the return value of the previous expression is passed to the next one.



ZQL is a “stack-based” query language. Results of the expression are stored in the Stack and the next expression can access those results.


```
##
## Take the list of hosts
##

(Hosts)

##
## Filter them, so you include only the ones where proxy
## do have a value
##

  (Filter TRUE [proxy_hostid Ne 0])

##
## So, now we do have a filtered list of the host in stack,
## Let's merge it with Proxy configuration
##

##
## First, we will put the list of the Proxies to stack
##
(Proxies)

##
## Then merge previous data in the stack with the last one.
## The previous were the Hosts, so if the value proxy_hostid and hostid
## in (Hosts) and (Proxies) will be the same, the proxy value will be
merged with the Host
## and the new value wiull be stored in the stack
##

  (Merge proxy_hostid proxyid)

##
## Tak the last value from the stack. It was merged results, remember ?
##
(Out)
```

https://github.com/vulogov/zq/