# A guide to techno-galactic software observation

As part of the Techno-Galactic Software Observatory, we collectively produced a Software Survival Guide. The guide proposes ways to achieve critical distances from the seemingly endless software systems that surround us and will offer practical tools for the tactical (mis)use of software, empowering/enabling users to resist embedded paradigms and assumptions. The guide will include a description of the myths and realities of software, a list of currently available methods for approaching software including their risks and benefits, and a glossary of terms.

Participants will (can? are invited to?) contribute to the guide by naming and describing methods of observing software, by nominating and defining terms for the glossary, and by sharing personal stories of close encounters with software.

> *I am less interested in the critical practice of reflection, of showing once-again that emperor has no clothes, than in finding a way to diffract critical inquiry in order to make difference patterns in a more worldly way.* [1]

Methods for (engaged) observation of software

An inclusive list of methods that appeared during the worksession. Question: are we validating if they are “interesting or even”engaged”? If the object is produced by the mode of observation. or what if the mode of observation is produced by the object . . . Are we producing a guide that we want to exist or producing a document of what happened (a bit of both?) Name the things that we might otherwise take for granted. How do we name what’s different about software and observation across time, without rendering the new or the old into an ideal?

From the call:

---

[1] Haraway, Donna: “Modest Witness: Feminist Diffractions in Science Studies”. In: Galison, Peter/Stump, David J. (eds.): *The Disunity of Science: Boundaries, Contexts, and Power*. 1996, 428–442.

From 7 until 12 June, the Techno-Galactic Software Observatory will explore practices of proximate critique with and of software. This worksession has been called an 'Observatory' because we are interested in different ways to look at software, and at the implications of how it is currently probed and scrutinized as part of its production. Techno-galactic because we want to extend the observation to include different scales of computation, of software communities and their political economies.

With the rise of online services, software use has been increasingly knitted into production, while suggesting that these roles constitute separate realms. This has an effect on the way software is used and produced, and radically alters its operative role in society. The shifts ripple across galaxies, through social structures, working conditions and personal relations, resulting in a profusion of seamless apparatuses that optimize and monetize individual and collective flows of information. The diffusion of software services affects the personal, in the form of intensified identity shaping and self-management. It also affects the public, as more and more libraries, universities and public infrastructures rely on "solutions" provided by private companies.

Given how fast these changes resonate and reproduce, there is a growing urgency to engage in a critique of software that goes beyond taking a distance, and that deals with the fact that we are inevitably already entangled. How can we interact, intervene, respond and think with software? What approaches can allow us to recognize the agency of different actors, their ways of functioning and their politics? What methods of observation enable critical inquiry and affirmative discord?

Working with theories of software and computation developed in academia and elsewhere, our priority is to ground the Observatory in hands-on exercises and experiments that might have an effect on how software is done. At the Techno-Galactic Software Observatory we will explore methods developed in the context of software production, hacker culture, software studies, computer science research, Free Software communities, privacy activism, and artistic practice to experiment with ways to stay with the trouble of software.

During three sessions of two days, we will be collectively inspecting the space-time of computation and probing the universe of hardware-software separations. We will try out various perspectives and methods to look at the larger picture of software as a concept, as a practice, and as a set of paradigms.

Worksessions are intensive transdisciplinary moments, organised twice a year by Constant. We aim to provide conditions for participants with different experiences and capabilities to temporarily link their practice and to develop ideas, prototypes and research projects together. We primarily use Free, Libre and Open Source software and material that is available under Open Licenses. This worksession calls for software-curious people of all kinds. Constant has invited 9 participants from different disciplines and backgrounds to prepare the subject. To ask questions about software from different types of engagement will require a heterogeneous group of participants with different levels of expertise, skill and background so we are looking for an additional 16 participants to join. You can apply for one or more blocks of minimum two days [see programme below], depending on your interest and availability.

PROGRAMME

Wednesday 7 + Thursday 8 June

The first two days of The Techno-Galactic Software Observatory will be developed in collaboration with the NAM-IP in Namur and will take place in the surrounding of their collection of historical 'numerical artefacts'. Viewing software in this long-term context offers the occasion to reflect on the conditions of its appearance, and allows us to take on current-day questions from a genealogical perspective. What is software? How did it appear as a concept, in what industrial and governmental circumstances? What happens to the material conditions of its production (minerals, factory labor, hardware) when it evaporates into a cloud?

Friday 9 + Saturday 10 June

The second two days will focus on the space-time dimension of IT development. The way computer programs and operating systems are manufactured changed tremendously through time, so its production times and places changed too. From military labs via the mega-corporation cubicles to the open-space freelancer utopia, what ruptures and continuities can be traced? From time-sharing to user-space partitions and containerization, what separations were and are at work? Where and when is software made today?

Sunday 11 + Monday 12 June

The last two days at the Techno-galactic software observatory will be dedicated to observation and its consequences. The development of software encompasses a series of practices whose evocative names are increasingly familiar: feedback, report, probe, audit, inspect, scan, diagnose, explore ... What are the systems of knowledge and power within which these activities take place, and what other types of observation are possible? As a practical set for our investigations, we will together set up a walk-in clinic in the basement of the World Trade Center, where users and developers can arrive on Monday with software-questions of all kinds.

# Technogalactic Software Observation Essentials

**WARNING**

The survival techniques described in the following guide are to be used at your own risk in case of emergency regarding software curiosity. The publisher will not accept any responsability in case of damages caused by misuse, misundestanding of instruction or lack of curiosity. By trying the action exposed in the guide, you accept the responsability of loosing data or altering hardware, including hard disks, usb key, cloud storage, screens by throwing them on the floor, or even when falling on the floor with your laptop by tangling your feet in an entanglement of cables. No arms have been done to human, animal, computers or plants while creating the guide. No firearms or any kind of weapon is needed in order to survive software.
Just a little bit of patience.

**Software observation survival stresses**

**Physical fitness plays a great part of software observation. Be fit or CTRL-Quit.**

When trying to observe software you might experience stresses as such :

- Anxiety
- Sleep deprivation
- Forgetting about eating
- Loss of time tracking

**Can you cope with software ? You have to.**

"*our methods for observation, like mapping, come with their luggage.*"

**See also:** http://observatory.constantvzw.org/etherdump/toc.md.diff.html

# Flow-regulation, logistics, seamlessness

## **Method:** Continuous integration

**Remember:**

**What:**

**HOW:**

**When:**

**Urgency:**

**Note:**

**Warning:**

**See also:**

**Source:**

## **Method:** make make do

**Remember:**

**What:** Makefile as a method for quick/collective assemblages + observing amalgamates/pipelines **HOW:**

**When:**

**Urgency:**

**Note:** Note: http://observatory.constantvzw.org/etherdump/makefile.raw.html etherpad->md->pdf->anything pipeline. makefile as a method for quick/collective assemblages + observing amalgamates/pipelines CHRISTOPH

**Warning:**

**See also:**

**Source:**

# Temporality

## **Method:** Fountain refreshment

**Remember:**

**What:**

**HOW:**

**When:**

**Urgency:**

**Note:**

**Warning:**

The Technogalactic Software Observatory – Comfortable silence, one way mirrors

A drinking fountain and screens of one-way mirrors as part of the work session The Technogalactic Software Observatory organised by Constant. (link)

For the past 100 years the western ideal of a corporate landscape has been has been moving like a pendulum, oscillating between grids of cubicles and organic, open landscapes, in a near to perfect 25-year rhythm. These days the changes in office organisation is supplemented by sound design, in corporate settings mostly to create comfortable silence. Increase the sound and the space becomes more intimate, the person on the table next to you can not immediately hear what you are saying. It seems that actual silence in public and corporate spaces has not been sought after since the start of the 20th century. Actual silence is not at the moment considered comfortable. One of the visible symptoms of our desire to take the edge off the silence is to be observed through the appearance of fountains in public space. The fountains purpose being to give off neutral sound, like white noise without the negative connotations. However as a sound engineer's definition of noise is unwanted sound that all depends on ones personal relation to the sound of dripping water.

This means that there needs to be a consistent inoffensiveness to create comfortable silence.

In corporate architecture the arrival of glass buildings were originally seen as a symbol of transparency, especially loved by governmental buildings. Yet the reflectiveness of this shiny surface once combined with strong light – known as the treason of the glass – was only completely embraced at the invention of one-way-mirror foil. And it was the corporate business-world that would come to be known for their reflective glass skyscrapers. As the foil reacts to light, it appears transparent to someone standing in the dark, while leaving the side with the most light with an opaque surface. Using this foil as room dividers in a room with a changing light, what is hidden or visible will vary throughout the day. So will the need for comfortable silence.

Disclaimer :
Similar to the last 100 years of western office organisation,
this fountain only has two modes:
on or off

If it is on it also offers two options cold water and hot water

This fountain has been tampered with and has not in any way been approved by a proffesional fountain cleaner. I do urge you to consider this before you take the decision to drink from the fountain.

Should you chose to drink from the fountain, then I urge you to write your name on your cup, in the designated area, for a customised experience of my care for you.

I do want you to be comfortable.

http://observatory.constantvzw.org/documents/mia/mia6.gif http://observatory.constantvzw.org/documents/mia
http://observatory.constantvzw.org/documents/mia/IMG_5695.JPG http://observatory.constantvzw.org/docume

**See also:**

**Source:** Mia Melvaer

# Resistance

**Method:** Useless scroll against productivity

**Remember:**

**What:**

**HOW:**

**When:**

**Urgency:**

**Note:**

**Warning:**

**See also:**

**Source:**

### useless scroll against productivity

```
* Source: friday pad, etherpad snippet
```

## **Method:** Interface Dtournement

**Remember:**

**What:**

**HOW:**

**When:**

**Urgency:**

**Note:**

**Warning:**

**See also:**

**Source:**

# Languaging

**Method:** Quine **What:** A program whose function consists of displaying its own code. Also known as ”self-replicating program” **Why:** With quines, the tension between ”software as language” and ”software as operation” is shown in all its density. **HOW:** Self-reflexive coding **Note:** Part of Aquine, a discussion of and research into dualism in software http://pad.constantvzw.org/p/observatory.guide.aquine

Example of a quine (Python). When executed it outputs the same text as the source:

```
s =  's = %r“nprint(s%%s)'
print(s%s)
```

Example of a oneline etherpad quine, created during relearn 2017: ¡pre class=”quaverbatim”¿ wget -qO- http://192.168.73.188:9001/p/quine/export/txt — curl -F ”file=@- ;type=text/plain” http://192.168.73.188:9001/p/quine/import ¡/pre¿

**See also:** http://pad.constantvzw.org/p/observatory.guide.monopsychism **Source:**

**Method:** Glossaries as an exercise

**Remember:**

**What:**

**HOW:**

**When:**

**Urgency:**

**Note:**

**Warning:**

**See also:**

**Source:**

**Method:** Adding qualifiers (secure, bad, bourgeois, queer...)

**Remember:**

**What:** Bringing a moral, ethical, or otherwise evaluative/adjectival/validating lens: ”This morning, Jan had difficulties to answer the question ”what is software”, but he said that he could answer the question ”what is good software”. What is good software?

**HOW:** The more adjectives, the easier the answering? **When:**

**Urgency:**

**Note:** A qualifier like ”good”, ”bad”, ”spy”, ”queer”, ”proletarian”, ”bourgeisoie” can help narrow down definitions.” **Warning:**

**See also:**

**Source:** http://observatory.constantvzw.org/etherdump/multiple-software-axes.html

**Method:** Searching ”software” through software **What:** A quick way to sense the ambiguity of the term 'software', is to go through the manual files on your hard drive and observe in which cases is the term used. **HOW:** command-line oneliner **Why:** : Software is a polymorphic term that take different meanings and comes with different assumptions for the different agents involved in its production, usage and all other forms of encounter and subjection. From the situated point of view of the software present on your machine, when and why does software call itself as such?

```
so software exists only outside your computer? only in general terms?
checking for the word software in all man pages:

grep -nr software /usr/local/man

!!!!

software appears only in terms of license:

This program is free software
This software is copyright (c)
```

```
we don't run software..we still run programs
nevertheless software is everywhere
```

**Source:** [Day1], etherpad snippet, line 574-589 http://observatory.constantvzw.org/etherdump/

**See also:** http://pad.constantvzw.org/p/observatory.guide.samequestion

## **Method:** Persist in calling everyone a Software Curious Person

**Remember:**

**What:** Persistance in naming is a method for changing a person's relationship to software by (forcibly?) persisting in calling everyone a Software Curious Person. **HOW:** Recognising different levels of expertise **When:**

**Urgency:** [overcoming fear of engaging with software] **Note:**

**Warning:**

**See also:**

**Source:**

# Healing

## **Method:** Setup a Relational software observatory consultancy (RSOC)

**Remember:**

- Collectivise research around hacking to save time.
- Self-articulate software needs as your own Operating (system) perspective.
- Change the lens by looking to software through a time perspective.

**What:** By paying a visit to our ethnomethodology interview practice youll learn to observe software from different angles / perspectives. Our practionners passion is to make the what is the relation to software discussion into a service.
**HOW:** Reading the signs. Considering the everchanging nature of software development and use and its vast impact on globalized societies, it is necessary to recognize some of the issues of how software is (often) either passively-perceived or actively-observed, without an articulation of the relations. We offer a method to read the signs of the relational aspect of software observance. It's a crucial aspect of our guide. It will give you another view on software that will shape your ability to survive any kind of software disaster.

**Strategy**: Case study
**Tool**: Qualitative interview

**When:**

**Urgency:**

**Note:**

**Warning:**

What follows is an example of a possible diagnostic questionnaire.

### Sample Questionnaire

**What to expect** You will obtain a cartography of software users profiles. It will help you to shape your own relation to software. You will be able to construct your own taxonomy and classifcation of software users that is needed in order to find a means of rescue in case of a software catastrophy.

- SKILLS

- What kind of user would you say that you are?
- What is your most frequently used type of software?
- How often do you install/experiment/learn new software?
- History
- What is your first recollection of software use?
- How often do / when did you last purchase software or pay for a software service?
- Ethics
- What is the software feature you care about the most?
- Do you use any free software?
  - if yes than
  - do you remember your first attempt at using this software service? Do you still use it? If not why?
- Do you pay for media distribution/streaming services?
- Do you remember your first attempt at using free software and how did that make you feel?
- Have you used any of these software services : facebook, dating app (grindr, tinder, etc.), twitter, instagram or equivalent.
- Can you talk about your favorite apps or webtools that you use regularly?
- What is most popular software your friends use ?
- SKILL
- Would you say that you are a specilised user?
- Have you ever used the command line?
- Do you know about scripting?
- Have you ever edited an HTML page? A CSS file? A PHP file? A configuration file?
- Can you talk about your most technical encounter with your computer / telephone ?
- ECONOMY

- How do you pay for your software use ?
  - Please elaborate (for example, do you buy the software? / contribute in kind / deliver services or support)
  - What is the last software that you paid for using ?
  - What online services are you currently paying for ?
  - Is someone paying for your use of service ?
- Personal

## Sample questionnaire results

## Possible/anticipated user profiles

**. . . meAsHardwareOwnerSoftwareUSER:**

“I did not own a computer personally until very very late as I did not enjoy gaming as a kid or had interest in spending much time behind PC beyond work (and work computer). My first was hence I think in 2005 and it was a SGI workstation that was the computer of the year 2000 (cost 10.000USD) and I got it for around 300USD. Proprietary drivers for unified graphics+RAM were never released, so it remained a software dead-end in gorgeous blue curved chassis http://www.sgidepot.co.uk/sgidepot/pics/vwdocs.jpg”

**. . . meAsSoftwareCONSUMER:**

“I payed/purchased software only twice in my life (totalling less then 25eur), as I could access most commercial software as widely pirated in Balkans and later had more passion for FLOSS anyway, this made me relate to software as material to exchange and work it, rather than commodity goods I could or not afford.”

**. . . meAsSoftwareINVESTOR:**

“I did it as both of those apps were niche products in early beta (one was Jeeper Elvis, real-time-non-linear-video-editor for BeOS) that failed to reach market, but I think I would likely do it again and only in that mode (supporting the bleeding edge and off-stream work), but maybe with more than 25eur.”

**. . . meAsSoftwareUserOfOS:**

“I would spend most of 80s ignoring computers, 90ties figuring out software from high-end to low-end, starting with OSF/DecAlpha and SunOS, than IRIX and MacOS, finally Win 95/98 SE, that permanently pushed me into niches (of montly LINUX distro install fests, or even QNX/Solaris experiments and finally BeOS use).”

**. . . meAsSoftwareWEBSURFER:**

“I got used to websurfing in more than 15 windows on UNIX systems and never got used to less than that ever since, furthermore with addition of more browser options this number only multiplied (always wondered if my first system was Windows 3.11 - would I be a more focused person and how would that form my relations to browser windows>tabs).”

**. . . meAsSoftwareUserOfPropertarySoftware:**

”I signed one NDA contract in person on the paper and with ink on a rainy day while stopping of at trainstaion in north Germany for the software that was later to be pulled out of market due to problematic licencing agreement (intuitivly I knew it was wrong) - it had too much unprofessional pixeleted edges in its graphics.

**. . . meAsSoftwareUserOfDatingWebsites:**

“I got one feature request implemented by a prominent dating website (to search profiles by language they speak), however I was never publicly acknowledged (though I tried to make use of it few times), that made our relations feel a bit exploitative and underappreciated.”

**. . . meAsSoftwareUserTryingToGoPRO:**

“my only two attempts to get into the software company failed as they insisted on full time commitments. Later I found out ones were intimidated in interview and other gave it to a person that negotiated to work part time with friend! My relation to professionalism is likely equally complex and pervert as one to the software.”

### Case study : W. W.

. . . ww.AsExperiencedAdventerousUSER - experiments with software every two days as she uses FLOSS and Gnu/Linux, cares the most for maliabity of the software - as a result she has big expectations of flexibility even in software category which is quite conventional and stability focused like file-hosting.

. . . ww.AsAnInevstorInSoftware - paid compiled version of FLOSS audio software 5 years ago as she is supportive of economy and work around production, maintainance and support, but she also used closed hardware/software where she had to agree on licences she finds unfair, but then she was hacking it in order to use it as an expert - when she had time.

. . . ww.AsCommunicationSoftwareUSER - she is not using commercial social networks, so she is very concious of information transfers and time relations, but has no strong media/format/design focus.

Q: What is your first recollection of software use? A: ms dos in 1990 at school _ i was 15 or 16. oh no 12. Basic in 1986.

Q: What are the emotions related to this use? A: fun. i'm good at this. empowering

Q: How often do / when did you last purchase software or pay for a software service? A: I paid for ardour five years ago. I paid the developper directly. For the compiled version. I paid for the service. I pay for my website and email service at domaine public.

Q: What kind of user would you say you are? A: An experienced user drawing out the line. I don't behave.

Q: Is there a link between this and your issue A: Even if it's been F/LOSS there is a lot of decision power in my package.

Q: What is your most frequently used type of software? A: Web browser. email. firefox & thunderbird

Q: How often do you install/experiment/learn new software? A: Every two days. I reinstall all the time. my old lts system died. stop being supported last april. It was linux mint something.

Q: Do you know about scripting ? A: I do automating scripts for any operation i have to doi several times like format conversion.

Q: Can you talk about your most technical encounter with your computer / telephone ? A: I've tried to root it. but i didn't succeed.

Q: How much time do you wish to spend on such activities like hacking, rooting your device? A: hours. you should take your time

Q: Did you ever sign licence agreement you were not agree with? how does that affect you. A: This is the first thing your when you have a phone. it's obey or die.

Q: What is the software feature you care for the most ? A: malleability. different ways to approach a problem, a challenge, an issue.

Q: Do you use any free software ? A: yes. there maybe are some proprietary drivers.

Q: Do you remember your first attempt at using free software and how did that make you feel? A: Yes i installed my dual boot in . . . 10 years ago. scared and powerful.

Q: Do you use one of this software service : facebook, dating app (grindr of sort), twitter, instagram or equivalent. A: Google, gmail that's it

Q: Can you talk about your favorite apps or webtools that you use regularly? A: music player. vanilla music and f-droid. browser. i pay attention to clearing my history, no cookies. I also have iceweasel. Https by default. Even though i have nothing to hide.

Q: What stories around contracts and administration in relation to your software internet or computer ? A: Nothing comes to my mind. i'm not allowed to do, to install on phone. When it's an old phone, there is nothing left that is working you have to do it.

Q: How does software help you shape your relations with other people ? A: It's a hard question. if it's communication software of course it's it's nature to be related to other people.there is an expectency of immediate reply, of information transfer. . . It's troubling your relation with people in certain situations.

Q: From which countries does your softwares live / is coming from? How do you feel about that? A: i think i chose the netherlands as a miror. you are hoping to reflect well in this miror.

Q: Have you ever read a terms of software service; one that is not targeting the American market? A: i have read them. no.

**See also:**

**Source:**

http://observatory.constantvzw.org/etherdump/SCP.see.html http://observatory.constantvzw.org/etherdump/SC
http://observatory.constantvzw.org/etherdump/SCP.wendy.html This method was developed by The RSOC Group.

## **Method:** Agile Sun Salutation

**Remember:**

> *Agile software development describes a set of values and principles for software development under which requirements and solutions evolve through the collaborative effort of self-organizing cross-functional teams. It advocates adaptive planning, evolutionary development, early delivery, and continuous improvement, and it encourages rapid and flexible response to change. These principles support the definition and continuing evolution of many software development methods. Source: https://en.wikipedia.org/wiki/Agile_software_development*

**What:** You will be observing yourself **HOW:**

**When:** Anywhere where its possible to lie on the floor **Urgency:** Using Agile software development methods to develop a new path into your professional and personal life towards creativity, focus and health. **Note:**

**Warning:**

Hello and welcome to the presentation of the agile yoga methodology. I am Allegra, and today I'm going to be your personal guide to YOGA, an acronym for why organize? Go agile! I'll be part of your team today and we'll do a few exercises together as an introduction to a new path into your professional and personal life towards creativity, focus and health.

A few months ago, I was stressed, overwhelmed with my work, feeling alone, inadequate, but since I started practicing agile yoga, I feel more productive. I have many clients as an agile yoga coach, and I've seen new creative business opportunities coming to me as a software developer.

For this first experience with the agile yoga method and before we do physical exercises together, I would like to invite you to close your eyes. Make yourself comfortable, lying on the floor, or sitting with your back on the wall. Close your eyes, relax. Get comfortable. Feel the weight of your body on the floor or on the wall. Relax.

Leave your troubles at the door. Right now, you are not procrastinating, you are having a meeting at the <SAY THE NAME OF YOUR LOCATION HERE>, a professional building dedicated to business, you are meeting yourself, you are your own business partner, you are one. You are building your future.

You are in a room standing with your team, a group of lean programmers. You are watching a white board together. You are starting your day, a very productive day as you are preparing to run a sprint together. Now you turn towards each other, making a scrum with your team, you breathe together, slowly, inhaling and exhaling together, slowly, feeling the air in and out of your body. Now you all turn towards the sun to prepare to do your ASSanas, the agile Sun Salutations or ASS with the team dedicated ASS Master. She's guiding you. You start with Namaskar, the Salute. your palms joined together, in prayer pose. you all reflect on the first principle of the agile manifesto. your highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Next pose, is Ardha Chandrasana or (Half Moon Pose). With a deep inhalation, you raise both arms above your head and tilt slightly backward arching your back. you welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage. then you all do Padangusthasana (Hand to Foot Pose). With a deep exhalation, you bend forward and touch the mat, both palms in line with your feet, forehead touching your knees. you deliver working software frequently.

Surya Darshan (Sun Sight Pose). With a deep inhalation, you take your right leg away from your body, in a big backward step. Both your hands are firmly planted on your mat, your left foot between your hands. you work daily throughout the project, business people and developers together. now, you're flowing into Purvottanasana (Inclined Plane) with a deep inhalation by taking your right leg away from your body, in a big backward step. Both your hands are firmly planted on your mat, your left foot between your hands. you build projects around motivated individuals. you give them the environment and support they need, and you trust them to get the job done.

You're in Adho Mukha Svanasana (Downward Facing Dog Pose). With a deep exhalation, you shove your hips and butt up towards the ceiling, forming an upward arch. Your arms are straight and aligned with your head. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Then, Sashtang Dandawat (Forehead, Chest, Knee to Floor Pose). With a deep exhalation, you lower your body down till your forehead, chest, knees, hands and feet are touching the mat, your butt tilted up. Working software is the primary measure of progress.

Next is Bhujangasana (Cobra Pose). With a deep inhalation, you slowly snake forward till your head is up, your back arched concave, as much as possible. Agile processes promote sustainable development. You are all maintaining a constant pace indefinitely, sponsors, developers, and users together.

Now back into Adho Mukha Svanasana (Downward Facing Dog Pose). Continuous attention to technical excellence and good design enhances agility.

And then again to Surya Darshan (Sun Sight Pose). Simplicity–the art of maximizing the amount of work not done–is essential. Then to Padangusthasana (Hand to Foot Pose). The best architectures, requirements, and designs emerge from self-organizing teams.

You all do again Ardha Chandrasana (Half Moon Pose). At regular intervals, you as the team reflect on how to become more effective, then tune and adjust your behavior accordingly. you end our ASSanas session with a salute to honor your agile yoga practices. you have just had a productive scrum meeting. now i invite you to open your eyes, move your body around a bit, from the feet up to the head and back again.

Stand up on your feet and let's do a scrum together if you're ok being touched on the arms by someone else. if not, you can do it on your own. so put your hands on the shoulder of the SCP around you. now we're joined together, let's look at the screen together as we inhale and exhale. syncing our body together to the rythms of our own internal software, modulating our oxygen level intake requirements to the oxygen availability of our service facilities.

Now, let's do together a couple of exercise to protect and strengthen our wrists. as programmers, as internauts, as entrepreneurs, they are a very crucial parts of the body to protect. in order to be able to type, to swipe, to shake hands vigourously, we need them in good health. So bring to hands towards each other in a prayer pose, around a book, a brick. You can do it without but I'm using my extreme programming book - embrace change - for that. So press the palms together firmly, press the pad of your fingers together. do that while breathing in and out twice.

Now let's expand our arms towards us, in the air, face and fingers facing down. like we're typing. make your shoulders round. let's breath while visualizing in our heads the first agile mantra : Individuals and interactions over processes and tools.

Now let's bring back the arms next to the body and raise them again. And let's move our hands towards the ceiling this time. Strenghtening our back. In our head, the second mantra. Working software over comprehensive documentation. now let's bring back the hands in the standing position. Then again the first movement while visualizing the third mantra : Customer collaboration over contract negotiation and then the second movement thinking about the fourth and last mantra : Responding to change over following a plan and of course we continue breathing. Now to finish this session, let's do a sprint together in the corridor !

**See also:**

**Source:** Developed by: Anne Laforet, performed by: Allegra

**Method:** Hand reading **HOW:** Visit the Future Blobservation Booth to have your fortunes read and derive life insight from the wisdom of software. **What:** put your hand in the reading booth and get your line read. **Why:** The hand which holds your mouse everyday hides many secrets.

```
 sample reading timeline
a test user, all tests clear and systems are online a user who said
 15:35  another nice user
  15:40 another nice user
 15:47 happy user (laughing)
15:51 user complaining about her fortune, saying it's not true. Four
15:59 another nice user:  http://etherbox.local:9001/p/SCP.sedyst.
 16:06 a polite user
 16:08 a  friendly playful user (stephanie)
 16:12 a very giggly user (wendy)
 16:14  a playful user - found the reading process erotic - DEFRAGME
 16:19 a  curious user
16:27 a friendly user but oh no, we had a glitch and  computer crash
16:40 a nice user, the printer jammed but it was sorted out  quickly
```

```
16:42 another nice user
16:50 nice user (joak)
16:52 yet another  nice user (jogi)
16:55 happy user! (peter w)
16:57 more happy user  (pierre h)
16:58 another happy user
17:00 super happy user (peggy)
17:02  more happy user
```

**Source:**

**See also:**

## **Method:** Bug reporting for sharing observations

**Remember:**

**What:** Etherpad had stopped working but it was unclear why. Where does etherpad 'live'? **HOW:** Started by looking around the pi's filesystem by reading /var/log/syslog in /opt/etherpad and in a subdirectory named var/ there was dirty.db, and dirty it was. **When:** Monday morning **Urgency:** Software (etherpad) not working and the Walk-in Clinic was about to start. **Note:** http://pad.constantvzw.org/p/observatory.inver

**Warning:**

from jogi@mur.at to [Observatory] When dirty.db get's dirty

Dear all,

as promised yesterday, here my little report regarding the broken etherpad.

<md> ### When dirty.db get's dirty

When I got to WTC on Monday morning the etherpad on etherbox.local was disfunct. Later someone said that in fact etherpad had stopped working the evening before, but it was unclear why. So I started looking around the pi's filesystem to find out what was wrong. Took me a while to find the relevant lines in /var/log/syslog but it became clear that there was a problem with the database. Which database? Where does etherpad 'live'? I found it in /opt/etherpad and in a subdirectory named var/ there it was: dirty.db, and dirty it was.

A first look at the file revealed no apparent problem. The last lines looked like this:

```
–"key":"sessionstorage:Ddy0gw7okwbkv5BzkR1DuSLCV˙IA5˙jQ","val":–"coo
":–"path":"/","˙expires":null,"originalMaxAge":null,"httpOnly":true,
–"key":"sessionstorage:AU1cffgcTf˙q6BV9aIdAvES2YyXM7Gm1","val":–"coo
":–"path":"/","˙expires":null,"originalMaxAge":null,"httpOnly":true,
–"key":"sessionstorage:˙H5SdUlDvQ3XCuPaZEXQ5lx0K6aAEJ9m","val":–"coo
":–"path":"/","˙expires":null,"originalMaxAge":null,"httpOnly":true,
cure":false˝˝˝
```

What I did not see at the time was that there were some (AFAIR something around 150) binary zeroes at the end of the file. I used tail for the first look and that tool silently ignored the zeroes at the end of the file. It was Martino who suggested using different tools (xxd in that case) and that showed the cause of the problem. The file looked something like this:

```
00013730: 6f6b 6965 223a 7b22 7061 7468 223a 222f  okie":–"path":"/
00013740: 222c 225f 6578 7069 7265 7322 3a6e 756c  ","˙expires":nul
00013750: 6c2c 226f 7269 6769 6e61 6c4d 6178 4167  l,"originalMaxAg
00013760: 6522 3a6e 756c 6c2c 2268 7474 704f 6e6c  e":null,"httpOnl
00013770: 7922 3a74 7275 652c 2273 6563 7572 6522  y":true,"secure"
00013780: 3a66 616c 7365 7d7d 7d0a 0000 0000 0000  :false˝˝˝.......
00013790: 0000 0000 0000 0000 0000 0000 0000 0000  ................
```

So Anita, Martino and I stuck our heads together to come up with a solution. Our first attempt to fix the problem went something like this:

```
dd if=dirty.db of=dirty.db.clean bs=1 count=793080162
```

which means: write the first 793080162 blocks of size 1 byte to a new file. After half an hour or so I checked on the size of the new file and saw that some 10% of the copying had been done. No way this would get done in time for the walk-in-clinic. Back to the drawing board.

Using a text editor was no real option btw since even vim has a hard time with binary zeroes and the file was really big. But there was hexedit! Martino installed it and copied dirty.db onto his computer. After some getting used to the various commands to navigate in hexedit the unwanted zeroes were gone in an instant. The end of the file looked like this now:

```
00013730: 6f6b 6965 223a 7b22 7061 7468 223a 222f  okie":{"path":"/
00013740: 222c 225f 6578 7069 7265 7322 3a6e 756c  ","_expires":nul
00013750: 6c2c 226f 7269 6769 6e61 6c4d 6178 4167  l,"originalMaxAg
00013760: 6522 3a6e 756c 6c2c 2268 7474 704f 6e6c  e":null,"httpOnl
00013770: 7922 3a74 7275 652c 2273 6563 7572 6522  y":true,"secure"
00013780: 3a66 616c 7365 7d7d 7d0a                 :false}}}.
```

Martino asked about the trailing '.' character and I checked a different copy of the file. No '.' there, so that had to go too. My biggest mistake in a long time! The '.' we were seeing in Martino's copy of the file was in fact a '' (0a)! We did not realize that, copied the file back to etherbox.local and waited for etherpad to resume it's work. But no luck there, for obvious reasons.

We ended up making backups of dirty.db in various stages of deformation and Martino started a brandnew pad so we could use pads for the walk- in-clinic. The processing tool chain has been disabled btw. We did not want to mess up any of the already generated .pdf, .html and .md files.

We still don't know why exactly etherpad stopped working sometime Sunday evening or how the zeroes got into the file dirty.db. Anita thought that she caused the error when she adjusted time on etherbox.local, but the logfile does not reflect that. The last clean entry in /var/log/syslog regarding nodejs/etherpad is recorded with a timestamp of something along the line of 'Jun 10 10:17'. Some minutes later, around 'Jun 10 10:27' the first error appears. These timestamps reflect the etherbox's understanding of time btw, not 'real time'.

It might be that the file just got too big for etherpad to handle it. The size of the repaired dirty.db file was already 757MB. That could btw explain why etherpad was working somewhat slugishly after some days. There is still a chance that the time adjustment had an unwanted side effect, but so far there is no obvious reason for what had happened. </md> – J.Hofmller

```
http://thesix.mur.at/
```

**See also:**

**Source:** jogi, http://pad.constantvzw.org/p/observatory.inventory.jogi

# Close encounters

**Method:** Encounter several collections of historical hardware back-to-back

**Remember:**

**What:**

**HOW:**

This can be done by identifying one or more computer museums and visit them with little time in-between. Visiting a friend with a large basement and lots of left-over computer equipment can help. Seeing and possibly touching hardware from different contexts (state-administration, business, research, . . . ), periods of time, cultural contexts (California, Germany, French-speaking Belgium) and price ranges allows you to sense the interactions between hardware and software development. xxxx A way to hear people speak about the objects

**When:**

**Urgency:**

**Note:**

**Warning:**

**See also:**

**Source:**

**Method:** Interview people about their histories with software

**Remember:**

**What:** Observe personnal narratives around software history. Retrace the path of relation to software, how it changed during the years and what are the human access memories that surrounds it. To look at software through personal relations and emotions. **HOW:**

**When:**

**Urgency:**

**Note:**

**Warning:**

Jean Heuns has been collecting servers, calculators, softwares, magnetic tapes hard disks for xxx years. Found an agreement for them to be displayed in the department hallways. Department of Computer sciences - Kul Leuven.

[interview transcription goes here]

http://gallery.constantvzw.org/var/albums/Techno-Galactic-Software-Observatory/PWFU3350.JPG
http://gallery.constantvzw.org/var/albums/Techno-Galactic-Software-Observatory/PWFU3361.JPG
http://gallery.constantvzw.org/var/albums/Techno-Galactic-Software-Observatory/PWFU3356.JPG
http://gallery.constantvzw.org/var/albums/Techno-Galactic-Software-Observatory/PWFU3343.JPG

**See also:**

**Source:**

**Method:** Ask several people from different fields and age-groups the same question: "$_{*}*Whatissoftware?**$"

**Remember:**

**What:** By paying close attention to the answers, and possibly logging them, observations on the ambiguous place and nature of software can be made.
**HOW:**

**When:**

**Urgency:**

**Note:**

**Warning:**

Example: Jan Huens (xxxx): "It is difficult to answer the question 'what is software', but I know what is good software" (see also [LINK TO INTERVIEW] Thomas Cnudde (xxxx): "*Software is a list of sequential instructions!*". "*Hardware for me is made of silicon, software a sequence of bits in a file. But naturally I am biased: I'm a hardware desginer so I like to consider it as unique and special*". Amal . . . (Guide from Day 1 in Namur) "You have to ask the specialists."

**See also:**

**Source:**

### **Method:** Pan/Monopsychism

**Remember:** aquinas famously opposed averroes..who's philosophy can be interpreted as monopsychist :)

**What:** reading and writing sectors of memory of/to different computers **HOW:** bash commands. **When:**

**Urgency:** to challenge the process/file divide. to have a more intimate relation with your and others computer **Note:**

**Warning:** may damage your ram

**See also:**

**Source:** etherpad snippets, line

### **Method:** FMEM and /DEV/MEM

**Remember:** In the early days, the easiest was to dump the memory from the memory device (/dev/mem) but over time the access was more and more restricted in order to avoid malicious process to directly access the kernel memory directly. The kernel option CONFIG$_STRICT_DEVMEMwasintroducedinkernelversion HEAD).Soyou'llneedtouseaLinuxkernelmoduleinordertoacquirememory.$

**What:** Different ways of exploring your memory (RAM). Because in unix everything is a file, you can access your memory as if it were a file. **HOW:**

**When:**

**Urgency:** To try and observe the operational level of software, getting closer to the workings, the instruction-being of an executable/executing file, the way it is when it is loaded into the memory rather than when it sits in the harddisk **Note:** Part of Aquine, a discussion of and research into dualism in software http://pad.constantvzw.org/p/observatory.guide.aquine

**Warning:**

fmem 1.5.0 This module creates /dev/fmem device, that can be used for dumping physical memory, without limits of /dev/mem (1MB/1GB, depending on distribution) How:

to find read/write memory addresses of a certain process awk -F "-|" '$3 ~ /rw/ { print $1 " " 2′ $/proc/PID/mapstaketherangeanddropittohexdumpsudoddif = /dev/membs = 1skip =$(( 16#b7526000 - 1 )) count=$(( 16#b7528000 - 16#7b7526000 + 1)) | hexdump -C

todo: list commands to explore memory. to find specific process.to dump.to visualize it etc.

**See also:** http://pad.constantvzw.org/p/observatory.guide.monopsychism

**Source:**

# Embodiment / body techniques

**Method:** Comportment of software (occupational hazards)

**Remember:**

**What:** Observing ways software produces bodies **HOW:**

**When:**

**Urgency:**

**Note:**

**Warning:**

**See also:**

**Source:**

# Collections / collecting

## **Method:** Compiling a bestiary of software logos

**Remember:**

**What:** Since the early days of GNU-linux and cemented through the ubiquitous O'Reilly publications, the visual culture of software relies heavily on animal representations. But what kinds of animals, and to what effect? **HOW:**

Compile a collection of logos and note the metaphors for observation: - stethoscope - magnifying glass - long neck (giraffe)

**When:**

**Urgency:**

**Note:**

**Warning:**

[check Testing the testbed pads for examples] [something on bestiaries]

**See also:**

**Source:**

## **Method:** Interview people about their histories with software

**Remember:**

**What:** Observe personnal narratives around software history. Retrace the path of relation to software, how it changed during the years and what are the human access memories that surrounds it. To look at software through personal relations and emotions. **HOW:**

**When:**

**Urgency:**

**Note:**

**Warning:**

Jean Heuns has been collecting servers, calculators, softwares, magnetic tapes hard disks for xxx years. Found an agreement for them to be displayed in the department hallways. Department of Computer sciences - Kul Leuven.

[interview transcription goes here]

http://gallery.constantvzw.org/var/albums/Techno-Galactic-Software-Observatory/PWFU3350.
http://gallery.constantvzw.org/var/albums/Techno-Galactic-Software-Observatory/PWFU3361.
http://gallery.constantvzw.org/var/albums/Techno-Galactic-Software-Observatory/PWFU3356.
http://gallery.constantvzw.org/var/albums/Techno-Galactic-Software-Observatory/PWFU3343.

**See also:**

**Source:**

**Method:** Testing the testbed: testing software with observatory ambitions (SWOA) **Warning:** this method may make more sense if you first take a look at the [Something in the Middle Maybe (SitMM)](http://pad.constantvzw.org/p/observatory.guide.s which is an instance of a SWOA **HOW:** The interwebs hosts many projects that aim to produce software for observing software, (from now on Software With Observatory Ambitions (SWOA)). A comparative methodology can be produced by testing different SWOA to observe software of interest. Example: use different sniffing software to observe wireless networks, e.g., wireshark vs tcpdump vs SitMM. Comparing SWOA reveals what is seen as worthy of observation (e.g., what protocols, what space, which devices), the granularity of the observation (e.g., how is the observation captured, in what detail), the logo and conceptual framework of choice etc. This type of observation may be turned into a service (See also: Something in the Middle Maybe (SitMM)). **When:** Ideally, SWOA can be used everywhere and in every situation. In reality, institutions, laws and administrators like to limit the use of SWOA on infrastructures to people who are also administering these networks. Hence, we are presented with the situation that the use of SWOA is condoned when it is down by researchers and pen testers (e.g., they were hired) and shunned when done by others (often subject to name calling as hackers or attackers). **What:** Deep philosophical moment: most software has a recursive observatory ambition (it wants to be observed in its execution, output etc.). Debuggers, logs, dashboards are all instances of software with observatory ambitions and can not be separated from software itself. Continuous integration is the act of folding

the whole software development process into one big feedback loop. So, what separates SWOA from software itself? Is it the intention of observing software with a critical, agonistic or adversarial perspective vs one focused on productivity and efficiency that distinguishes SWOA from software? What makes SWOA a critical practice over other forms of sotware observation. If our methodology is testing SWOA, then is it a meta critique of critique? **Urgency:** If observation is a form of critical engagement in that it surfaces the workings of software that are invisible to many, it follows that people would develop software to observe (SWOAs). Testing SWOAs puts this form of critical observation to test with the desire to understand how what is made transparent through each SWOA also makes things invisible and reconfigures power. **Note:** Good SWOA software usually uses an animal as a logo.:D

**Warning:** Many of the SWOA projects we looked at are promises more than running software/available code. Much of it is likely to turn into obsolete gradware, making testing difficult. **Remember:**

**Source:** By/history/source/origin/process: the ”original testbed” was proposed by researchers/collaborators at Princeton University. Testing this testbed at a local Constant workshop led to the meta-project on testing software meant for testing software.

**See also:** making a bestiary of visual cultures/logos around it **See also:** http://pad.constantvzw.org/p/observato

**See also:** http://pad.constantvzw.org/p/observatory.guide.sitmm

## **Method:** Prepare a reader to think theory with software

**Remember:**

**What:** Software observations are mostly done in the realm of the technological and the pragmatic. Manuals, technical documentation. Software studies vs. software as a critique. Ways of speaking/writing about. **HOW:**

**When:**

**Urgency:**

**Note:**

**Warning:**

Pull some quotes from the reader, for example: Observation and its consequences

- Lilly Irani, Hackathons and the Making of Entrepreneurial Citizenship, 2015 http://sci-hub.bz/10.1177/0162243915578486
- Kara Pernice (Nielsen Norman Group), Talking with Participants During a Usability Test, January 26, 2014, https://www.nngroup.com/articles/talking-to-users/
- Matthew G. Kirschenbaum, Extreme Inscription: Towards a Grammatology of the Hard Drive. 2004 http://texttechnology.mcmaster.ca/pdf/vol13_2_06.pdf
- Alexander R. Galloway, The Poverty of Philosophy: Realism and Post-Fordism, Critical Inquiry. 2013, http://cultureandcommunication.org/galloway/pdf/Galloway,%
- Edward Alcosser, James P. Phillips, Allen M. Wolk, How to Build a Working Digital Computer. Hayden Book Company, 1968. https://archive.org/details/howtobuildawork
- Matthew Fuller, “It looks like you’re writing a letter: Microsoft Word”, Nettime, 5 Sep 2000. https://library.memoryoftheworld.org/b/xpDrXE_VQeeuDDpc5RrywyT
- Barbara P. Aichinger, DDR Memory Errors Caused by Row Hammer. 2015 www.memcon.com/pdfs/proceedings2015/SAT104_FuturePlus.pdf
- Fangfei Liu, Yuval Yarom, Qian Ge, Gernot Heiser, Ruby B. Lee. Last-Level Cache Side-Channel Attacks are Practical. 2015 http://palms.ee.princeton.edu/system

**See also:** http://pad.constantvzw.org/p/observatory.guide.samequestion

**Source:** http://pad.constantvzw.org/p/observatory.reader

# Beingontheside/inthemiddle/behind

## **Method:** Something in the Middle Maybe (SitMM)

**Remember:**

**What:** The network traffic gets observed. There are different sniffing software out there which differ in granularity and how far the user can taylor the different functionality. SitMM builds on one of these tools called [scapy](http://www.secdev.org/projects/scapy/). **HOW:** SitMM takes a closer look at the network traffic coming from/going to a software curious person's device. The software curious person using SitMM may ask to filter the traffic based on application or device of interest. **When:**

The software curious person gets to observe their own traffic. Ideally, observing ones own network traffic should be available to anyone, but using such software can be deemed illegal under different jurisdictions.

For example, in the US wiretap law limit packet-sniffing to parties owning the network that is being sniffed or the availability of consent from one of the communicating parties. Section 18 U.S. Code 2511 (2) (a) (i) says: 1 See here for a paper [2] on the topic. Google went on a big legal spree to defend their right to capture unencrypted wireless traffic with google street view cars. The courts were concerned about wiretapping and infringements on the privacy of users, and not with the leveraging of private and public WiFi infrastructure for the gain of a for profit company. The case raises hard questions about the state, ownership claims and material reality of WiFi signals. So, while WiFi sniffing is common and the tools like SitMM are widely available, it is not always possible for software curious persons to use them legally or to neatly filter out "their traffic" from that of "others".

- When: SitMM can be used any time a software curious person feels the weight of the (invisible) networks.
- Why: SitMM is intended to be a tool that gives artists, designers and educators an easy to use custom WiFi router to work with networks and explore the aspects of our daily communications that are exposed when we use WiFi. The goal is to use the output to encourage open discussions about how we use our devices online.

---

[2] http://spot.colorado.edu/ sicker/publications/issues.pdf

**Urgency:**

**Note:**

**Warning:**

Snippets of a Something In The Middle, Maybe - Report “‘ UDP 192.168.42.32:53649 -> 8.8.8.8:53 TCP 192.168.42.32:49250 -> 17.253.53.208:80 TCP 192.168.42.32:49250 -> 17.253.53.208:80 TCP/HTTP 17.253.53.208:80 GET http://captive.apple.com/mDQArB9orE TCP 192.168.42.32:49250 -> 17.253.53.208:80 TCP 192.168.42.32:49250 -> 17.253.53.208:80 TCP 192.168.42.32:49250 -> 17.253.53.208:80 UDP 192.168.42.32:63872 -> 8.8.8.8:53 UDP 192.168.42.32:61346 -> 8.8.8.8:53 . . . TCP 192.168.42.32:49260 -> 17.134.127.97:443 TCP 192.168.42.32:49260 -> 17.134.127.97:443 TCP 192.168.42.32:49260 -> 17.134.127.97:443 TCP 192.168.42.32:49260 -> 17.134.127.97:443 TCP 192.168.42.32:49260 -> 17.134.127.97:443 TCP 192.168.42.32:49260 -> 17.134.127.97:443 TCP 192.168.42.32:49260 -> 17.134.127.97:443

Destination Address: 17.253.53.208 Destination Name: nlams2-vip-bx-008.aaplimg.com

Port: Connection Count 80: 6

Destination Address: 17.134.127.79 Destination Name: unknown

Port: Connection Count 443: 2 ##################################################
Destination Address: 17.248.145.76 Destination Name: unknown

Port: Connection Count 443: 16

**See also:**

**Source:** *Sitm* emerges from the collective practice of the Alternative Learning Tank and is heavily inspired by projects such as [Dowse](http://dowse.equipment/), [alt.exit](http://alternativelearningtank.net/) and the [NetAidKit](https://netaidkit.net/). http://observatory.constantvzw.org/SomethingInTheMiddle/

## **Method:** What is it like to be AN ELEVATOR?

**Remember:**

**What:** Understanding software systems by becoming them **HOW:** Creating a flowchart to incarnate a software system you use everyday **When:** An elevator, for example. **Urgency:**

**Note:**

**Warning:** Uninformed members of the public may panic when confronted with a software performance in a closed space.

For example:

what
is
it
like
to
be
an
elevator?

“from 25th floor to 1st floor”

light on button light of 25th floor

check current floor

if current floor is 25th floor

no

if current floor is . . .

go one floor up

. . . smaller than 25th floor

go one floor down

. . . bigger than 25th floor

stop elevator

turn button light off of 25th floor

turn door light on

open door of elevator

play sound opening sequence

yes

start

user pressed button of 25th floor

close door of elevator

if door is closed

user pressed 1st floor button

start timer for door closing

if timer is running more than three seconds

yes

yes

light on button

go one floor down

no

if current floor is 1st floor

update floor indicator

check current floor

stop elevator

no

yes

light off button

turn door light on

open door of elevator

play sound opening sequence

end

update floor indicator

**See also:**

**Source:** Developed by Joseph Knierzinger

## **Method:** Side Channel Analysis

**Remember:**

**What:**

**HOW:**

**When:**

**Urgency:** Side Channel attacks are possible by disregarding the abstraction of software into pure logic: the physical effects of the running of the software become backdoors to observe its functioning, both threatening the control of processes and the re-affirming the materiality of software. **Note:**

**Warning:** ** engineers are good guys! **

https://www.tek.com/sites/default/files/media/image/119-4146-00%20Near%20Field%20Probe%20Set.png.jpg

**See also:**

**Source:**

## Methods for inadequate observation of software [inadequate methods?]

but then the things in these groupings can be arranged/divided by form? like descrptive entries/ exercises/ diagrams/ conversation with/ gloss/

==========