

Squeak 3.10, visto desde adentro.

Es conocido que despues de la partida de los “padres fundadores” que llevaban adelante la comunidad (Squeak Central), hubo un período de desorientación.

Esto se debió a que habiendo varios miembros de renombre, no se destacó especialmente alguno.

La conducción de la comunidad en este momento es un Board que se elige anualmente y que consta de 7 miembros.

Este es el Segundo Board

Tambien ocurrió que empezaron a proliferar los “forks”, es decir desarrollos a partir de Squeak, en algunos casos resultando en productos especializados como Scratch, Sophie, Croquet, para nombrar algunos.

Algunos miembros empezamos a trabajar en la idea de “modularizar” la imagen.

Esto se había intentado con el Squeak 3.3,.

El que no se haya logrado, no tiene nada que ver como algunos piensan porque la idea no fuera la correcta.

Lamentablemente lo que ocurrió es que perdimos a Henrik Gederik a la temprana edad de 32 años.

Luego de tomar contacto con su trabajo, me da la impresion que Henrik era un adelantado en algunos años con respecto al tema.

Si pensamos en el tamaño de la imagen , tenemos

Minima imagen posible (2+2 y exit) de Alejandro Reimondo

Minimo Squeak posible (52 clases) de Craig Latta (Spoon)

Kernel o Consola Squeak de Pavel Krivanek

Minimo Squeak con Morhic (basado en 3.7) SqueakLight

MinimalMorphic o Squeak futuro de Pavel Krivanek (con algunas contribuciones mias)

Squeak 3.10 base.

El Board anterior llamó a la constitución de un equipo para llevar a cabo la release.

Personalmente venía colaborando con Pavel Krivanek en el MinimalMorphic.

Cuando me distingue Ralph Johnson invitandome a trabajar con el , llamo su atención hacia el trabajo de Pavel e intento que estemos todos en el mismo team.

Lamentablemente, no lo conseguí.

Los objetivos de la release entre otros han sido ver cuales paquetes podían descargarse en forma segura.

Tambien establecer un control de calidad mucho mas estricto, con un procedimiento bastante extricto bassdo en test con SUnit

En las palabras de Ralph:

```
The goal of the 3.10 release is to improve the process of making Squeak releases and to improve the quality of the code.
```

The release team is not concentrating on adding features or packages, rather in improving testing, in making Squeak more modular, and in getting rid of bad code. However, we expect to add features and packages. These will come from you. Naturally, they must pass the quality standards, and we would prefer to add things that have been tried out by more than the author, and that are generally considered to be a good idea.

Asi tenemos como requisitos para que algo sea incorporado en la imagen.

Debe existir en Mantis un reporte.

El reporte debe tener, además del código propuesto, test .

Cuando el código que se desea incorporar se considere maduro, colocarlo en la página

<http://wiki.squeak.org/squeak/5934>

Ahi vemos la excelente contribución de mi amigo Jerome Peace (wiz)

* Prove the changes work as intended.

o This often means:

1. Loading bug tests before fixes.
2. Run the bug test. (It should fail)
3. Install the fix.
4. Run the bug test. (Now it should pass.)

1. Then prove all standard quick tests work.
2. Then prove all standard long tests work.
3. And as icing on the cake prove all tests work together

Inconvenientes que hemos encontrado.

Esta idea actual de paquetes , tal vez comienza en trabajos internos de Impara .

Es tomada por el release team anterior y llevo toda la release 3.9 armar esta forma de trabajo basada en Monticello.

Es conocido que durante el 3.9 se pierde el poder actualizar la imagen en forma sencilla. (boton load code updates)

Tambien , debido a limitaciones en la estructura actual de Squeak, se deben realizar dos Fuentes.

Mis sugerencias al iniciar la release están basadas a mis experimentos con el SqueakLight.

Propongo retomar la forma de trabajo que se tiene practicamente en todas las ramas derivadas de Squeak, excepto 3.9.

Se comienza de esta forma, pero lamentablemente alguien convence a Ralph y esto se abandona.

Tambien sugiero que si se va a usar Monticello (yo me opuse), se haga una “cumbre Monticello” , con los principales desarrolladores y gente con más experiencia.

Ralph decide hacer su propia version, que está algo cuestionada en los últimos tiempos, pero que mientras estemos en el team y el no decida otra cosa , es la que se debe utilizar.

Esta forma de trabajo es sumamente tediosa y ha probado en este ultimo año no ser adecuada.

Por lo menos para manejar una release completa.

Tenemos casos de algunos informes de Mantis imposibles de realizar con esta técnica y otros que nos han dado muchos dolores de cabeza.

En particular hemos tenido dos puntos de quiebre, uno al incorporar los cambios de Pavel Krivanek, 7109ChangesOrganizer-pk.cs y otro que es el que tiene parado en este momento el proceso , 7142ReformatOnscanForEqSmallConstant-di.cs.

Este ultimo es un cambio propuesto por Dan Ingalls y que al tomarlo yo ya tenía un paquete de Monticello creado.