CHAPTER 4

Current Development Effort

Now we explain the current development efforts and why there are done that way and their impact on the resulting system. Scenario presented in the first section are also a reason why we are doing them. The list is not exhaustive but we are conscious that this is important to finish a task before passing to the next one.

Note that some efforts are simpler to integrate or require less development effort. We take a pragmatic approach: if a change is ready to get integrate we do it. Now we are conscious that we should finish some work because else we risk to lose them.

4.1 A Robust and Extensible System Events

We need an infrastructure to easily be able to plug multitouch, Genie like behavior (Genie was a hand recognition system developed by Nathanael Schaerli in 2002) and experiment with new devices or UI metaphors.

Context. Before Pharo, the VM filled up a buffer for each new event and the system checked at regular interval if there was something present (See Figure 4.1).

Then different UI frameworks (Morphic and MVC) were calling the InputSensor which had a polling logic to see if a new events was present in the buffer. In the case of Morphic, HandMorph interprets itself this buffer and creates and interprets events. Notice that there was confusion and code in Morphic was directly calling InputSensor logic and not the HandMorph one, breaking layers. In addition HandMorph handles events using an explicit case statement logic which hampers extensibility.

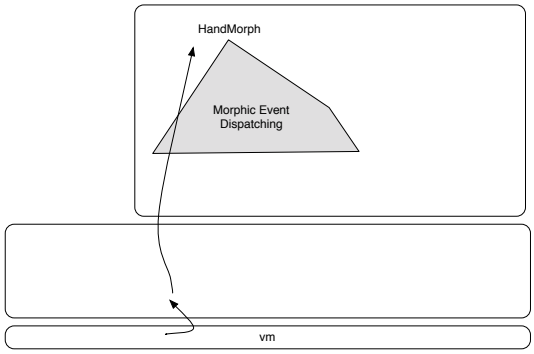

Figure 4.1: Old days: VM filled up a buffer and the system checked at regular interval if there was something present. HandMorph interprets this buffer and creates and interprets events.

Current ongoing effort. Since Pharo 1.0, several actions were made:

- An effort to use an event-driven structure was made first by Michael Rueger then followed up by Igor Stasenko and Stéphane Ducasse. An input fetcher and a input handler were introduced. This new infrastructure was already validated by the fact that it is possible to build an independent UI event interpretation loop.
- Igor Stasenko built a wait-free collection so that the events are managed in a solid data-structure.
- We systematically fixed layer violations. Morphic code should not shortcut anymore the HandMorph logic and should not access low-level input sensor.
- Improved modal logic. We introduced abstraction to HandMorph to simplify the handling of modal interaction (dropping certain event while a certain condition holds). It simplifies the logic and encapsulates it.

Pharo 1.4 can now drop the polling semantics. All the VMs support now the event infrastructure. This moves could have been done earlier but the Windows VM did not integrate a fix adding a semaphore input during one and half year. With the new VM infrastructure (jenkins and GIT/CMake) put in place by Igor Stasenko. We can now compile daily all the platforms.

Current ongoing effort.

- We are introducing a registration mechanism, so that it can be easy to build a record mechanism of events or other broadcasting systems.
- We are introducing a major refactoring (dashed line in Figure 4.2): the event fetcher interprets the eventBuf and creates SystemInputEvents. Such events

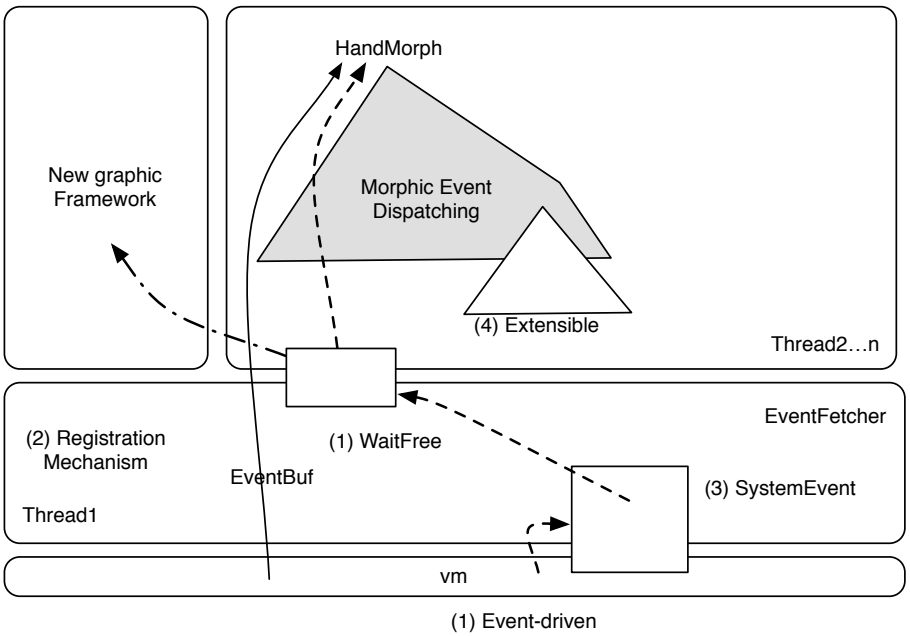

Figure 4.2: Current and future Pharo effort.

fully encapsulate the logic and behavior of low level events (avoid complex logic spread in HandMorph, duplicated code, ...) are then passed to the handler queue. Then HandMorph manipulates such objects and can decide to republish them.

What is left to do.

- We are rewriting the HandMorph logic to not use anymore hardcoded case statements. In fact there are at least two case statements (one for building morphic events from raw buffer, and one for defining the way morphic works). We have nearly fixed the first one using double dispatch.
- We should migrate HostMenus processing to an adequate place, the pasteUpMorph should be responsible to publish its menu. Now we may simply deprecate the host menu facilities because of development resource limitations.
- We should probably make sure that the libraries code that has interaction do it via Morph and not direct input sensor. For example, Rectangle fromUser should be packaged with Morph and use HandMorph API and not directly invoke input sensor.
- The second case statement logic should be rethought to support a smoother extensibility.

- We should validate the extensibility by evaluating how easy it is to introduce multitouch events and Genie in the system.

Open questions. Fernando Olivero proposed a new way to manage event and devices at the level of Morphic. Basically it acknowledge that HandMorph has too much responsibilities that the logic of the event processing is complex, brittle and difficult to extend. The model proposed by Fernando is based on decomposing responsibilities and using state pattern to identify double-click and other state-oriented situations. One question is do we propose a state-machine that can be integrated in HandMorph in a compatible way or is it better to define a separate infrastructure and after a moment to remove HandMorph. Frankly we do not have the answer (yes no magic ball here). Fernando already integrated his model with the refactoring we are performing. A possible path is to integrate our refactorings, finish the first two steps of the previous sections and evaluate how the solution of Fernando is a good alternative.

Business Value. Touch Event, MultiTouch and others like new user interface development can radically benefit from our effort. We should be able to plug easily multiple touch devices.

4.2 Rewrite of Filesystem/Streams

Context. Current file support is rather old, cumbersome and counter intuitive. Pharo as a scripting language should have a good and easy to use file library.

Possible Solutions. We want to use the FileSystem library. However, the state of FileSystem is not totally at the level required to support a replacement of the existing one.

- We started an effort to add comments in the code.
- We improved the FileSystem library.
- We started to write a documentation in a form of a chapter for the Pharo by Example volume two book.

What is left to do.

- Evaluate if the complete API is implemented (for example rename was missing).
- Migrate existing code to use the FileSystem library.