Zoe Documentation

Publicación 0.1.0

GUL UC3M

11 de August de 2014

Índice general

CAPÍTULO 1

Introducción

1.1 ¿Qué es Zoe?

Zoe es una asistente virtual para la organización sin ánimo de lucro del *Grupo de Usuarios de Linux UC3M* (GUL UC3M). Su objetivo es el de realizar tareas repetitivas y que consumen mucho tiempo de forma asíncrona, desacoplada y perezosa.

1.2 ¿Cómo funciona?

Zoe tiene dos componentes principales: el *servidor de Zoe*, y un grupo de *agentes*, pequeños programas que realizan algunas pequeñas y bien definidas tareas tales como captura de estadísticas de los sistemas de la organización, envío de emails, generación de informes y control de alarmas.

El *servidor* implementa un *bus de mensajes* que los agentes utilizan para intercambiar mensajes de forma asíncrona. Zoe también puede recibir comandos de los administradores del sistema para realizar tareas bajo demanda.

1.3 Licencia

El servidor Zoe se publica bajo la licencia MIT. Sin embargo, la licencia de los agentes puede variar dependiendo del desarrollador del agente.

CAPÍTULO 2

Empezando

Esta sección te ayudará a empezar con el kit de iniciación de Zoe.

2.1 Requisitos

- OS X o Linux (BSD también debería servir, aunque no ha sido probado)
- Bash
- Java 1.7
- Python 3
- Perl 5

2.2 Estructura de directorios

El kit de iniciación de Zoe tiene la siguiente estructura:

```
agents/         Installed Zoe agents
cmdproc/        Administrator commands in natural language
etc/            Configuration files
lib/            Libraries and agent dependencies by language
    bash/
    java/
    perl/
    python-dependencies/
    python/
logs/           Agent log files
mailproc/       Actions to execute when receiving mails
server/         Zoe server binaries
var/            Databases and temp files generated by agents
zoe.sh          Zoe script
```

2.3 Instalación

Descarga el código del kit de iniciación de Zoe y ponlo en el directorio desde el que quieras ejecutar *Zoe*, como por ejemplo `/home/zoe`. Navega hasta el directorio y haz los siguientes scripts ejecutables:

```
$ chmod +x zoe.sh

$ chmod +x etc/environment.sh
```

2.4 Configuración

Los archivos de configuración de *Zoe* se encuentran en el directorio `etc/`. Estos son los archivos base que se encuentran en el directorio:

```
environment.sh          Script to set environment variables
zoe-users.conf          Zoe users configuration
zoe.conf                Configuration of agent ports and topics
```

2.4.1 Entorno

El script `environment.sh` establece variables tales como el *host del servidor*, *credenciales de twitter*, *información de jabber*, etc. Tiene el siguiente contenido:

```
# Typical values
export ZOE_SERVER_HOST=localhost
export ZOE_SERVER_PORT=30000

# Get this from twitter
export zoe_twitter_consumer_key="..."
export zoe_twitter_consumer_secret="..."
export zoe_twitter_access_token="..."
export zoe_twitter_access_token_secret="..."

# Parameters for Google Talk
export zoe_jabber_host="talk.google.com"
export zoe_jabber_port="5222"
export zoe_jabber_user="..."
export zoe_jabber_password=""

# Parameters for a GMail account
export zoe_mail_smtp="smtp.gmail.com"
export zoe_mail_smtp_port="587"
export zoe_mail_pop3="pop.gmail.com"
export zoe_mail_pop3_port="995"
export zoe_mail_enable_dkim="false"
export zoe_mail_user="$zoe_jabber_user"
export zoe_mail_password="$zoe_jabber_password"
```

Nunca des a Zoe tus propias credenciales de twitter/jabber. Crea una nueva cuenta si pretendes usarlos.

- La información requerida para conectarse a una cuenta de Twitter se puede obtener del perfil de usuario de Twitter.
- Por defecto, las cuentas de Jabber y GMail usan las mismas credenciales debido a que el servicio de Google Talk (Hangouts) utiliza Jabber. Puedes utilizar otros servicios de Jabber y correo electrónico si lo deseas.

2.4.2 Usuarios

El archivo `zoe-users.conf` sigue esta estructura:

```
[subject admin]
name = Admin
twitter = your twitter name without @
preferred = twitter
jabber = your_jabber_id@wherever.com
mail = your_email@address.com
alias = god master

[group admins]
members = admin

[group broadcast]
members = admin
```

Cada usuario está identificado por la etiqueta `[subject USUARIO]` y tiene los siguientes atributos:

- `name`: El nombre del usuario (ej. Zoe).
- `twitter`: El nombre de usuario de twitter (ej. gul_zoe).
- `preferred`: La forma de comunicación con Zoe preferida por el usuario. Los posibles valores son `twitter`, `jabber`, `mail`.
- `jabber`: La dirección de jabber del usuario. Si utilizas *Google Hangouts*, las direcciones de jabber siguen un formato `abcdefghijklmnopqrstuwxyz1@public.talk.google.com`. Si no sabes cuál es la dirección, puedes intentar hablar con *Zoe* por Jabber y obtener la dirección de los registros.
- `mail`: La dirección de correo del usuario.
- `alias`: El alias del usuario.

Algunos agentes pueden utilizar *grupos* para envíar correos a varios usuarios al mismo tiempo o comprobar permisos. Se puede definir un nuevo grupo usando la estructura `[group NOMBRE DE GRUPO]` y añadiendo a sus usuarios en el atributo‘‘members‘‘.

2.5 Primera ejecución

Desde una terminal, navega al directorio raíz de *Zoe* (ej. `/home/zoe`) y ejecuta:

- Carga la configuración de *Zoe*:

  ```
  $ . etc/environment.sh
  ```

- Ejecuta *Zoe*:

  ```
  $ ./zoe.sh start
  ```

Verás que la terminal muestra los agentes que se han iniciado. Puedes comprobar el estado de estos agentes ejecutando:

```
$ ./zoe.sh status
```

Si hay agentes muertos, quizá quieras comprobar sus archivos de registro (en el directorio `log/`). La mayoría de los errores suelen deberse a errores de configuración.

Eso es todo, ahora tienes una instancia de *Zoe* funcional y ejecutándose en tu máquina. Puedes detener el servidor mediante:

```
$ ./zoe.sh stop
```

Lo cuál detendrá también todos los agentes en ejecución.

CAPÍTULO 3

Agentes

En esta sección encontrarás información relacionada con los agentes.

3.1 ¿Qué es un agente?

Los agentes son pequeños programas a los que *Zoe* tiene acceso para poder realizar diferentes acciones. Cada agente se encarga de tareas específicas y se comunican entre ellos mediante el *bus de mensajes* del servidor. El servidor crea sockets para cada agente dinámicamente o usando el archivo `etc/zoe.conf`, que contiene el puerto de cada agente y una lista de agentes relacionados con un tema específico.

3.1.1 Estructura

Los agentes suelen consistir en 1-2 archivos, aunque esto depende enteramente del desarrollador. La mayoría de agentes están escritos en *Python*, pero hay disponibles librerías para escribir agentes en lenguajes como *Bash*, *Java*, *Python*, etc.

El script principal se coloca en el directorio `agents/NOMBRE_AGENTE`. El agente también puede incluir *comandos de lenguaje natural* para ejecutar acciones bajo demanda sin la necesidad de usar la línea de comandos para mandar el mensaje al servidor manualmente. De esta forma, se puede *hablar* con *Zoe* mediante, por ejemplo, Jabber. Estos comandos de lenguaje natural se colocan en el directorio `cmdproc/`.

3.2 Mensajes

3.2.1 Estructura de mensajes

Los agentes intercambian mensajes mediante el *bus de mensajes* del servidor. Estos mensajes tienen una estructura *clave-valor* de la forma:

```
key1=val1&key2=val2&key3=val3...
```

Por ejemplo, el agente `broadcast` puede enviar un mensaje a un usuario específico. La cadena para hacerlo sería algo parecido a esto:

```
dst=broadcast&tag=send&msg=Hello World!&to=john
```

Analizando el mensaje:

- `dst`: Agente al que va destinado el mensaje.

- `tag`: Acción a realizar. Puede haber varios `tag` en el mensaje.
- `msg`: Mensaje que mandar al usuario.
- `to`: Usuario al que mandar el mensaje. El nombre se comprueba con la lista de usuarios para ver cuál es la dirección de John y su método de contacto preferido.

La cadena puede variar dependiendo de cada agente, pero la mayoría tendrán partes comunes como `dst` y `tag`.

3.2.2 Enviando mensajes al servidor

Aparte del intercambio de mensajes de los agentes, también se puede enviar un mensaje al servidor manualmente para que un agente ejecute alguna acción. Tomando el anterior ejemplo del agente `broadcast`, el mensaje se podría enviar como:

```
$ echo -n "dst=broadcast&tag=send&msg=Hello World!&to=john" | nc ZOE_HOST ZOE_PORT
```

Ahora, el archivo `logs/broadcast.log` tendría una entrada con el mensaje recibido y la acción realizada (o cualquier error mientras se llevaba a cabo).

3.3 Instalando nuevos agentes

Hay dos formas de instalar un nuevo agente en una instancia de *Zoe*.

3.3.1 Instalación manual

Para instalar un agente manualmente hay que descargar el código y colocar los archivos en la instalación de *Zoe* siguiendo las instrucciones del desarrollador. Normalmente será una mera cuestión de copiar y pegar archivos y quizá modificar alguna configuración. La próxima vez que se inicie el servidor, el agente debería funcionar.

3.3.2 Usando el gestor de agentes

El kit de iniciación de Zoe debería incluir un agente llamado *zam*. Se trata del Gestor de Agentes de Zoe, cuyo fin es el de facilitar la instalación y gestión de agentes en la instancia de *Zoe*. Si el gestor de agentes no está instalado, puedes instalarlo fácilmente de forma manual descargando el código y siguiendo las instrucciones.

El gestor utiliza repositorios git para obtener e instalar agentes. Por ejemplo, el agente dummy tiene la siguiente URL de git:

```
https://github.com/rmed/dummy_agent.git
```

Si se quisiera decirle al gestor que instalara este agente, se podría mandar el siguiente mensaje al servidor:

```
$ echo -n "dst=zam&tag=install&name=dummy&source=https://github.com/rmed/dummy_agent.git" | nc ZOE_H
```

O si prefieres usar lenguaje natural y Jabber:

```
Zoe, install agent "dummy" from "https://github.com/rmed/dummy_agent.git"
```

Para más información sobre el gestor y todos los comandos disponibles, por favor échale un vistazo a la wiki de zam.