Overview of the problem

Configuring large systems is a very complex problem. Often hundreds or thousands of different machines like servers, working-stations (what would be third good example here?) require constant maintenance. Updating, installing new software or reconfiguring only one machine often leads to unexpected results. Performing those actions on such extreme scale and under rigid requirements lead to non-trivial approaches – often creation of new tools which are complex programs on their own. Among them are DSLs (Domain Specific Languages) which are offering standardized techniques for operating with large scale systems. To name a few: (???) LCFG L3 and some others. In this paper (?? paper ??) L3 language is going to be described and then compared to another existing technology – Google's App Engine configuration, finally it would be also tested if L3 can replace solution from Google.

Problems of Declarative languages??

Google's configuration language

On April 2008 Google launched Google App Engine (GAE), cloud computing platform for developing and hosting web applications on servers owned by Google. Since then GAE attracted many customers and became one of the Google’s best products. Among features of GAE is a simplified process of deployment and managing components like databases, virtual-machines and firewalls. Currently except command-line deployer Google is not providing users any specialized tool for configuring their product. Instead a combination of technologies not developed for configuring systems is used: data-serialization format Yaml, procedural programming language Python and templating language designed for webpages and webapps Jinja2. This decision could be driven by popularity of Jinja2 and Python among developers, although this argument does not applies to Yaml, which can be substituted by JSON – well-known data format. Nevertheless current approach makes it harder for users to properly configure their applications. Undocumented behavior and many unexpected nuances are leading to code that is harder to write and maintain.

L3 configuration language

(??) here I will describe features and semantics of the L3? How deeply should I go with that?

Can l3 substitute Google's?

Conclusions