Enhancing SqueakNOS functionalities

Guido Chari, Javier Pimás

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires.

Instituto de Ciencias
Universidad Nacional General Sarmiento

November, 2010

SqueakNOS

“Operating System: An operating system is a collection of things that don’t fit into a language. There shouldn’t be one.” -Dan Ingalls

SqueakNOS

“Operating System: An operating system is a collection of things that don’t fit into a language. There shouldn’t be one.” -Dan Ingalls

- Two minutes (most of you already know about it)

SqueakNOS

“Operating System: An operating system is a collection of things that don’t fit into a language. There shouldn’t be one.” -Dan Ingalls

- Two minutes (most of you already know about it)
- Gerardo Richarte is one of the creators and he already presented it several times (ESUG, Smalltalks)

SqueakNOS

“Operating System: An operating system is a collection of things that don’t fit into a language. There shouldn’t be one.” -Dan Ingalls

- Two minutes (most of you already know about it)
- Gerardo Richarte is one of the creators and he already presented it several times (ESUG, Smalltalks)
- Our M.Sc. thesis relates to extending it (directed by Hernán Wilkinson and Gerardo)

What is SqueakNOS

An Operating System?

- Reificates OS concepts inside the image
- Interacts with devices using messages
- Developed in Smalltalk (Squeak/Pharo) and a bit of C
- Requires (small) changes to the Object Engine
- Doesn't require an OS behind it

What is SqueakNOS

An Operating System?

- Reificates OS concepts inside the image
- Interacts with devices using messages
- Developed in Smalltalk (Squeak/Pharo) and a bit of C
- Requires (small) changes to the Object Engine
- Doesn't require an OS behind it

So, is it an Operating System?

What is SqueakNOS

An Operating System?

- Reificates OS concepts inside the image
- Interacts with devices using messages
- Developed in Smalltalk (Squeak/Pharo) and a bit of C
- Requires (small) changes to the Object Engine
- Doesn't require an OS behind it

So, is it an Operating System?

- OS can be considered a reification of the machine and its hardware, playing the role of a Virtual Machine

Missing concepts and functionalities (to analyze)

- Support for many basic devices and mechanisms
  - **Hard disk (persistency)**
  - **Hardware paging**
- Multi-core
- **Memory Management**
- Security/Protection (users)
- Aplications?
- Tests

Motivation

We wanted to experiment with memory management mechanisms

Motivation

We wanted to experiment with memory management mechanisms

1. Virtual Memory
2. Paging
3. Automatic image persistency
4. Transactional memory

Motivation

We wanted to experiment with memory management mechanisms

1. Virtual Memory
2. Paging
3. Automatic image persistency
4. Transactional memory

Philosophy

- Do as much as possible in Smalltalk

Prerequisites

Prerequisites

We needed a dynamic environment

- To browse source (not decompiled)
- To debug with the real hardware
- To save changes (without rebooting!)

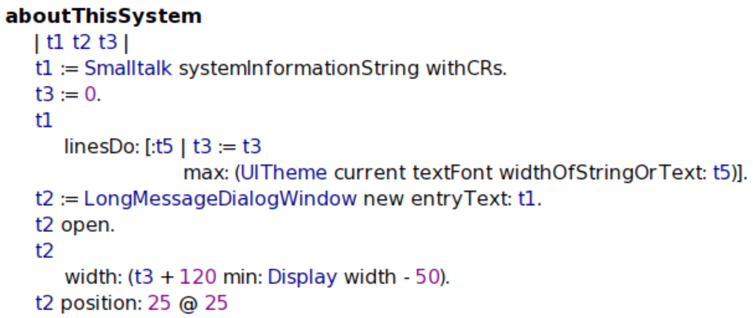
Prerequisites

We needed a dynamic environment

- To browse source (not decompiled)
- To debug with the real hardware
- To save changes (without rebooting!)

We needed to work in SqueakNOS as we do in Squeak

- We needed persistency support



```
aboutThisSystem
    | t1 t2 t3 |
    t1 := Smalltalk systemInformationString withCRs.
    t3 := 0.
    t1
        linesDo: [:t5 | t3 := t3
                        max: (UITheme current textFont widthOfStringOrText: t5)].
    t2 := LongMessageDialogWindow new entryText: t1.
    t2 open.
    t2
        width: (t3 + 120 min: Display width - 50).
    t2 position: 25 @ 25
```

Figure: No sources!

We'd rather not have files...

... but we need them (just for now?)

We'd rather not have files...

... but we need them (just for now?)

1. Interaction with other OS
2. Squeak's architechture

We'd rather not have files...

... but we need them (just for now?)

1. Interaction with other OS
2. Squeak's architechture

... but in order to work with files...

- We need a filesystem.

We'd rather not have files...

... but we need them (just for now?)

1. Interaction with other OS
2. Squeak's architechture

... but in order to work with files...

- We need a filesystem.

... but in order to work with a filesystem...

- We need a storage device and a driver.

Investigation

Specifications

Investigation

Specifications

- ATA (Standards 1 to 6)
- PCI Bus
- FAT32 Filesystem
- x86
  - Paging
  - Segmentation
  - Control Registers
  - PIC
- Multiboot

Investigation

Informal

- Object Engine
  - Garbage Collector
  - Bytecode encoding and interpretation
  - Primitive methods
  - C interfacing
  - VMMaker
  - Plugins
- Bootloading process

FAT32 Filesystem

1. MS/DOS Filesystem
2. Still used frequently
3. Simple but inefficient
4. Based on a big index (FAT)

Demo

FAT32 Filesystem

1. MS/DOS Filesystem
2. Still used frequently
3. Simple but inefficient
4. Based on a big index (FAT)

Demo

Main benefits

- Really expressive knowledge representation
- Unbeatable dynamism

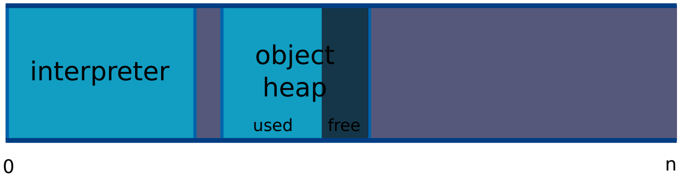
Save the image

Save the image

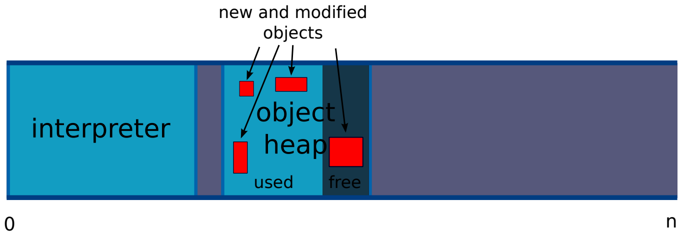
Challenges

1. Snaphshot is a very complex primitive
2. Some things to care of
   - Garbage collection
   - Atomicity
3. Atomicity goes against resolving things in Smalltalk

Save the image

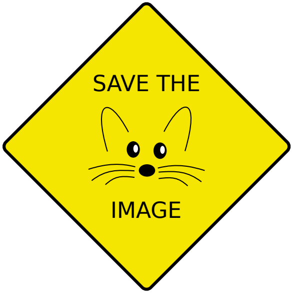


Save the image



Save the image

Challenges



1. Snaphshot is a very complex primitive
2. Some things to care of
   - Garbage collection
   - Atomicity
3. Atomicity goes against resolving things in Smalltalk

Save the image

Challenges

1. Snaphshot is a very complex primitive
2. Some things to care of
   - Garbage collection
   - Atomicity
3. Atomicity goes against resolving things in Smalltalk
4. File saving isn't atomic anymore (done in Smalltalk now)
5. We don't intend to change snapshoting (for now)
6. Just implement little hacks to be able do it

Save the image

Challenges

1. Snaphshot is a very complex primitive
2. Some things to care of
   - Garbage collection
   - Atomicity
3. Atomicity goes against resolving things in Smalltalk
4. File saving isn't atomic anymore (done in Smalltalk now)
5. We don't intend to change snapshoting (for now)
6. Just implement little hacks to be able do it

... so the challenge is ...

How do we pretend being atomic when we are not atomic?

Various paths

Two explored ones

Various paths

Two explored ones

- Changing (a bit) snapshot primitive

Various paths

Two explored ones

- Changing (a bit) snapshot primitive
- Use paging protection mechanism

Others

Various paths

Two explored ones

- Changing (a bit) snapshot primitive
- Use paging protection mechanism

Others

- Don't save it at all

Various paths

Two explored ones

- Changing (a bit) snapshot primitive
- Use paging protection mechanism

Others

- Don't save it at all
- There should be more, suggestions?

Making file writting atomic

We can't write files atomically... but we can fake.

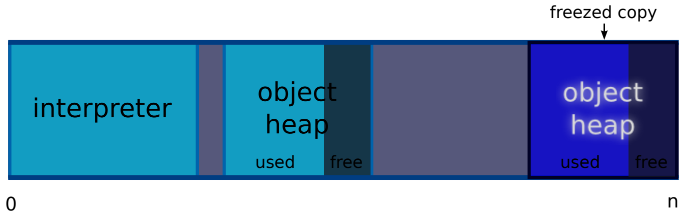
Remember that ...

- we want to modify the object engine as little as possible.
- we didn't want to modify snapshot primitive.

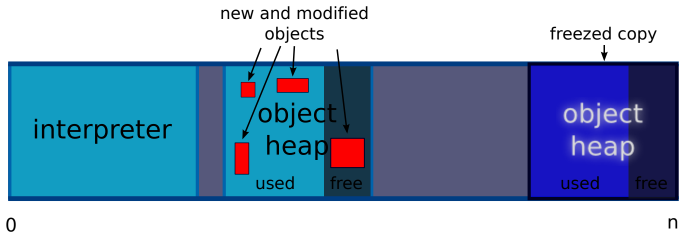
Solution

1. Hook into primitive file write
2. Replace file write (we don't have syscalls) with a memcpy
3. When out of primitiveSnapshot, save memory copy into file within Smalltalk

Solution


freezed copy
interpreter
object heap
used
free
object heap
used
free
0
n

Solution


new and modified objects
freezed copy
interpreter
object heap
used
free
object heap
used
free
0
n

Solution

Demo

Paging

Figure: Not this!

Paging

A memory-management scheme

- Widely known and supported by most hardware
- Virtual memory
- Protection
- Transparent

Figure: Not this!

Paging

A memory-management scheme

- Widely known and supported by most hardware
- Virtual memory
- Protection
- Transparent

Figure: Not this!

Translation structures

- Page Directory
- Page Tables
- Page Table entries

SqueakNOS Paging

Figure: Not this!

SqueakNOS Paging

Figure: Not this!

What have we done

- Reify paging structures as objects
- Implemented page fault handlers
- This means hardware page faults handled by Smalltalk code!!!
- Halt on page fault interrupts could be set...
- ... page fault interrupts can be debugged

How we did it

- More difficult than other interrupts
- Heavy usage of Alien callbacks

Problem

native interrupt handler

```
save status
signal interrupt semaphore
restore status
continue executing
```

image interrupt handler (sleeping on independent thread)

```
[true] whileTrue:
    [wait on interrupt semaphore.
    resolve interrupt ]
```

Figure: Asynchronic interrupts handling

Problem

native interrupt handler

```
save status
signal interrupt semaphore
restore status
continue executing
```

X

image interrupt handler (sleeping on independent thread)

```
[true] whileTrue:
    [wait on interrupt semaphore.
    resolve interrupt ]
```

Figure: Asynchronic interrupts handling

Solution

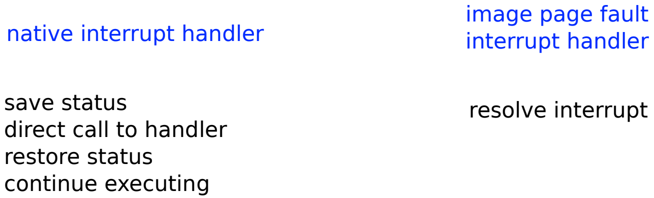


Figure: Synchronic interrupts handling

SqueakNOS paging

```
void pageFaultISR(unsigned long errorCode) {
  extern Computer computer;
  unsigned long virtualAddressFailure;
  asm volatile("movl %%cr2, %0" : "=a" (virtualAddressFailure));
  sti();
  computer.pageFaultHandler(virtualAddressFailure);
}
```

Demo

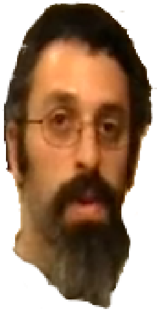
Taking advantage of paging

Going back to image saving and atomicity

Taking advantage of paging

Going back to image saving and atomicity

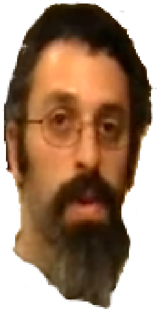
- While at ESUG 2010 Gerardo suggested us using paging to emulate atomicity



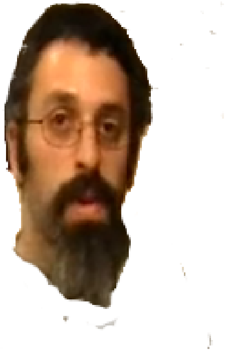
Taking advantage of paging

Going back to image saving and atomicity

- While at ESUG 2010 Gerardo suggested us using paging to emulate atomicity
- How?



Taking advantage of paging



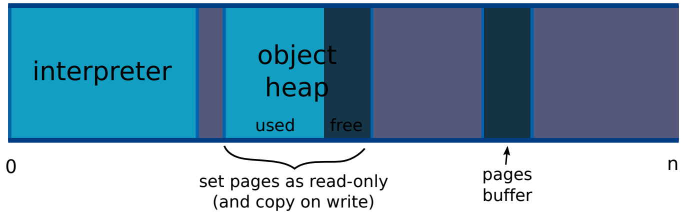
Going back to image saving and atomicity

- While at ESUG 2010 Gerardo suggested us using paging to emulate atomicity
- How?

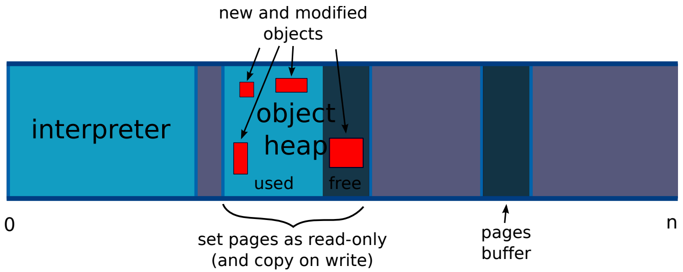
Idea

- Set image memory pages as read-only.
- Implement a copy-on-write (COW) page fault handler
- Write to the filesystem the original pages

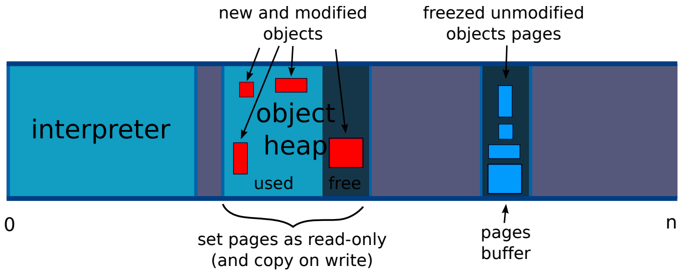
Solution


interpreter
object heap
used
free
0
n
set pages as read-only (and copy on write)
pages buffer

Solution


new and modified objects
interpreter
object heap
used
free
0
n
set pages as read-only (and copy on write)
pages buffer

Solution


new and modified objects
freezed unmodified objects pages
interpreter
object heap
used
free
0
n
set pages as read-only (and copy on write)
pages buffer

Taking advantage of paging

Problem!

- Invoking **any** method involves creating and modifying objects
- What happens with the first memory writes?

Taking advantage of paging

Problem!

- Invoking **any** method involves creating and modifying objects
- What happens with the first memory writes? Recursion!

Taking advantage of paging

Problem!

- Invoking **any** method involves creating and modifying objects
- What happens with the first memory writes? Recursion!
- How do we solve it?

SqueakNOS paging revisited (experimental)

```
void pageFaultISR(unsigned long errorCode) {
  extern Computer computer;
  unsigned long virtualAddressFailure;
  asm volatile("movl %%cr2, %0" : "=a" (virtualAddressFailure));
  sti();
  if ((errorCode & 1) == 1){
    /** Protection page fault **/
    if(computer.snapshot.pagesSaved < computer.snapshot.pagesToSave){
      saveSnapshotPage(&computer, virtualAddressFailure);
    } else {
      computer.pageFaultHandler(virtualAddressFailure);
    }
  } else {
    /** page not present **/
    computer.pageFaultHandler(virtualAddressFailure);
  }
}
```

Other things to try

- NativeBoost instead of C
- Which is best: FFI, Alien, CObjects (new)
- Object memory (no more image saving?)
- Cog
- Many cores

Conclusions and future work

By working in SqueakNOS we could ...

1. model low level concepts in Smalltalk
2. better understand these low level concepts
3. figure out new innovative ways of taking advantage of hardware

This is still a work in progress ...

- Tests (a lot)
- Figuring other ways of persisting objects
- Migrating tools to smalltalk (bash scripts, build tools)
- Benchmarking tools
- Performance tunning

¡That's (almost) all folks!

Questions?

Contact

Us

- Guido Chari (charig@gmail.com)
- Javier Pimás (jpimas@dc.uba.ar)

Website

- http://squeaknos.blogspot.com (new site)

Now it is, that's all folks!