Contributors: Tibi, Emily McGill, Chris Troutner, …

OVN proto-manual

How to start an OVN

Under construction…

If you can't wait, contact Tibi

NOTE: This doc may be outdated. Please use Tibi's fork, as the owner, Keli Yes, has been inactive within the Sensorica / OVN community or network

---

Table of contents

Contributors: Tibi, Emily McGill, Chris Troutner, …

Contributors: Tibi, Emily McGill, Chris Troutner, …

Contributors: Tibi, Emily McGill, Chris Troutner, …

Other links

Sensorica's Collaborative Entrepreneurship Course
Readme OVN manual
OVN metamaps
Help docs with the NRP

What is an OVN?

OVN stands for *open value network*. It is an organizational model adapted for **open collaboration**, **swarm-type projects** or crowd-based processes. OVNs are stigmergic environments and try to maximize the expression of collective intelligence. Its model is an assembly of governance, methodologies of work, and tools (IT infrastructure and others), with prescriptions about a work culture. One instantiation of the OVN model is www.sensorica.co. See more on the OVN wiki. See also types of OVNs.

What an OVN **isn't**

- A traditional legal form, such as a coop, an inc, a nonprofit, etc.
- An open community
- An intentional community

Why create an OVN?

An organization type.is a vehicle for action, it allows you to do something with others. Before you choose this organizational model please sit down and consider a few questions.

- Why would you create an OVN instead of another type of organization?
- How to know if the OVN model is the right vehicle for you, to reach your goals?
- What does the OVN model provide that other models don't?
- What should be in place to create an OVN?
- Add others...

There are many reasons why you would choose the OVN model for your organisation.

- **Reduce your dependency on the financial system to launch a project**

- ➢ The OVN model is tuned for **crowdsourcing**: in every process, agents can contribute with any type of resources that are needed. We find here the pattern of the so-called *sharing economy*; sharing resources and excess capacity in the context of a collaboration, project or joint-venture. In other words, the OVN model is designed to do more with less (money), thus avoiding trips to the bank.

❖ **Accelerate innovation** and **tackle very complex problems** that require transdisciplinary collaboration

- ➢ The OVN model prescribes a *transparent* and *open* network-type organization (where *transparent* means access to information and *open* means access to participation). An example of such a transparent and open network model is Wikipedia, where people who edit pages do not need permission to do so. The vast knowledge domain covered by Wikipedia is the living proof of the ability of OVNs to deal with complexity.

❖ **Build synergy, complementarity and compatibility among a large and diverse base of stakeholders** (protocols, standards, ...)

- ➢ OVNs are designed as highly collaborative organizations. Competition is pushed towards the use of scarce resources, and it is eliminated among agents who collaborate within projects to achieve the best outcome. If there is a disagreement the project can fork, the development splits into two parallel branches, which can continue to influence each other and can even merge in the future. A byproduct of alignment of goals among different collaborating entities, which can be stakeholders of the project, leads to synergy, complementarity and compatibility. For example, open source projects are big unifiers. The Internet is the best example.

❖ **Increase diffusion and/or adoption rate of deliverables**

- ➢ There are many reasons why the diffusion and/or adoption rate of deliverables are/is increased. These reasons are related to the process, as well as to properties that the deliverables of OVNs acquire. First, the process is transparent. Like with open source development, people get to know what is being created, they build expectations. Since the process is also open, people can even participate in different ways, which means that more feedback is injected into the process. But the time the deliverable is ready for launch, people know about it. Since some future users have contributed to the process, the deliverable has a better chance to fit people's needs. Moreover, these deliverables have a lower cost of ownership (purchasing, maintenance, etc) than commodity products created by traditional organizations.

❖ **Distribute influence or formal power among contributors,** avoid enclosure or monopolization

- ➢ OVNs are nondominiums and thus they cannot be captured or hijacked. Moreover, their deliverables or products cannot be alienated. OVNs are organizations that guarantee peer governance of their deliverables or products.

❖ Add others...

Contributors: Tibi, Emily McGill, Chris Troutner, …

Your Vision, Mission and Positioning

Open document.

Map your value system

Open document.

Information and feedback

To be continued…

Structure

Let the structure be mainly emergent. Sensorica took a minimalist approach to structure, i.e. put structure in place only in response to a problem. The minimalist approach is risky and can come with a high emotional cost on the network, because some problems develop into conflicts. You can always remix other OVNs' governance if the value system is similar.

Infrastructure, Governance, Methodologies

At the fundamental level, organizations are a bunch of individuals trying to do something together. As the number of individuals increases and their activities become more complex, groups develop different types of problems, which require different types of solutions. We see elements of infrastructure, governance and methodologies as tools that respond to different types of organizational problems. These three levels are interlocked into a tight system that is compatible with the Open Value Network model.

Illustration
A number of people get together to develop a new technology. Their work becomes complex and people lose track of activities. Their effectiveness as an organization is diminished. This is an organizational problem.

The solution touches all three levels of Infrastructure, Governance and Methodologies. The group decides to create a wiki (an IT tool, part of infrastructure) where they can document their work. A wiki is a collaborative documentation tool, which is coherent with collaborative work methodologies. The group adopts open source development methodologies where documentation is part of the work process. In order to make the process even more efficient, the group might also adopt a project management tool. They use it to plan their work, create tasks, and set priorities. They can gain more efficiency if they tie the wiki, where the content that results from their work activities is deposited, to the project management tool, and make documentation tasks part of the planning. Thus the work methodology is well integrated with the tools that the group adopts. But, people usually don't like to document. So the group might consider some rules that incentivise documentation or penalize those who don't document. For example, the rule can say that if the work is not documented in a way that everyone can find, understand, revise and contribute to, the work contribution is not taken into consideration. If different types of benefits are attached to the group's activity, the rule can tie documentation to access to benefits. This shows how infrastructure, work methodologies and governance interact.

Infrastructure

Infrastructure is needed in order to allow a group of people to get things done. At the most basic level, infrastructure is a collection of tools.
The *open value network infrastructure* (OVNi) is documented on the OVN wiki here. The most important OVNi modules are:

- *network resource planning* and *contribution accounting system* NRP-CAS
- *content management system* (CMS)
- *communication systems*
- *coordination systems*
- *feedback systems*

The NRP-CAS is a software application that assists with resource management, project and task management, accounting, including accounting of contributions, and redistribution of benefits. It is a tool that supports crowd-based economic processes, thus it also integrates crowdfunding and crowdsourcing. The NRP is a successor of the enterprise resource planning (ERP) system used by medium and large classical companies, designed for networks.

We'll also discuss the *governance and normative system*, and *legal framework*.

Creating and implementing infrastructure

As a group grows in complexity it develops problems or becomes dysfunctional. At this moment, the group feels the need for tools, people propose some tools and implement them for the group, new tools develop new work habits, it gets messy because tools aren't well integrated, at

some point the group stops and restructures its infrastructure, and the cycle restarts. This is never ending in the lifetime of the organization.

It is difficult to propose infrastructure to a formed group, in absence of a problem, because lay people don't always understand IT infrastructure and its future consequences on the organization and adoption will be poor. People do understand the need of a new tool when a problem arises, if someone can provide it and make the link. Then they see the value, even if the tool proposed is only a prototype, and only loosely connected with the other tools. Once a tool gets recognized by people and is adopted, people develop new habits, and the group can restructure everything around. In short, tools are introduced as prototypes, incrementally, as a response to problems, in context. Only if a tool gets adopted and its use results in new work habits the group should spend energy integrating it into the main infrastructure.

When it comes to tools, there are usually 3 categories of roles: user, planner and admin. The user is almost anyone. Being a **user** requires only knowledge about how the tool is used in a specific context. The focus at the user level is on getting the work done. The next level is the **planner**. People in this role understand what the tool can do and they can configure the tool. This role requires knowledge about what the tool can do, in general. Planners teach users how to use the tool, choose and configure tools for specific tasks, schedule work and implement methodologies with the tool, etc. The third level is the **admin**. These people understand how the tool is built. They maintain the tool, teach planners about what's possible to do with tbe tool, about new features, and take feedback from users and planners to improve the tool.

[NOTE: copy this text also in the part of this doc where we talk about the NRP, replace tool by NRP]

Self-organizing mechanisms

These mechanisms help the OVN become more efficient, in context, while allowing it to remain adaptive. They also reduce the burden on governance.

Incentive system

The benefit redistribution algorithm is an important part of the OVN's incentive system. The design of the benefit redistribution algorithm and the Governance Document will determine how efficient and effective resources can flow into a process/project, and the development time. It can also affect quality, for example, if important skills are not properly incentivised.

Contribution accounting

The contribution accounting system records contributions to processes/project. It also shows OVN affiliates how much other affiliates contribute to a process/project. Those who contribute a lot might gain more influence, thus it plays a structuring role.

See a possible representation of the contribution accounting system in Sensorica for the Greens for Good venture.

The role system

A role is a set of activities. Examples of roles are: administration, facilitation, animation, R&D. Roles are emergent in an OVN. *** link to list of roles

It is important to identify the most important activities required for your value system in order to seed the NRP-CAS. They are used by the NRP-CAS to log contributions. As time goes by, new activities can be introduced into the system.

The role system is structuring because it tells **network affiliates** what other affiliates are doing in the context of a project, thus, newcomers can rapidly identify experts. See a possible representation of the Role system in Sensorica for the Greens for Good venture.

The role system is used by the NRP-CAS to signal needs to OVN affiliates. In other words, every time a new task is created in the system, those who have already successfully performed the task in the past receive an email notification.

The role system can also be used by the Resource management system to grant use access. See Physical resources governance page on the OVN wiki.

See the Role system of the OVN.

Role weighing
Adjust the relative weight of roles

Proposed procedure: present a matrix of roles and ask all members to weight them relative to each other, based on a *reference role*. Take the average and show it to the entire group. Discuss the result and decide if we can reiterate the experience based on lessons learned.

The reputation system

The reputation system is structuring because it tells network affiliates how well other affiliates perform their tasks in the context of a project. An individual with a high reputation score can gain more influence in a project.

In order to make reputation operational in the socio-economic space we need to link it to the incentive system, i.e. the contribution accounting system.

Reputation also affects benefits through the benefit redistribution algorithm, thus providing a very efficient self-exclusion for those who lose their reputation, because the cost to benefit ratio becomes attractive.

Access to tasks can also be modulated/prioritized by reputation, thus ensuring that those who create higher value gain first access.

Commitment to a task (see Project and Task management section) is a dimension of reputation. This adds more determinism into the value system, since OVN affiliates will try to keep their reputation score up by delivering on time, with the expected quality.

NOTE: although reputation is already a parameter in the NRP-VAS, SENSORICA hasn't implemented a reputation system yet.

The Reputation system of the OVN.

Dimensions of reputation

Commitment is a dimension of reputation. Our experience shows that not keeping a commitment is a major problem in collaborative projects. One way to go around this problem, in systems based on voluntary involvement and in absence of power relations, is to link this dimension of reputation to the benefit redistribution algorithm. In other words, if there is a pattern of not keeping self-imposed commitments, other affiliates are allowed to diminish the reputation/commitment score of the person.

Commitment can be defined by dimensions of *time* (deliver x at some date) and *quality* (deliver x at some date with y quality level, or responding to all requirements).

Other reputation dimensions that have been singled out in our experience with SENSORICA are **related to the collaborative atmosphere**. People can be rude, conflictual, not respecting rules,

Governance and normative system

Start with these links

- OVN governance
- Sensorica governance page
- Fluid p2p governance
- OVN Governance Canvas

Contributors: Tibi, Emily McGill, Chris Troutner, …

There are 3 levels of governance.

- *network-of-networks* - mainly interoperability with other OVNs
- *network* - mainly about network operations and shared assets
- *project* - mainly about project operations

Access to governance can be provided to OVN affiliates through the Governance equation.
NOTE: Sensorica hasn't implemented a governance equation yet.

Norms are not yet formal rules. They are more or less explicit or documented. Fluid p2p governance describes an emergent and distributed normative system.

Legal aspects

Legal structure - see Sensorica's legal structure. See the wiki page on legal structure.

Roles, Relations and Rules

Tibi wants to put something here...

Contributors: Tibi, Emily McGill, Chris Troutner, …

Create spaces

Content management

Content management systems define a virtual environment. Its characteristics drive behavior and organizational structure.

Example: if capturing and posting information requires special technical skills (a website difficult to edit) a class of content keepers will emerge and the rest of participants will be dependent on them. Moreover, it will create a community where people are not empowered relative to content generation.

Main characteristics

- **access** (about openness) - not just a password, but also ability to act by reducing the barriers (technical and others) to content creation and maintenance.
- **transparency** - easy to find, understand, place in context
- **flow** - shareable content across different projects/networks/communities

Contributors: Tibi, Emily McGill, Chris Troutner, …

Project and task management

See the OVN wiki page on Planing.

Contributors: Tibi, Emily McGill, Chris Troutner, …

Determinism in processes

Having processes being carried out in a coherent manner (coherent with resources available, time constraints, etc.) and a deterministic manner (predictable in all outcomes). e.g.. Customer orders product, needs to be delivered at a date (with all the specified requirements).

This depends on the scale and type/nature of the process.

Scale

- swarming type of project, large scale, possible global (translational), has a long tail structure, relies on stigmergy
- small scale process, small group, local or not.

Large scale (swarm-type) processes operating in **long tail mode** must have a fair amount of redundancy, meaning that for every task there are more than one available resource (can be individuals who can work on them, or material resources, or other). The greater the redundancy the higher the probability that the task will be carried out. In this regime the load on governance is very low. This is how Wikipedia operates; at every moment there is someone in the world who can correct a page from vandalism.

Small scale projects don't have redundancy and tasks are shared across a small number of agents. This means that some processes are effectively monopolized by some individuals. If there is disagreement on how to do the task, how well, when to be delivered, the community needs to apply coercive measures, since there are no other individuals that can be allocated to the task. This generates the need for power relations (others accept or it is allowed that someone can force someone else to do things in a given way) or heavy governance systems (the group decides to coerce someone to do something).

See Tibi's presentation: no boss no chaos.

Design of benefit redistribution algorithm

Create a (visual) tool for open and collaborative design of benefit redistribution algorithms.
How people see contributions and how to link them to benefits. How do we design economic games that make projects sustainable?

Design a methodology that goes with this tool.

Contributors: Tibi, Emily McGill, Chris Troutner, …

Design the UX/UI for this tool.

Benefit redistribution algorithms are culture dependent and context dependent.
We need tools to structure the design process, and to help build consensus.

Example from SENSORICA

Use this space to design the benefit redistribution algorithm.

Text from Guerilla Translations

See text that describes an incentive system and governance system proposed within the Guerilla Translations OVN.

Contributors: Tibi, Emily McGill, Chris Troutner, …

Problems to be aware of

Navigation - Legibility of p2p networks

p2p networks share a very important problem: newcomers don't know where and how to engage, because the space is not legible.

This might be more about signaling/signalization than putting in place rigid structures. Networks are inherently emergent, they continuously morph into something else. Therefore, orientation is not defined by the structure itself, which is dynamic. The signaling needs to follow the evolution.

Some solutions

- structure projects (tells people what's cooking, why they should get involved)
    - How to create signaling that follows the continuous morphing of the environment?
    - What are the basic necessary signals?
        - Enter
        - Exit (and perhaps exclusion?)
        - Forking
        - Direction (goal)
        - Related to activity (what to do, when, why)
        - Related to resources (where they are, who has access [is there priority seating?] how to use them)
        - How to que up (what are the cultural norms?)
- define tasks (allows people to find and understand what to do NOW)
- put in place a Role system (tells others who is doing what)
    - Who maintains order? (is there a police system?) How are conflicts resolved?
    - Who does maintenance cleaning?
    - Who makes sure the trains don't crash?
    - Who collects the money? (and how can it be trusted?)
    - create a role of "integrator" (find a better name) who guides newcomers
    - facilitation, coordination, curating, scribing. (These roles need continuity)
- add more...

What else can go wrong?

First, networks are constantly renewing themselves. There is a constant flow of affiliates.
One way to see the OVN is as an attractor - presents some mechanisms of **accumulation**.

Quitting the network and forking of activity as well as forking of the network itself is natural, don't fight it, work with it.

Possible failure points

- the system is not transparent enough - people don't know who is doing what, how much and how well. This poses a lot of problems:
    - some are overworked (other don't know that),
    - governance imbalances (people who have contributed a lot are not are sometimes not allowed to take lead, responsibility, voluntary subordination doesn't happen),
    - Incentives are lacking (acknowledgment, being valued for the work done)
- lack of proper documentation (linked to transparency)
- culture:
    - people understanding that they can take initiative,
    - working in an open way - broadcast problems and needs along successes, in order to crowdsource problem solving
    - self-critical/self-reflective and allow constructive criticism

Tools

- Google: easy access.
- Share content across networks & allowing communities with different cultures to co-create at a scalable level.
- Presentable.
- A fractal structure that allows flow and ability for the network to evolve into something else.
- Something to look at: https://slack.com/
    - Perhaps Drupal/Civicrm's parent, child and "cousin" sites with a shared database and many levels of permissions, and open source nature can fit the OVN need.

See Tibi's article on open network evolution stages and problems.

Capacity building

Is an activity that is not directed towards a goal
Network weaving.
Reinforcing connections through passionate and empathic relations.
This work is not documented and has no deliverables.

Contributors: Tibi, Emily McGill, Chris Troutner, …

Capacity building is different than network animation, which is to give vitality to a specific action. Capacity building is building latent potential for whatever action might precipitate in the future. The p2p network is always starting and ending at the same time, there is always someone ending something and starting something else.

Onboard people in a cognitive structure.

This idea comes from Vinicious, in a discussion with Tibi.

Definitions

Term	**Definition**	**Example usage?**
Affiliates	Agents within the OVN who contribute (usually individuals, but could be organizations or networks themselves, depending on the project?)	
Fractal structure / system	Complex, non-linear and emergent systems which adapt to changing environments. Also; applicable at varying scales using the same basic rules/methodology.	
Open collaboration	“Any system of innovation or production that relies on goal-oriented yet loosely coordinated participants who interact to create a product (or service) of economic value, which they make available to contributors and non-contributors alike”*	
Self-organizing	Order arising out of apparent disorder through individual activities	
Swarm projects		

Contributors: Tibi, Emily McGill, Chris Troutner, …

NRP installation and configuration

This is about installing a new instance of an NRP and configuring it for a new OVN.
For questions, ask Bob and Chris Troutner, and David A Hand. Coordinate with Tibi.

See also Bob and Lynn's manual for configuring the NRP.

Choices

Which version of the NRP to install?
Currently, the Sensorica NRP lives on OpalStack. Log in to SSH as the shell user:
https://help.opalstack.com/article/14/ssh-access
NOTE: The NRP requires some legacy libraries and a new installation might not work on another hosting service. In fact, it is very hard to make it work on another service than OpenStack. We strongly recommend using the same service if you want to install and play with the NRP.

Branches of the NRP

- the original, which Sensorica is using now
- the Freedom Coop fork, integrates Faircoin [has been upgraded to more recent versions, but probably none of the process stuff is tested, they don't use that]
- the GoPacifia fork of the Freedom Coop fork [probably dead]
- django rea
    - the https://github.com/django-rea/rea-app project which aims to radically remake the NRP software, first with a new UI, and then with a decentralized back end:
        - Multi-platform UI application for OVN (Open Value Network) & REA (Resource / Event / Agent) backends- including Sensorica NRP, FreedomCoop OCP, GoPacifica DEEP & django-rea project.
- Chris Troutner's fork of the Sensorica version eventually
- Bob and Lynn fork of the Freedom Coop fork - not under development

Installation

Ask Tibi for the last conversation during the Matrioshka NRP installation, search [Tibi] email for ''*NRP-VAS for Matrioshka project*''.

Chris's instructions for installing the NRP

Open the source

NOTE: make sure you're installing the NRP version you want, from the list above.

How to Install the Value Network Accounting Software

The Value Network Accounting software is an implementation of Network Resource Accounting (NRP) software, used to track expenses and labor used in an open value network (OVN) like Sensorica. There is a slide deck of tutorials showing how to use the software. I've found that the best way to digest the tutorials is to install my own local copy of the software in order to play with it. The instructions below are adapted from the original installation instructions in the GitHub repository.

The instructions below are for installing the Value Network Accounting software on a Windows machine running a Linux Virtual Machine (VM) using VirtualBox. I was using the Windows 8.1 64-bit OS and using VirtualBox v5.0.10. I installed Ubuntu 14.04.03 Desktop 64-bit on the Linux VM. Your mileage may vary.

Note: Even though I was using a 64 bit OS on my Lenovo laptop, VirtualBox would only allow me to install a 32 bit operating system. It turns out that Lenovo laptops (and other computers) need to Intel Virtualization Technology turned on in their BIOS to allow the installation of a 64 bit VM, as per this discussion thread and this blog post.

- Here is the download page for VirtualBox
- Here is the download page for Ubuntu 14.04.03 LTS 64-bit Desktop

I'm assuming the reader has the technical competence to install VirtualBox and the Ubuntu OS onto it. If you run into problems, use Google. There is a lot of good documentation that is very easy to find on all the intricacies of the installation process. From here, I will assume that you are looking at your Ubuntu VM desktop. In terms of VM resources, I set up the VM with a 12GB hard drive, 1024MB of RAM, and 2 processors.

Steps to Install Value Network Accounting

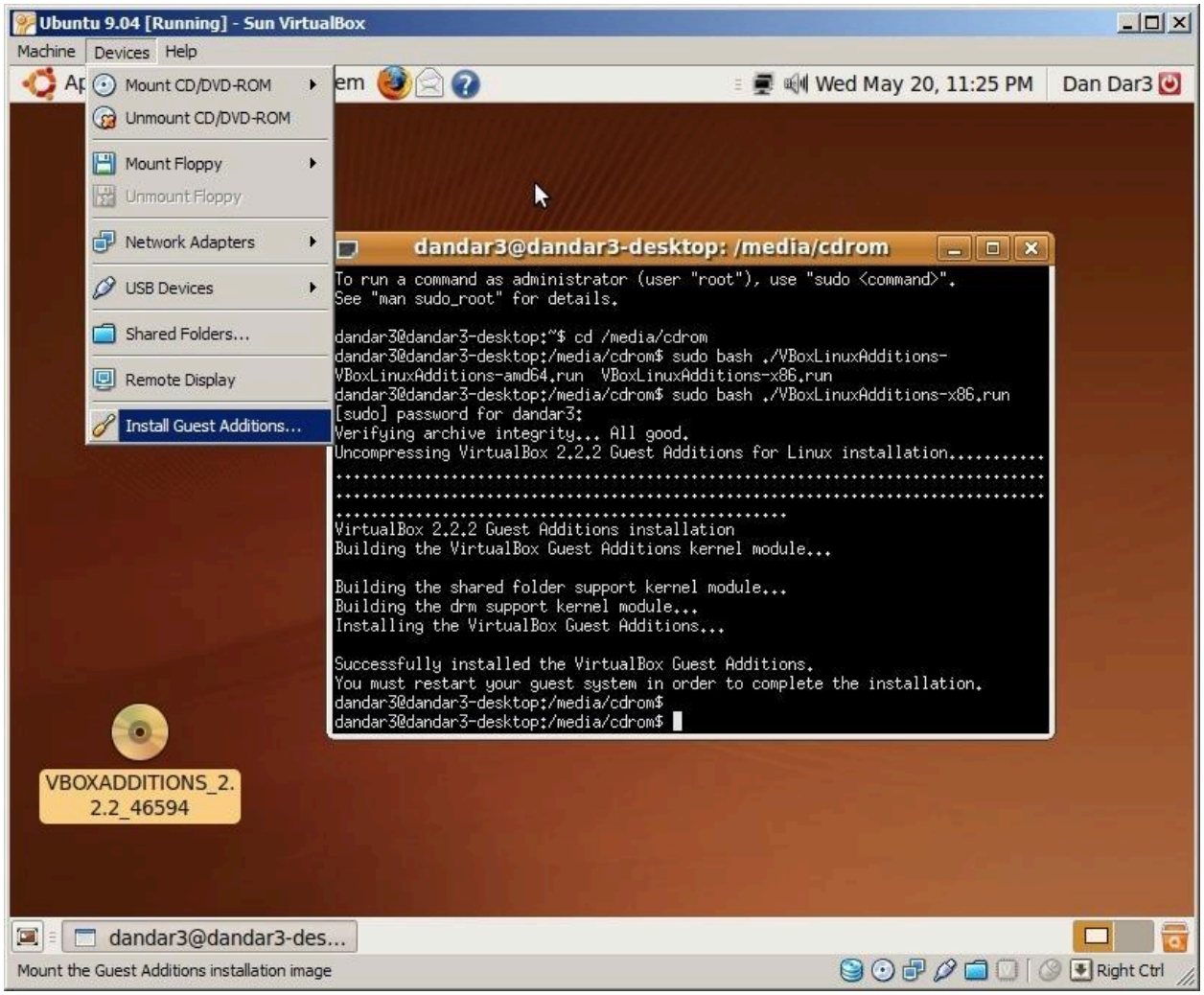
Environment Setup

There are a few helpful pieces of software we should install in the environment before installing the valuenet software. This will make use and development of the valuenet software a lot easier. It's a great idea to create a 'Base' snapshot of your VM when you're done with the section.

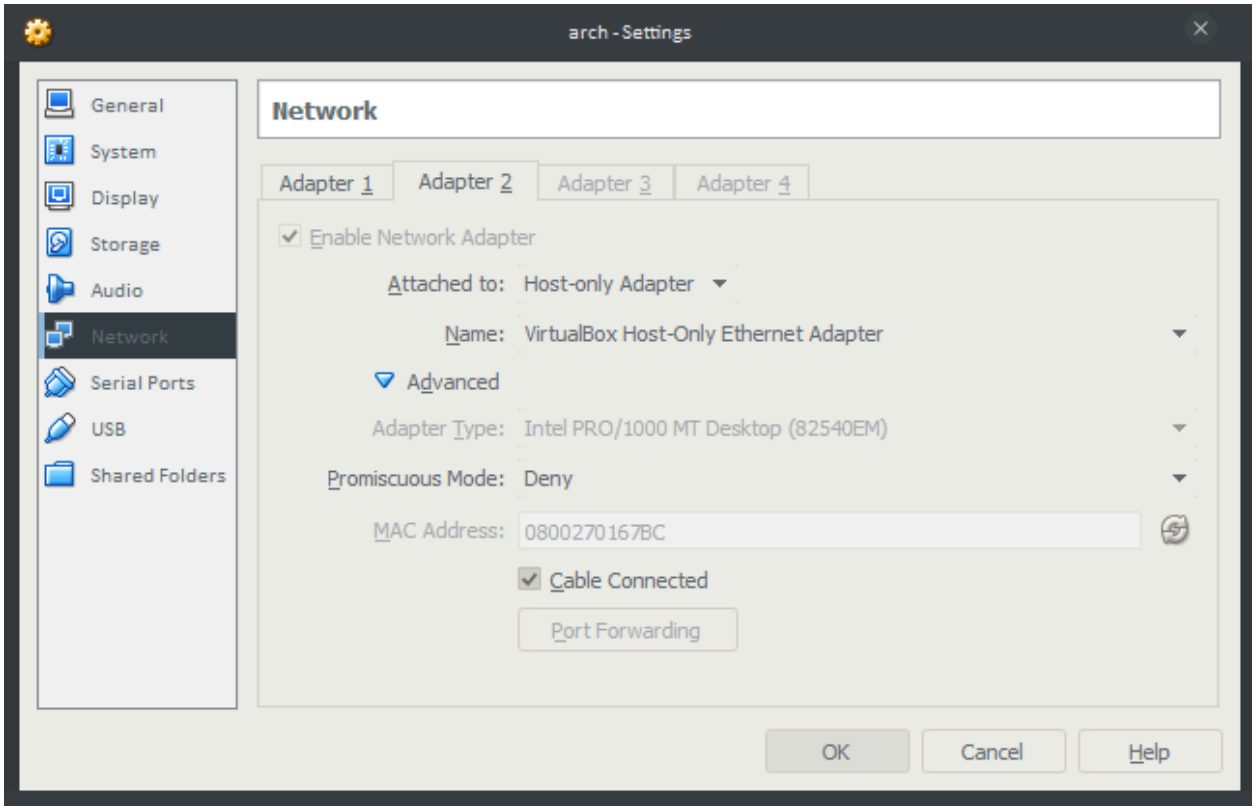
1. Start by opening up a Terminal window. If you're brand new to Ubuntu, click on the Ubuntu icon in the upper left corner and type in Terminal, then hit enter. Text you should type into the terminal window are marked in italics below. You can also use **CTRL + ALT + T**

2. When editing a text file, my preference is to use the **nano** text editor in the terminal. To open a file, enter nano textfile.txt in the terminal window. To exit the text editor hit CTRL + X and answer y or n to whether you want to save the changes.
3. Start by installing the Guest Additions CD. This will fix your screen if it's tiny and allow you to add and remove network cards. See the screenshot below. The CD should autoinstall a bunch of good stuff.
4. After the Guest Additions finish installing, shut down the VM. With the VM shut down, install a Host-Only Adapter in the Virtualbox Manager. See the second screenshot below. You should also check out this tutorial on installing SSH on a Linux VM, which is where I'm going with this.
5. Boot up the VM and Install the SSH server. I used this tutorial, which worked well for me. By getting the SSH server running, you can connect to your VM with PuTTY (or some other SSH terminal client) and work in your native Windows environment instead of switching back and forth between Windows and the Desktop.
6. From the Terminal window inside the VM or using PuTTY, run the following two commands:
    a. sudo apt-get update
    b. sudo apt-get upgrade
7. Your environment is now setup! This is an awesome time to take a snapshot of your VM in VirtualBox Manager.

Contributors: Tibi, Emily McGill, Chris Troutner, …



Screenshot of installing the VirtualBox Guest Additions CD



Installing a Host-Only Adapter in VirtualBox Manager

Installing the Value Network Accounting Software
NOTE: The original instructions pointed out that Python 2.7+, but not Python 3+ is required. This may be an issue in the future.

1. Install python setup tools:
    a. *sudo apt-get install python-setuptools*
2. Install Pip:
    a. *sudo easy_install pip*
3. Instal virtualenv:
    a. *sudo pip install virtualenv*
4. Install virtualenvwrapper:
    a. *sudo pip install virtualenvwrapper*
    b. *export WORKON_HOME=~/Envs*
    c. *mkdir -p $WORKON_HOME*
    d. *source /usr/local/bin/virtualenvwrapper.sh*
5. Git should be installed, but just in case, run:
    a. *sudo apt-get install git*
6. Install PILlow (python image library). I found this discussion thread helpful.
    a. *sudo apt-get install libjpeg-dev*
    b. *sudo apt-get install libtiff-dev*

    c. *sudo apt-get install libfreetype6-dev*
    d. *wget http://downloads.sourceforge.net/project/openjpeg.mirror/2.0.1/openjpeg-2.0.1.tar.gz*
    e. *tar xzvf openjpeg-2.0.1.tar.gz*
    f. *cd openjpeg-2.0.1/*
    g. *sudo apt-get install cmake*
    h. *cmake .*
    i. *sudo make install*
    j. *pip install pillow*
7. Create a virtual environment:
    a. *cd*
    b. *mkvirtualenv vn --system-site-packages*
    c. *workon vn*
    d. *Cdvirtualenv*
8. Clone the valuenet repository inside your virtual environment:
    a. *git clone https://github.com/valnet/valuenetwork.git*
    b. *cd valuenetwork*
9. Install the requirements:
    a. *pip install -r requirements.txt*
10. Install easy-thumbnails:
    a. *pip install --no-deps easy_thumbnails*
11. Create and initialize the SQLight database
    a. *./manage.py syncdb*
    b. *./manage.py migrate*
12. Install some starter facets and patterns:
    a. *./manage.py loaddata ./fixtures/starters.json*
    b. *./manage.py loaddata ./fixtures/help.json*
13. Run the tests to verify everything installed correctly:
    a. *./manage.py test valueaccounting*
14. Create a local settings file and add the following line to it:
    a. *nano valuenetwork/local_settings.py*
    b. *STATIC_URL = "/static/"*
    c. Cntl+X to exit, hit *y* to save.
15. Start the Django server:
    a. *./manage.py runserver*

That should hopefully get the Django server running without any issues. You can then connect to the valuenet front end by opening a browser in the Ubuntu desktop and connecting to http://127.0.0.1:8000.

Running the Software After Reboot

Upon rebooting the VM, here are the commands you'll need to get the Django server and NRP software running again:

> *export WORKON_HOME=~/Envs*
> *source /usr/local/bin/virtualenvwrapper.sh*
> *workon vn*
> *cdvirtualenv*
> *cd valuenetwork*
> *./manage.py runserver*

By default, Django only binds to the localhost address of 127.0.0.1. If you want to serve pages to your windows host box, then start the server with this command:

> *./manage.py runserver 0.0.0.0:8000*

That will start the server and allow you to access the page from the windows box by pointing a browser at the VMs IP address. For instance, something like http://192.168.56.101:8000.
So that completes the tutorial! Be sure to add comments below with your experiences, areas you got stuck on, and suggestions to others.

Tibi's notes on configuration

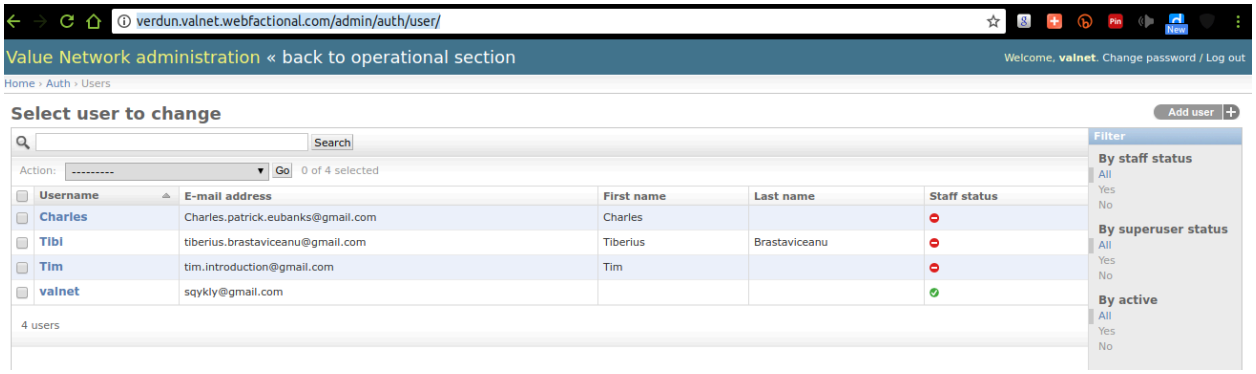
[from installation for Matrioshka]

> So as I am going through the process, I could remix your tutorial presentations http://nrp.matrioshka.io/accounting/tutorials/ to make them more useful. So there is an order of things here, that is missing in the tutorials. I think we need a step-by-step minded tutorial, in order to reduce the time it takes for someone to set up an NRP for a specific organization. I discovered how to do it because I was familiar with all these things. But I doubt that someone with absolutely no experience with the NRP would be able to do it without your help.

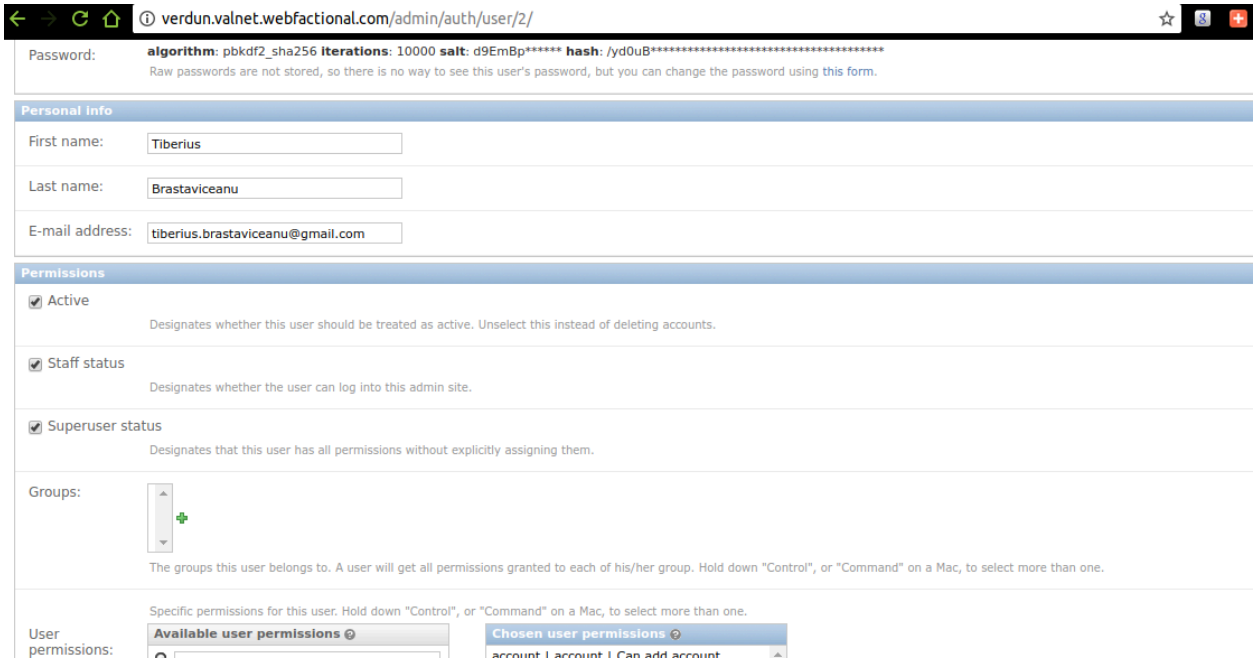
See also Bob and Lynn's manual for configuring the NRP.

Create **Users** from Admin

Contributors: Tibi, Emily McGill, Chris Troutner, …

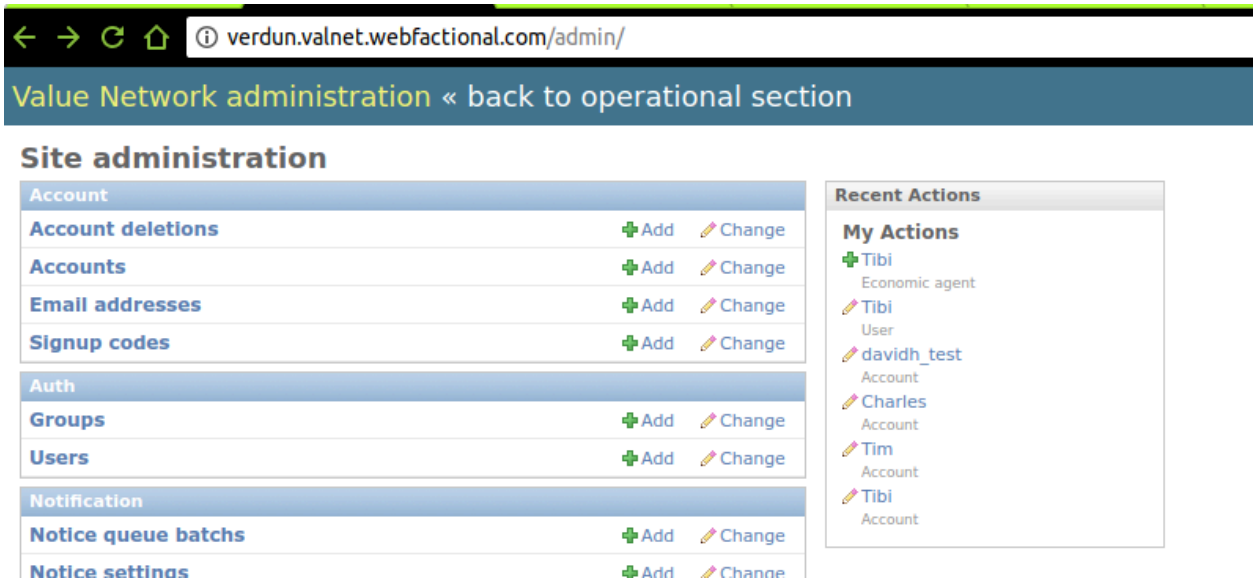


For providing admin access make sure you check the Stuff status button

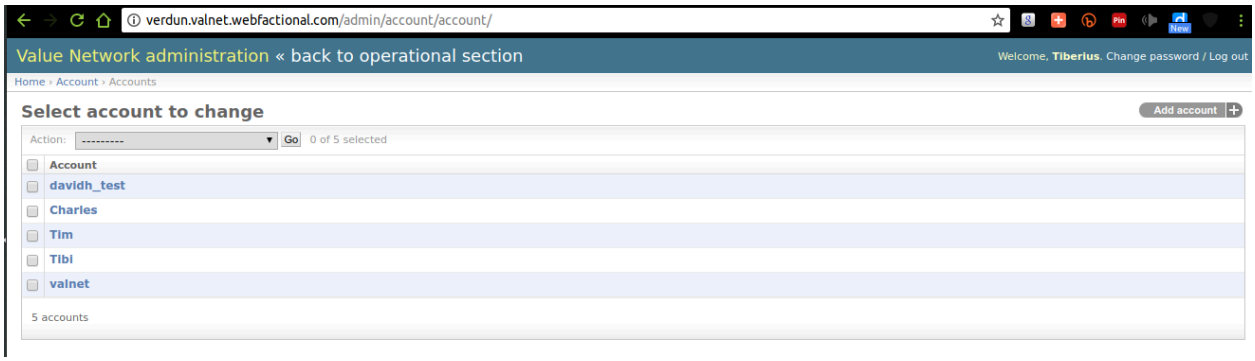


Create users from Accounts

Contributors: Tibi, Emily McGill, Chris Troutner, …

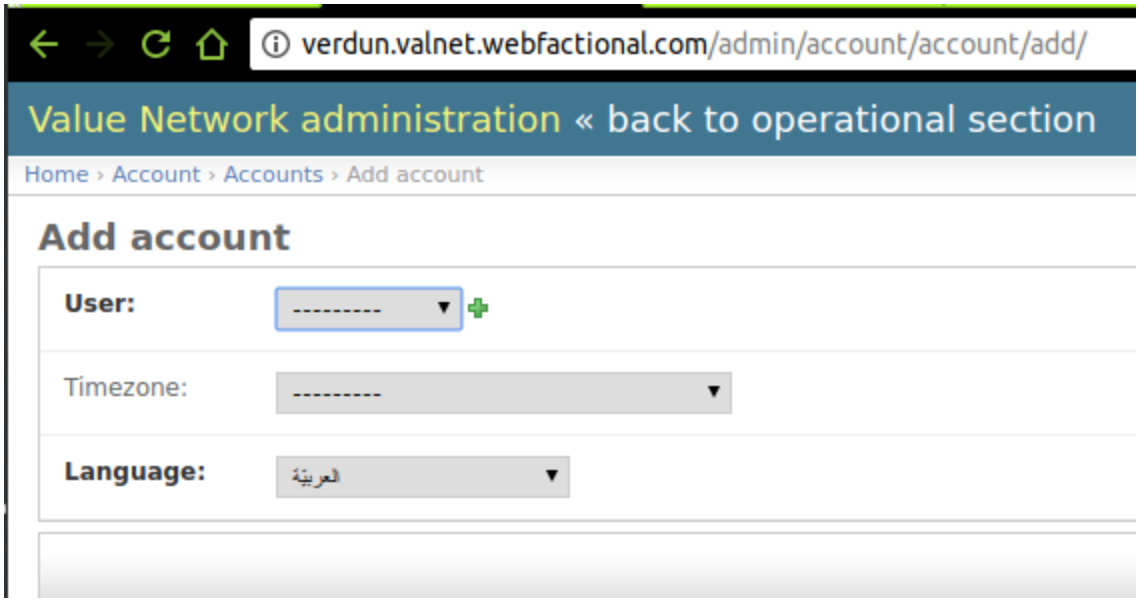


This is how the Accounts page looks like



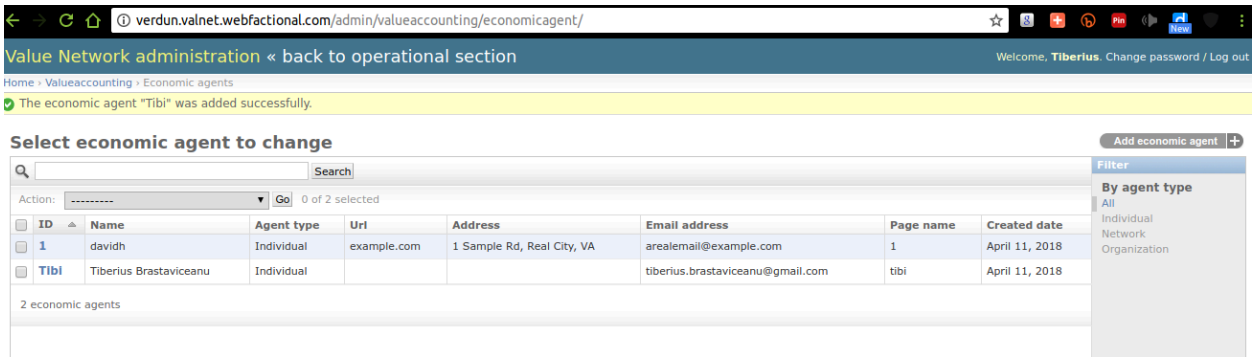
You add new account and in there you can create a new User, using the + button.

Contributors: Tibi, Emily McGill, Chris Troutner, …



Create **Economic Agents**

From Admin, go to Economic Agents and add new. Make sure you associate the proper User to this **Economic agent**. NOTE: Only Economic agents can operate on the UI. Without creating Economic agents you'll not have access to buttons to create Resource Types, or Recipes, or add new Agents, ...

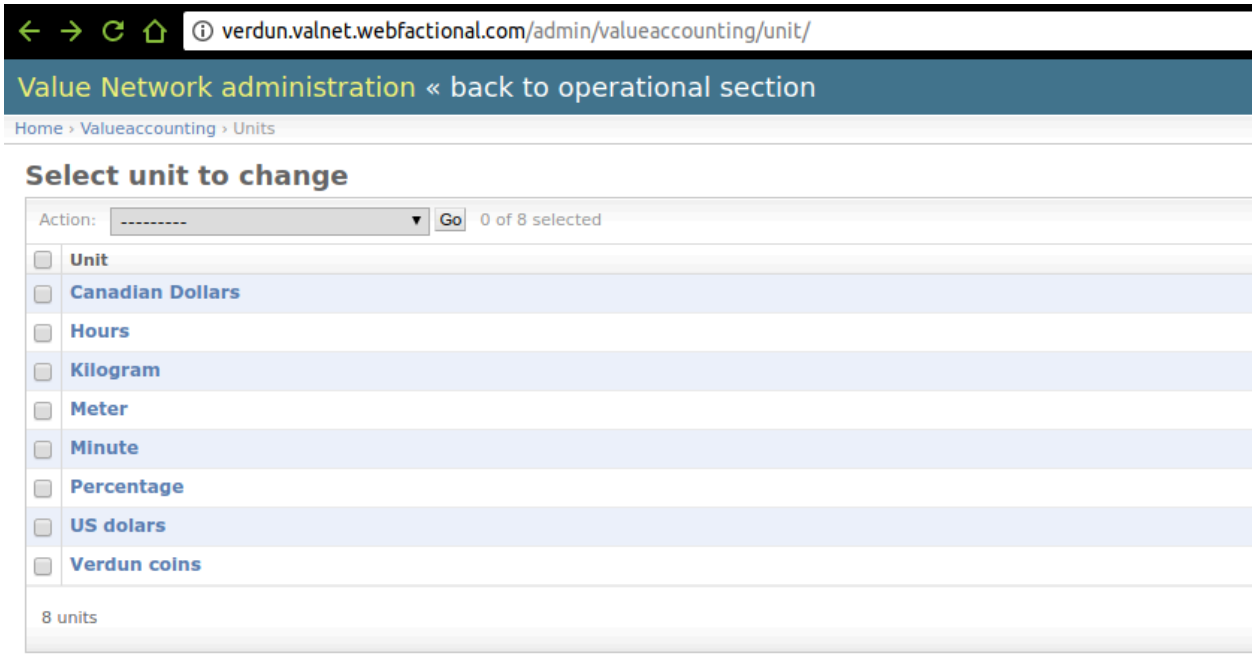


NOTE: make sure you adjust access rights (what every type of user can do in the NRP) from the beginning, because it will be harder later to go change that for everyone. Be mindful of what people can damage if not experienced and allow progressive access to do more and more things as people gain experience with the NRP. Create category of users.

Create Units

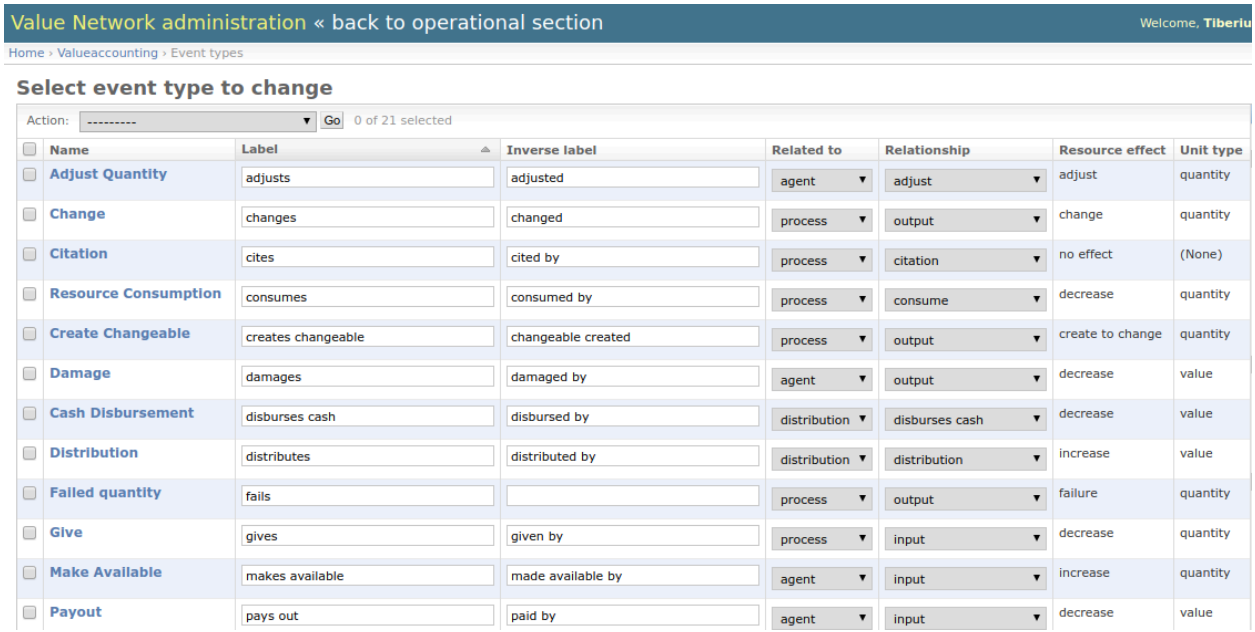
Before anything else, because creating Resource Types requires entering Units.

There is a hardcoded list of units like time, weight, length, value, … the idea is to create units like Time/hour or Time/minutes.



Create Event Types

You find the page to create and configure event types in Admin. It looks like this (Sensorica example).



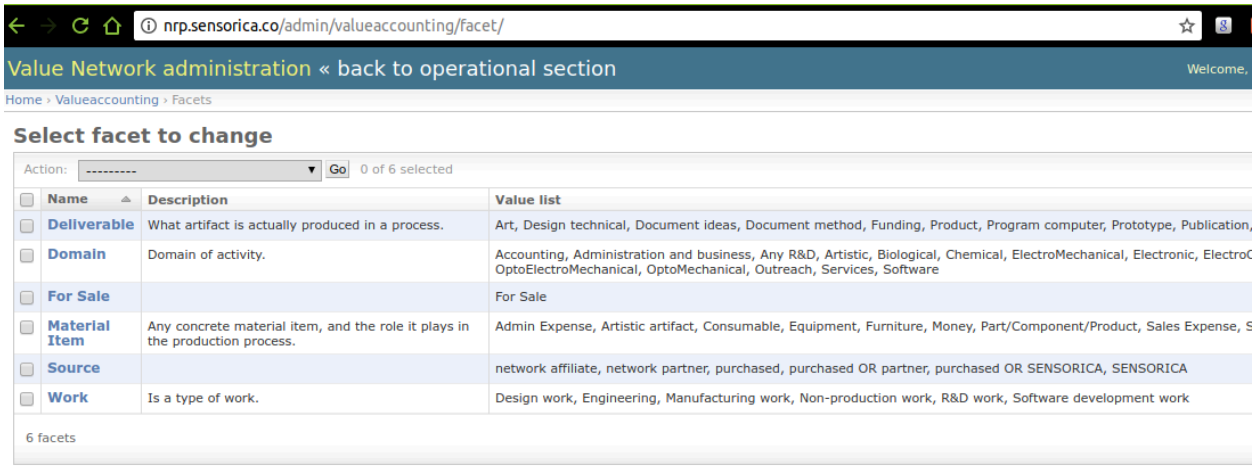
Name	Label	Inverse label	Related to	Relationship	Resource effect	Unit type
Adjust Quantity	adjusts	adjusted	agent	adjust	adjust	quantity
Change	changes	changed	process	output	change	quantity
Citation	cites	cited by	process	citation	no effect	(None)
Resource Consumption	consumes	consumed by	process	consume	decrease	quantity
Create Changeable	creates changeable	changeable created	process	output	create to change	quantity
Damage	damages	damaged by	agent	output	decrease	value
Cash Disbursement	disburses cash	disbursed by	distribution	disburses cash	decrease	value
Distribution	distributes	distributed by	distribution	distribution	increase	value
Failed quantity	fails		process	output	failure	quantity
Give	gives	given by	process	input	decrease	quantity
Make Available	makes available	made available by	agent	input	increase	quantity
Payout	pays out	paid by	agent	input	decrease	value

Note that EventTypes are hardcoded. See the NRP code.

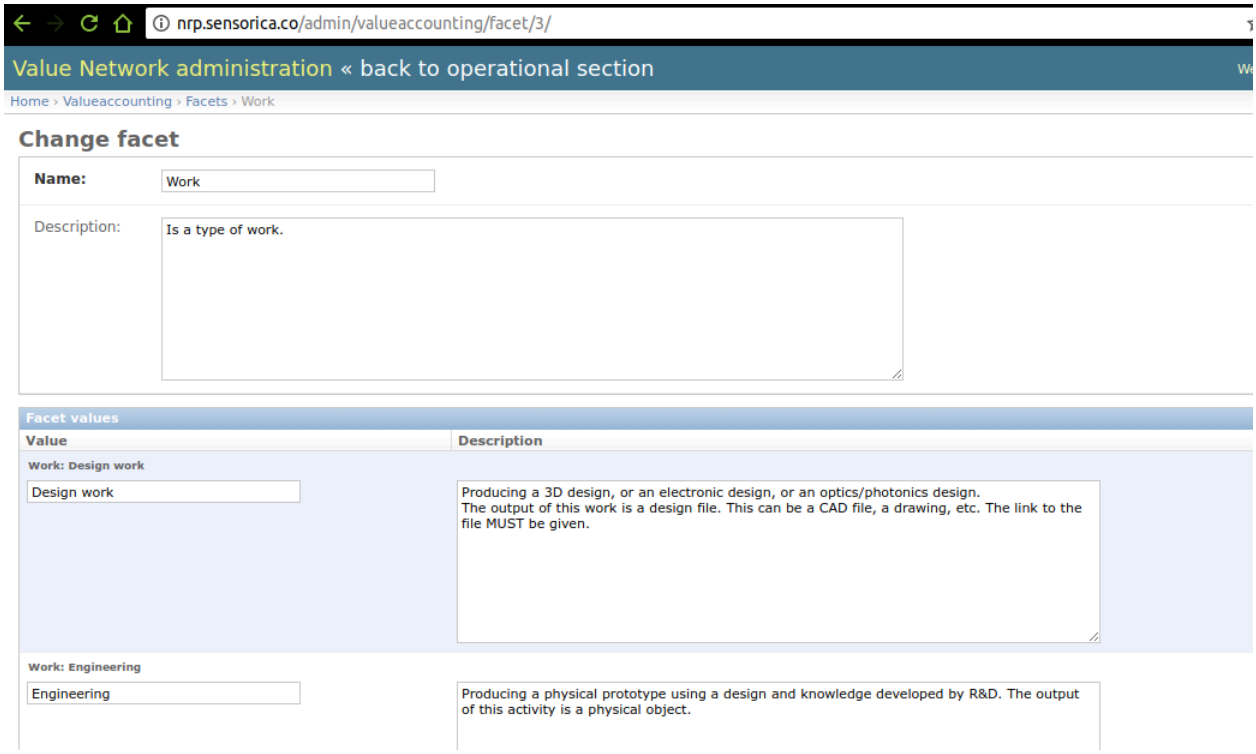
Create Facets

As I added more structure and moved towards planning, I discovered that I needed to **create Facets** first. I copied those from SENSORICA NRP. Once I created Facets I was able to create Process Patterns, which requires Facets.

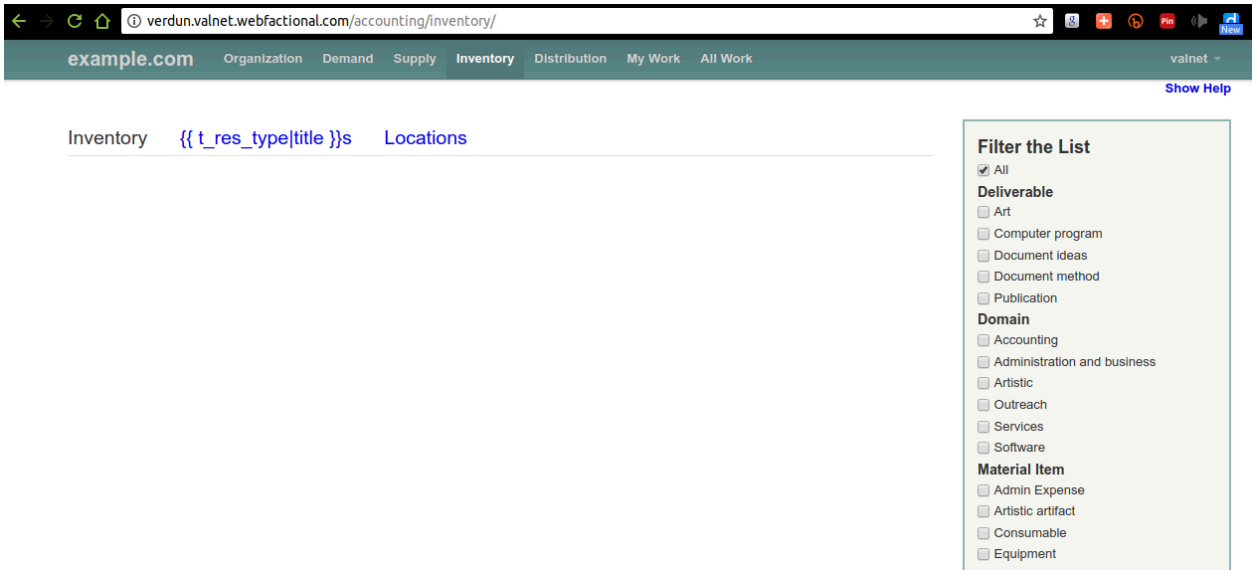
Here's an example from Sensorica



This is how the config page looks like



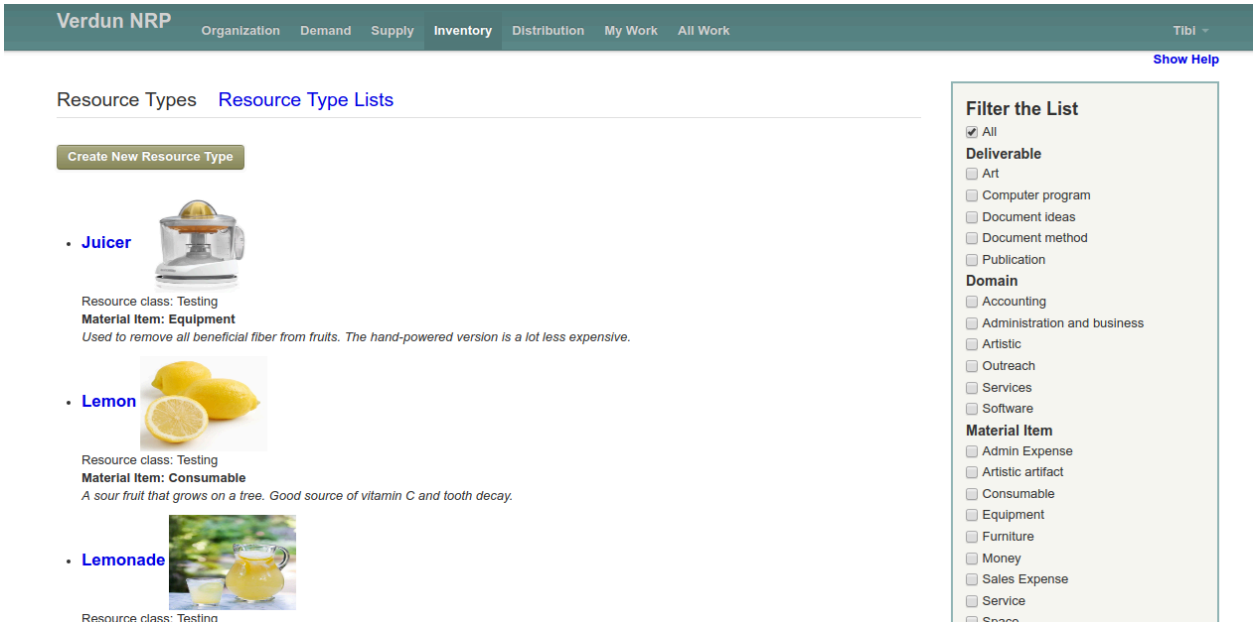
This will create filters for resource types. If you go to the Inventory page you'll see this being populated on the right side of the page.



After that, I was able to adjust facet values for *Resource types* in order to have them show up in lists in processes. This is when you **create *Resource types***.

They will end up on the Inventory/resource type page

Contributors: Tibi, Emily McGill, Chris Troutner, …



Create Resource Types

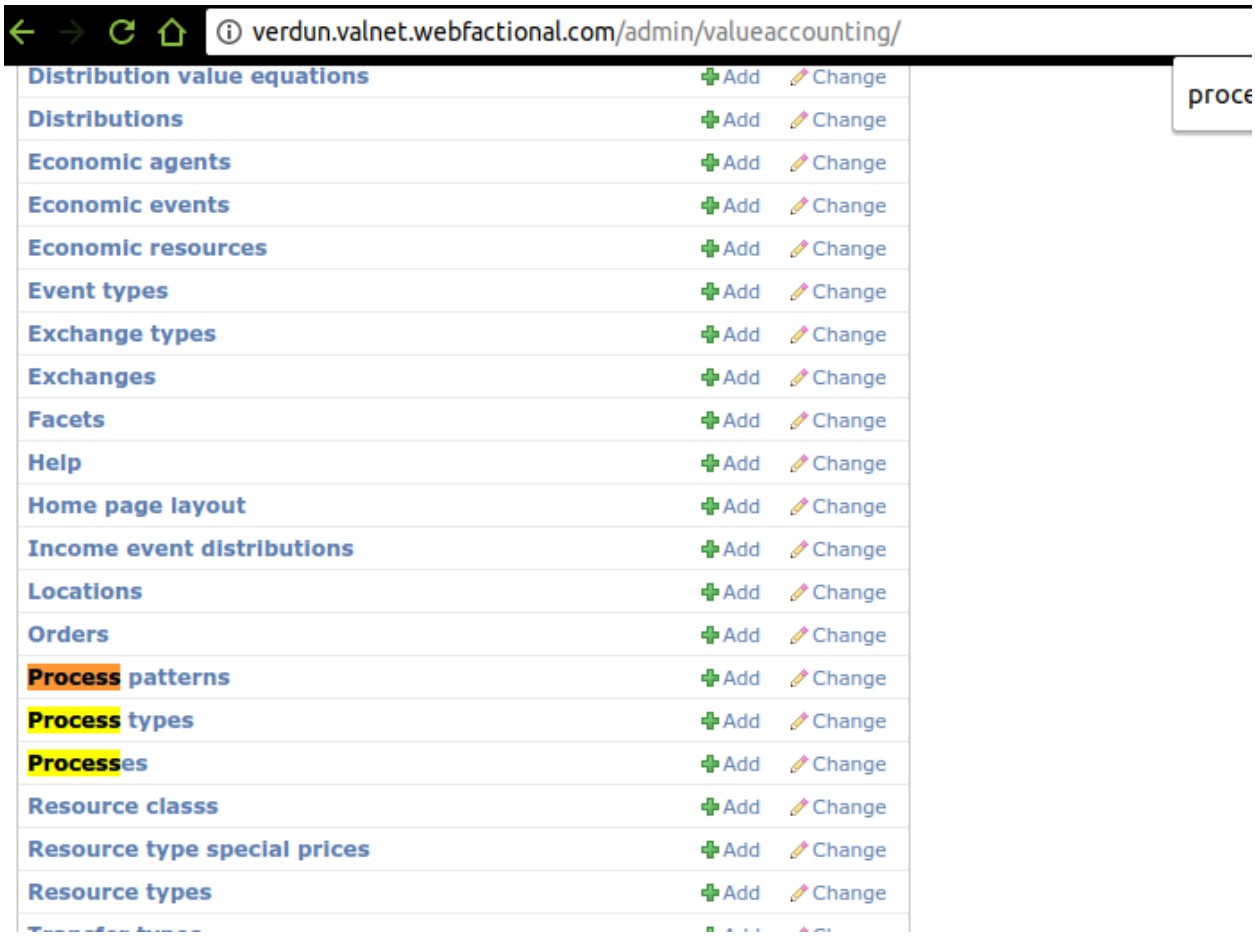
At this point, use the UI to create Resource types. If you have the proper credentials, you'll see the **Create New Resource type** button.

As you create new resource types, you need to set their facet values, in order to have them appear in your lists. So This is why Facets need to be defined before, see Above.

Create Process patterns

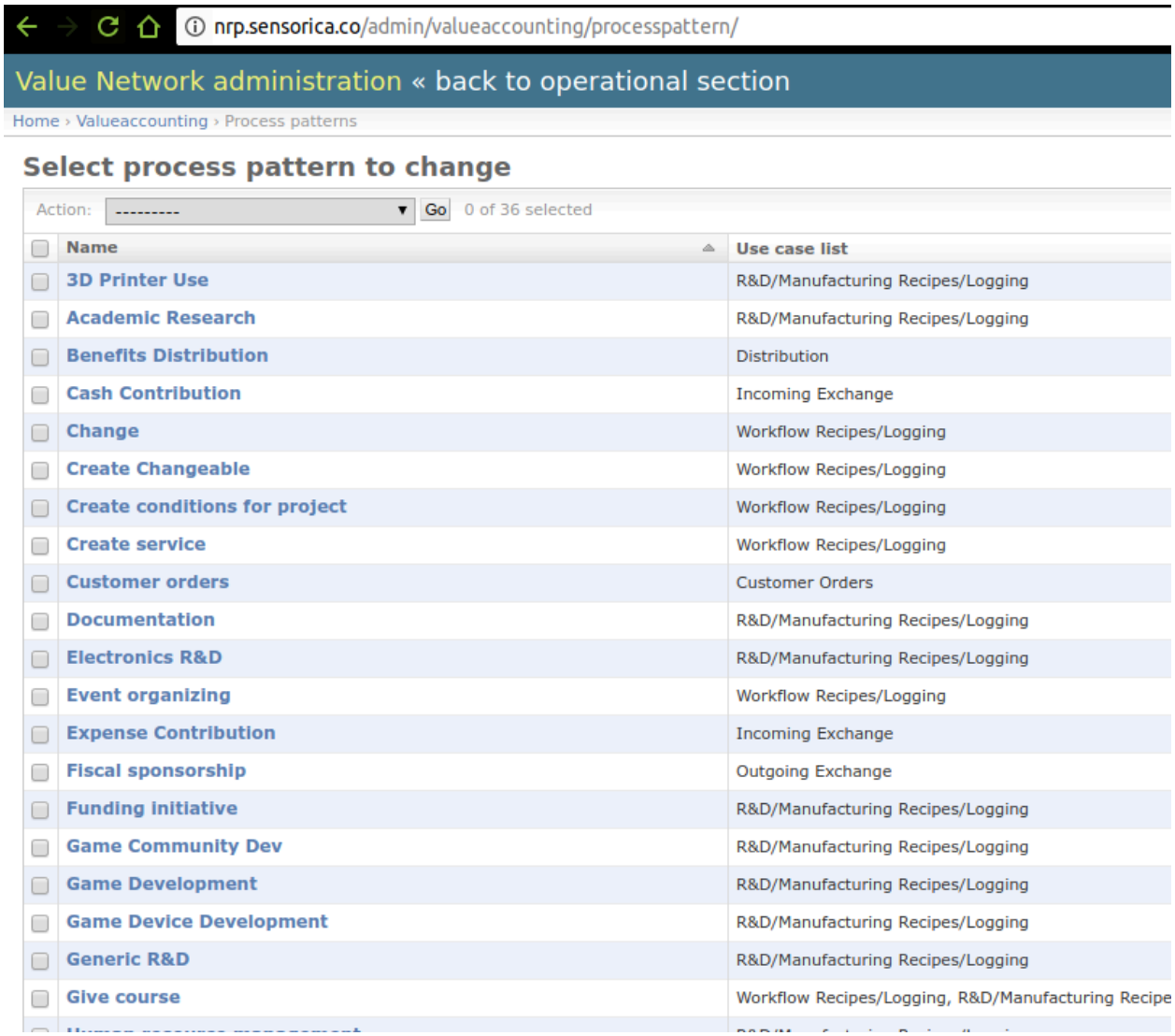
In Admin you go to Process patterns. Process patterns form a logging page. There, you define what goes on the page, such as consumption of consumables, use of equipment, time contributions, etc. Some are very important, like Benefit distribution for example.

Contributors: Tibi, Emily McGill, Chris Troutner, …



See an example in Sensorica's NRP

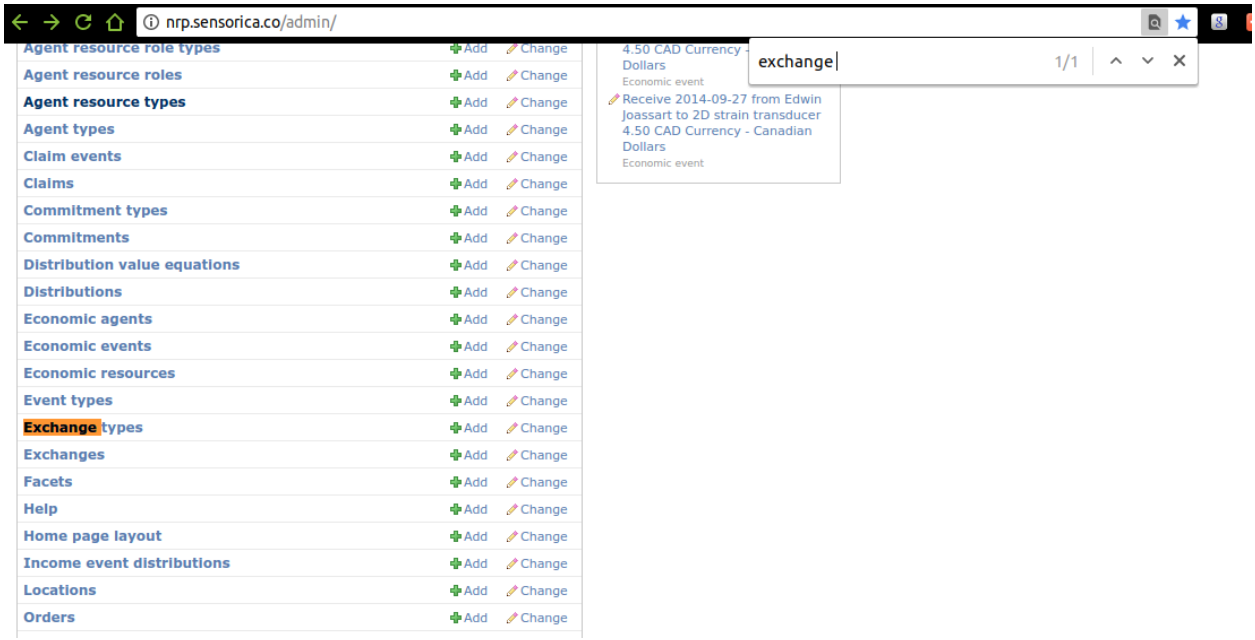
Contributors: Tibi, Emily McGill, Chris Troutner, …



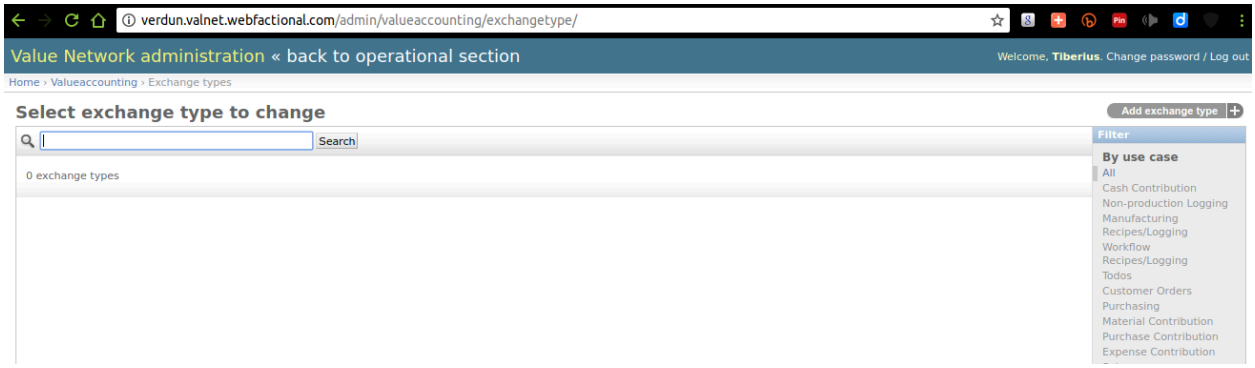
Create Exchange types

You need them before, to set **Transfer Types**.
You create them from Admin, like in the SENSORICA NRP.

Contributors: Tibi, Emily McGill, Chris Troutner, …



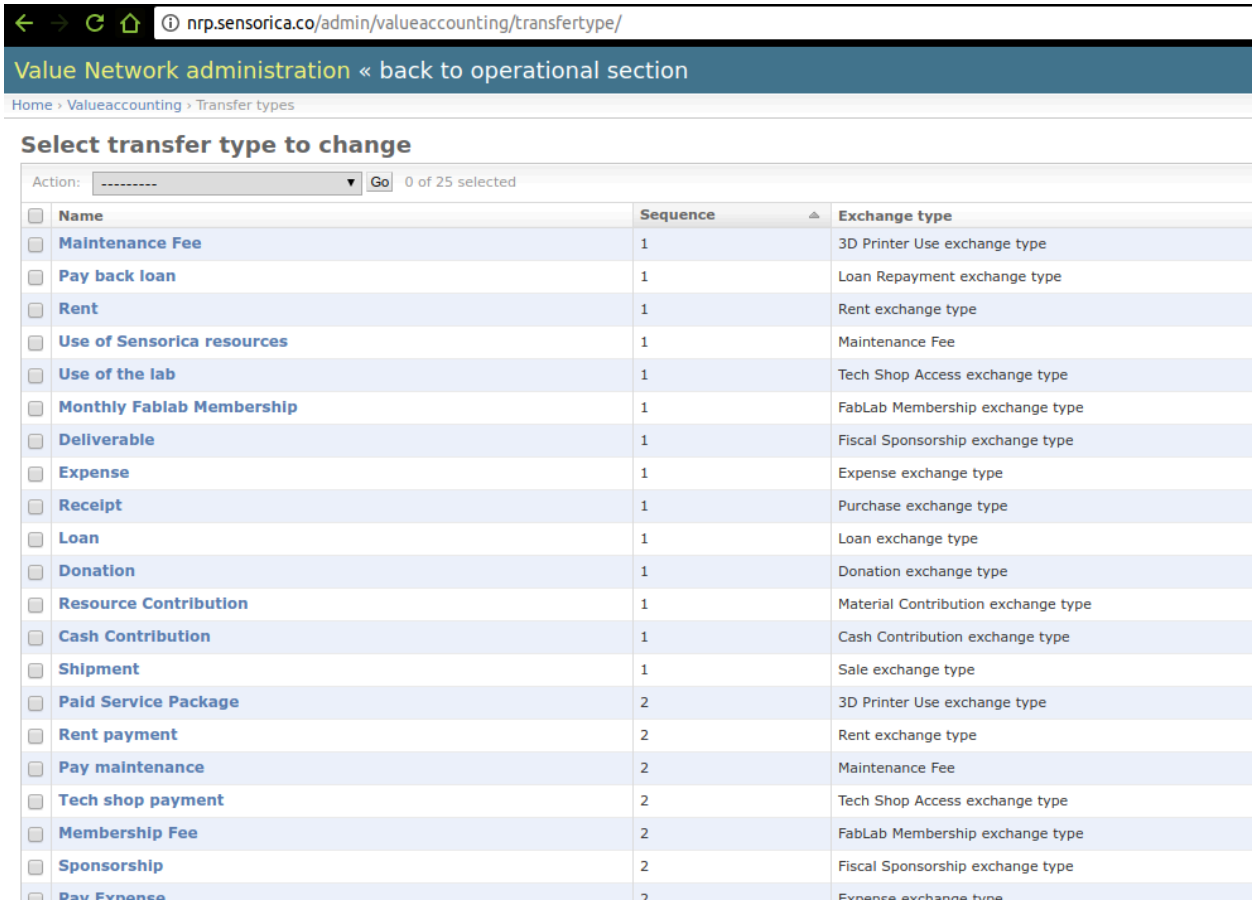
They end up here



Create Transfer Types

See example from Sensorica NRP

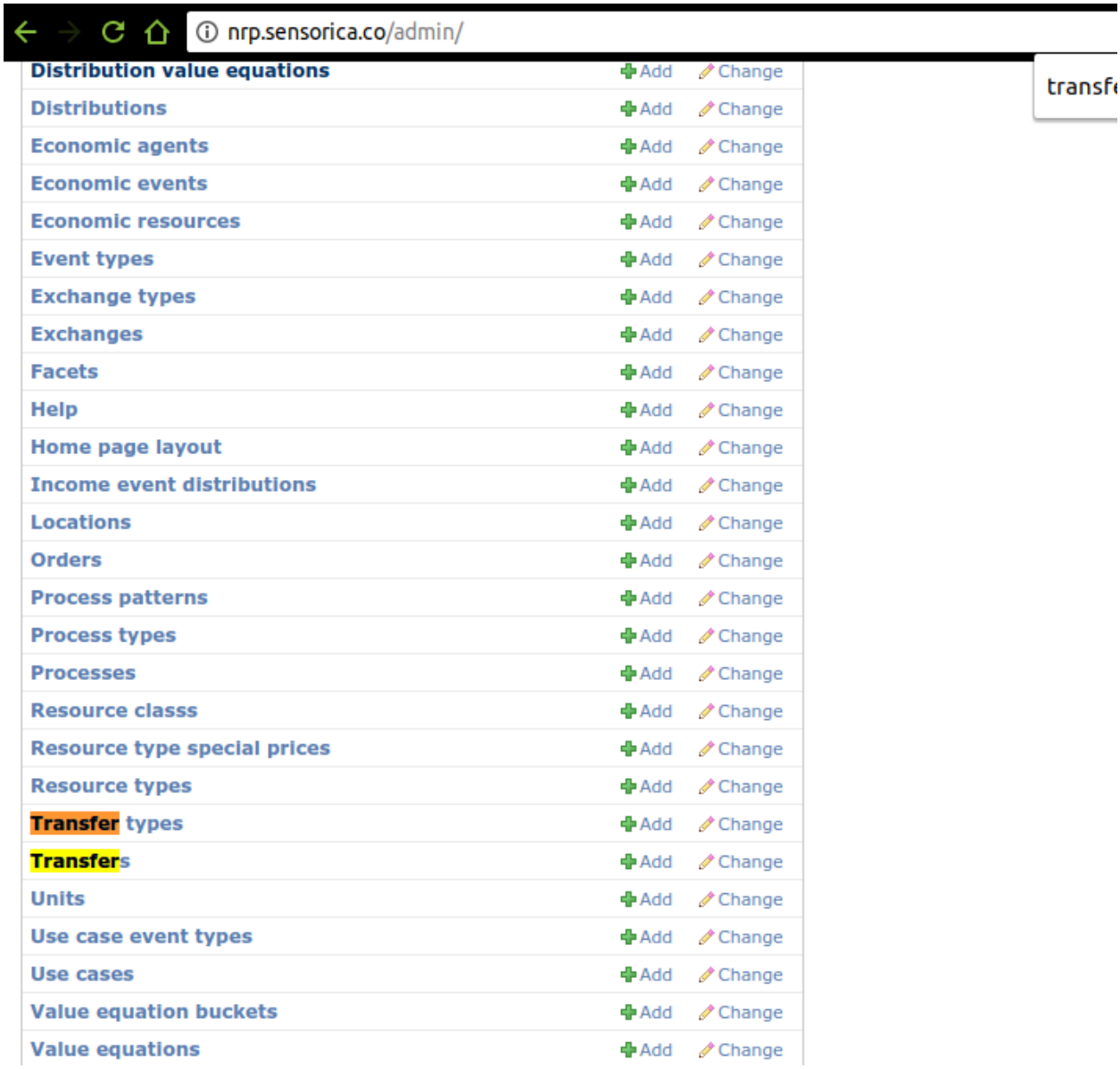
Contributors: Tibi, Emily McGill, Chris Troutner, …



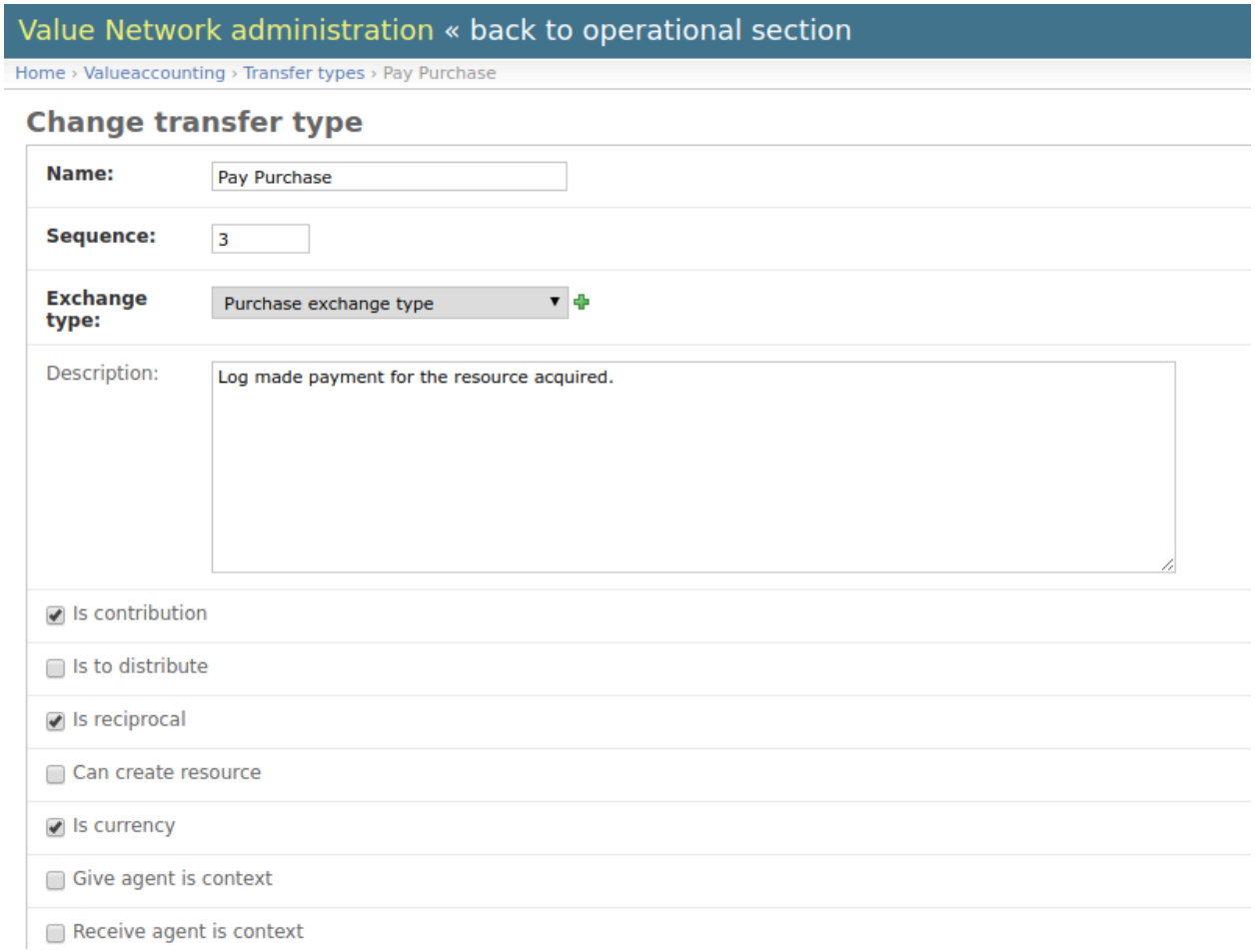
Name	Sequence	Exchange type
Maintenance Fee	1	3D Printer Use exchange type
Pay back loan	1	Loan Repayment exchange type
Rent	1	Rent exchange type
Use of Sensorica resources	1	Maintenance Fee
Use of the lab	1	Tech Shop Access exchange type
Monthly Fablab Membership	1	FabLab Membership exchange type
Deliverable	1	Fiscal Sponsorship exchange type
Expense	1	Expense exchange type
Receipt	1	Purchase exchange type
Loan	1	Loan exchange type
Donation	1	Donation exchange type
Resource Contribution	1	Material Contribution exchange type
Cash Contribution	1	Cash Contribution exchange type
Shipment	1	Sale exchange type
Paid Service Package	2	3D Printer Use exchange type
Rent payment	2	Rent exchange type
Pay maintenance	2	Maintenance Fee
Tech shop payment	2	Tech Shop Access exchange type
Membership Fee	2	FabLab Membership exchange type
Sponsorship	2	Fiscal Sponsorship exchange type
Pay Expense	2	Expense exchange type

You do them from the Admin, here, **Transfer Type**.

Contributors: Tibi, Emily McGill, Chris Troutner, …



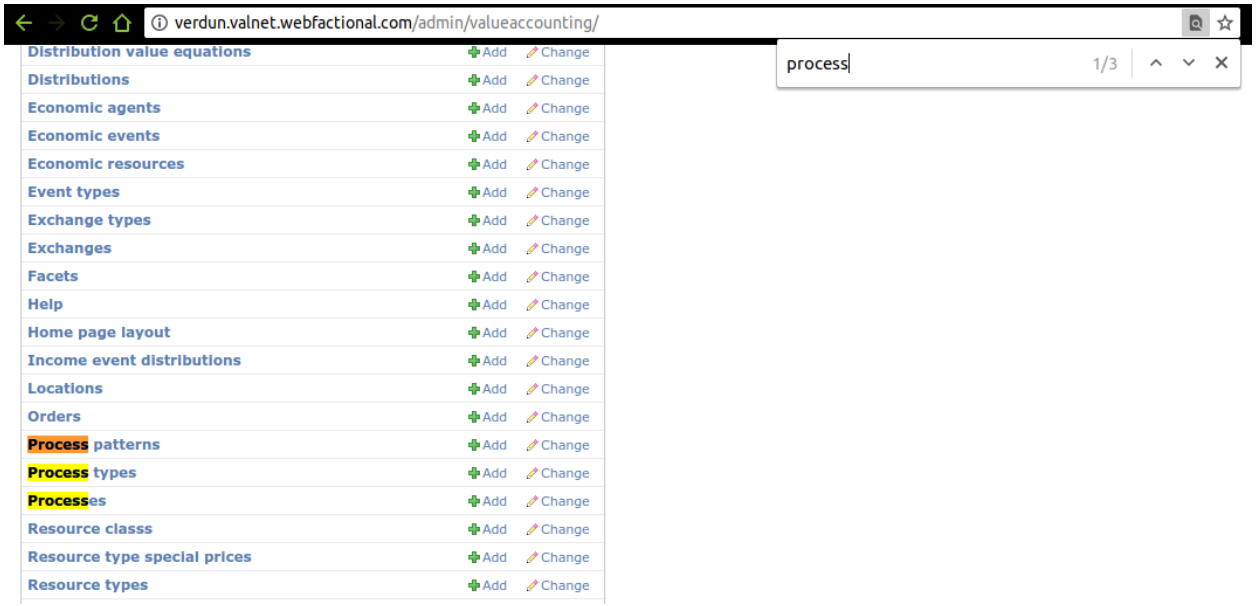
You need to properly set the parameters of these Transfer Types…



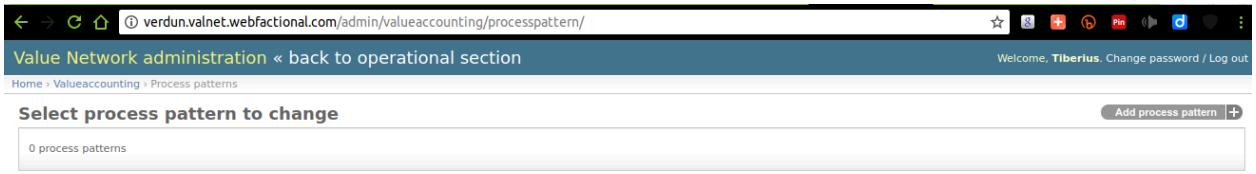
Create process patterns

You do that from Admin, here.

Contributors: Tibi, Emily McGill, Chris Troutner, …



They will end up here



There are some important process patterns that will be used in Recipes, like Change, Create changeable, .

Set **Exchanges**

…

Set **Process patterns**

…

Set **User cases** from admin

Use cases are predefined! They are linked to the UI and their names and identifiers are hard coded. Here's the list (copied below). Same for EventType, UseCaseEventType.

- *UseCase.create('cash_contr', _('Cash Contribution'), True)*
- *UseCase.create('non_prod', _('Non-production Logging'), True)*
- *UseCase.create('rand', _('Manufacturing Recipes/Logging'))*

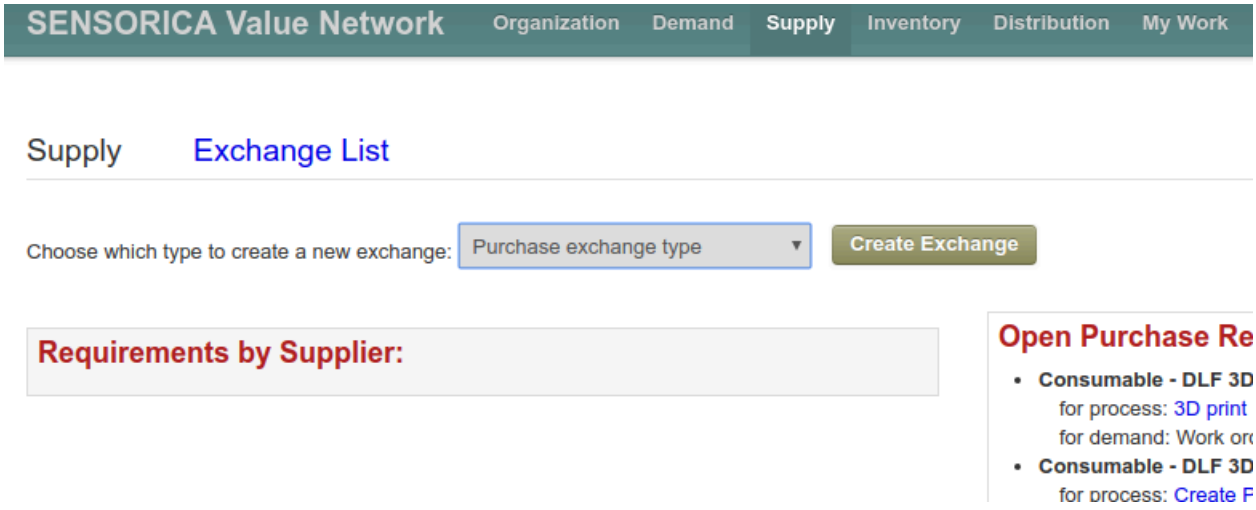
Contributors: Tibi, Emily McGill, Chris Troutner, …

- *UseCase.create('recipe', _('Workflow Recipes/Logging'))*
- *UseCase.create('todo', _('Todos'), True)*
- *UseCase.create('cust_orders', _('Customer Orders'))*
- *UseCase.create('purchasing', _('Purchasing'))*
- *UseCase.create('res_contr', _('Material Contribution'))*
- *UseCase.create('purch_contr', _('Purchase Contribution'))*
- *UseCase.create('exp_contr', _('Expense Contribution'), True)*
- *UseCase.create('sale', _('Sale'))*
- *UseCase.create('distribution', _('Distribution'), True)*
- *UseCase.create('val_equation', _('Value Equation'), True)*
- *UseCase.create('payout', _('Payout'), True)*
- *UseCase.create('transfer', _('Transfer'))*
- *UseCase.create('available', _('Make Available'), True)*
- *UseCase.create('intrnl_xfer', _('Internal Exchange'))*
- *UseCase.create('supply_xfer', _('Incoming Exchange'))*
- *UseCase.create('demand_xfer', _('Outgoing Exchange'))*
- *print "created use cases"*

Figuring out how to set up **Supply**

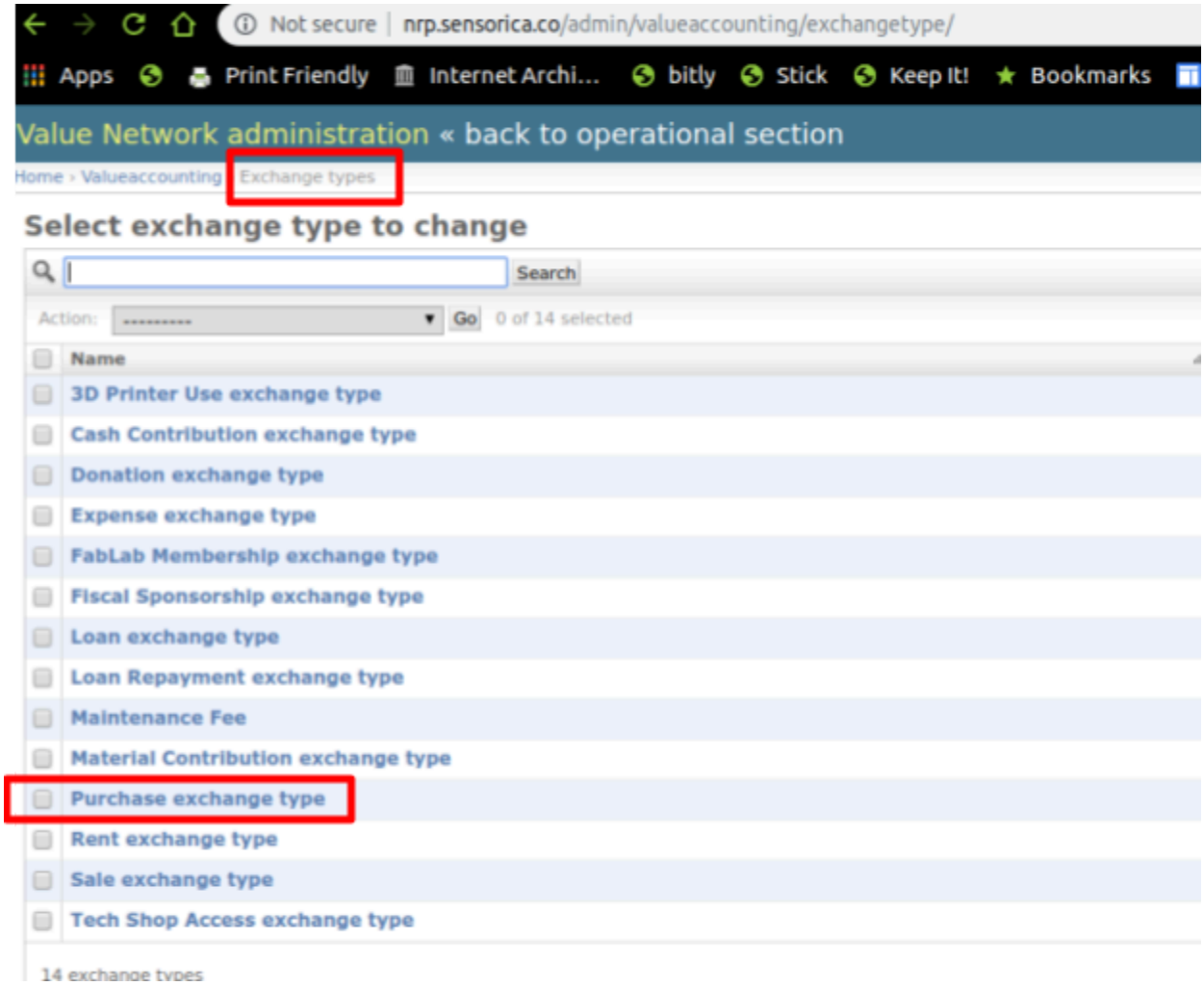
See tutorial about supply

Using the UI, you find Supply as a main tab. Creating a Supply event requires creating an *Exchange*, by choosing an *Exchange Type* from a dropdown list.

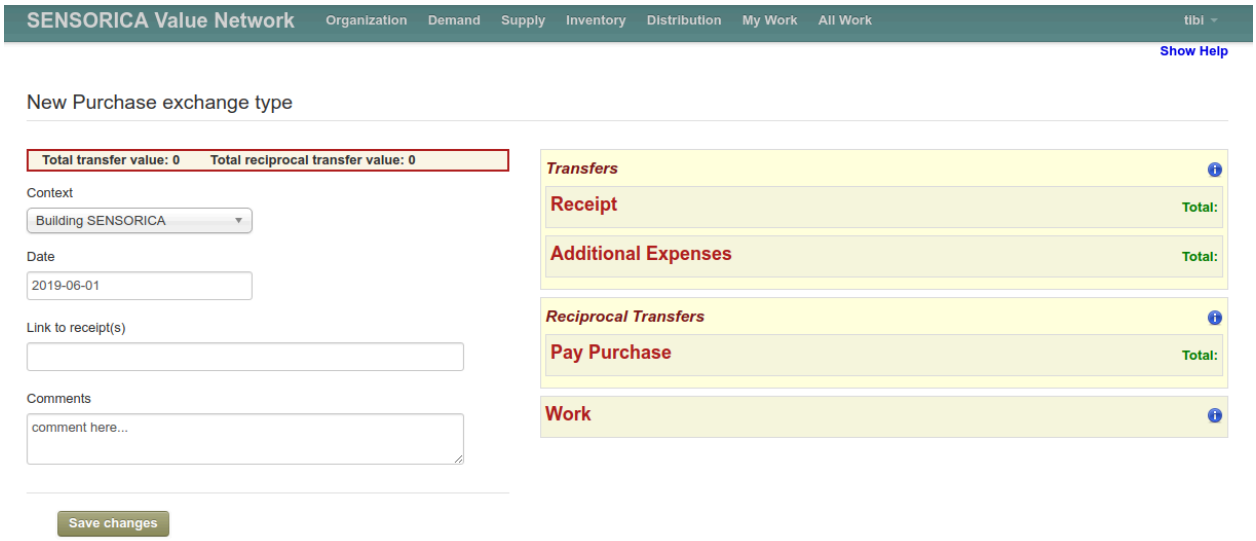


So setting this up requires setting up *Exchange Types* . You do that from Admin, like in the screenshot below, with the example of a Purchase Exchange Type. See other types of exchange in that list below, that you might need in your context.

Contributors: Tibi, Emily McGill, Chris Troutner, …

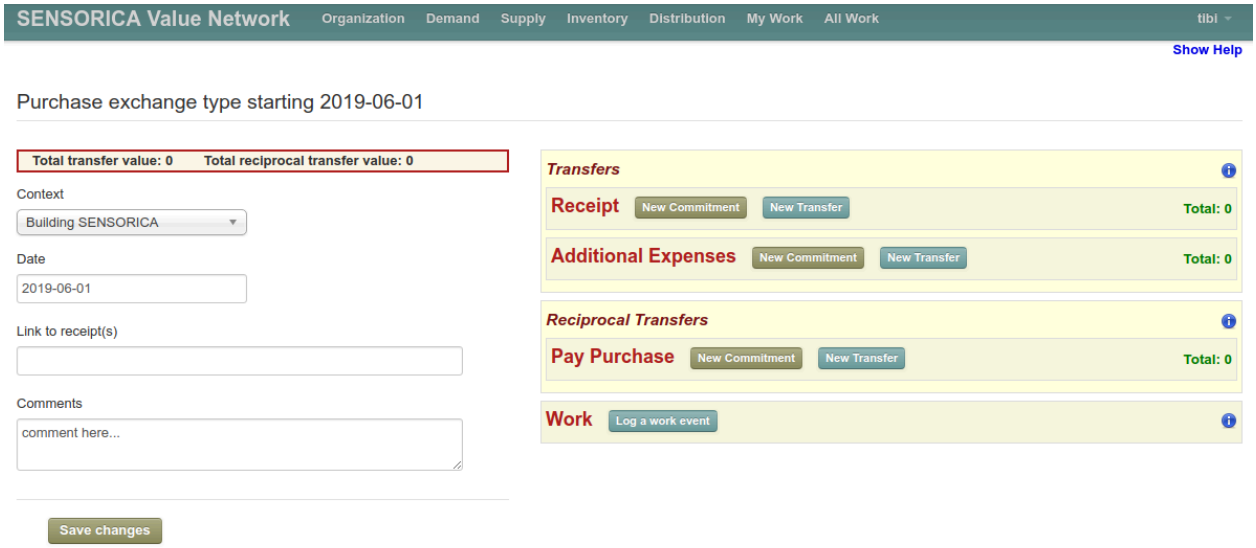


Once you create the Exchange, the following page will look like this (Sensorica case)



Contributors: Tibi, Emily McGill, Chris Troutner, …

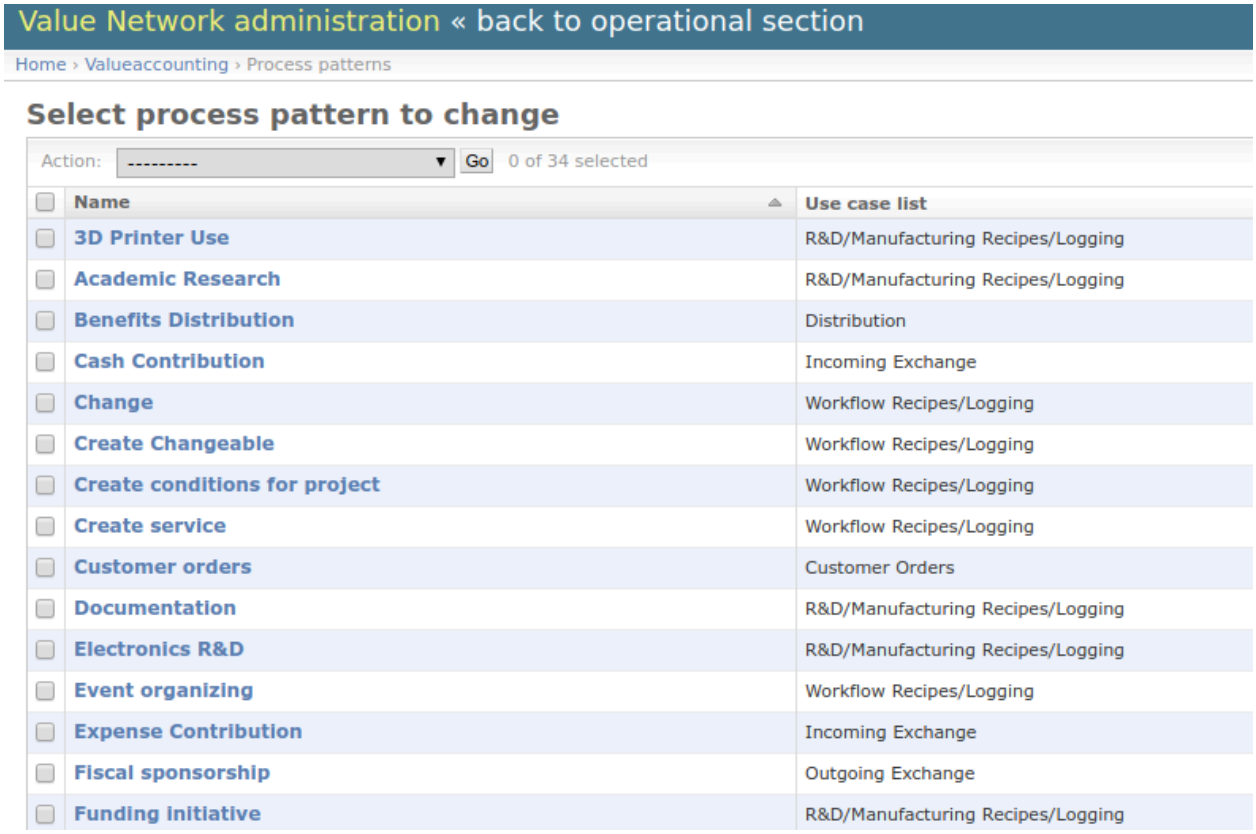
You need to set up the left side, the *exchange* information. The *transfer* information is situated on the right, see more explanation in Bob and Lyn's presentation. The
side (right side) only activates after you hit the *Save changes* button. The page should turn like this (Sensorica case).



In this example, *transfers* are: *Receipt*, *Additional Expenses*, *Pay Purchase* and *Work*. These are **Transfer Types**, which you need to set up prior to setting up Supply, see section above.

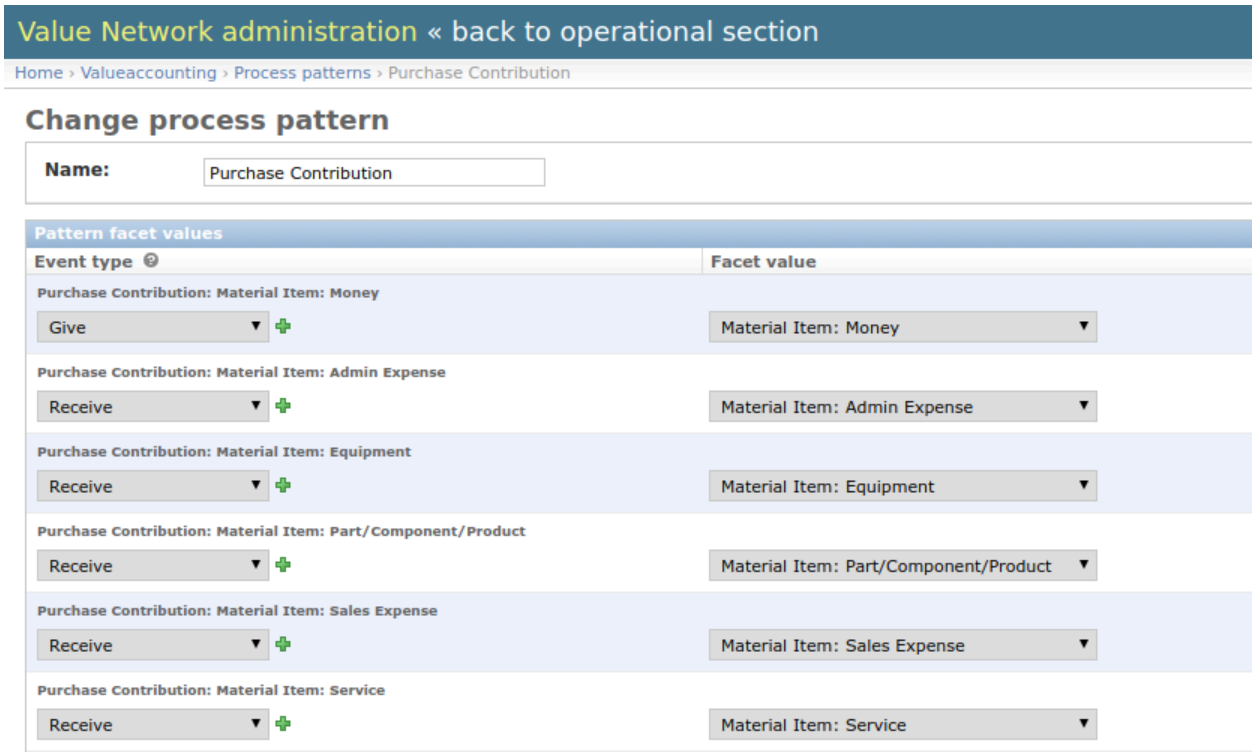
Like everything else, whatever you see on a page is governed by **Patterns**. So you need to have a pattern for every Supply case you have. In this example of Purchase, the **Process Pattern** is found in Admin The Page looks like this. Purchase, in this example, is low in the list, it doesn't appear in the picture.

Contributors: Tibi, Emily McGill, Chris Troutner, …



The page to edit a **Process Pattern** looks like this. Note that you need to create **Event Types** before, as well as **Facet Values**. See sections above.

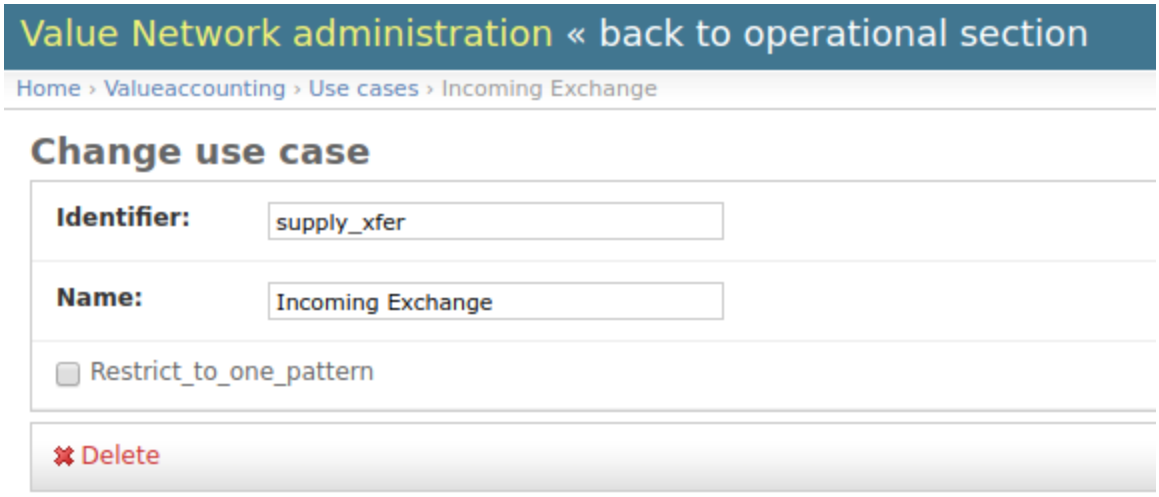
Contributors: Tibi, Emily McGill, Chris Troutner, …



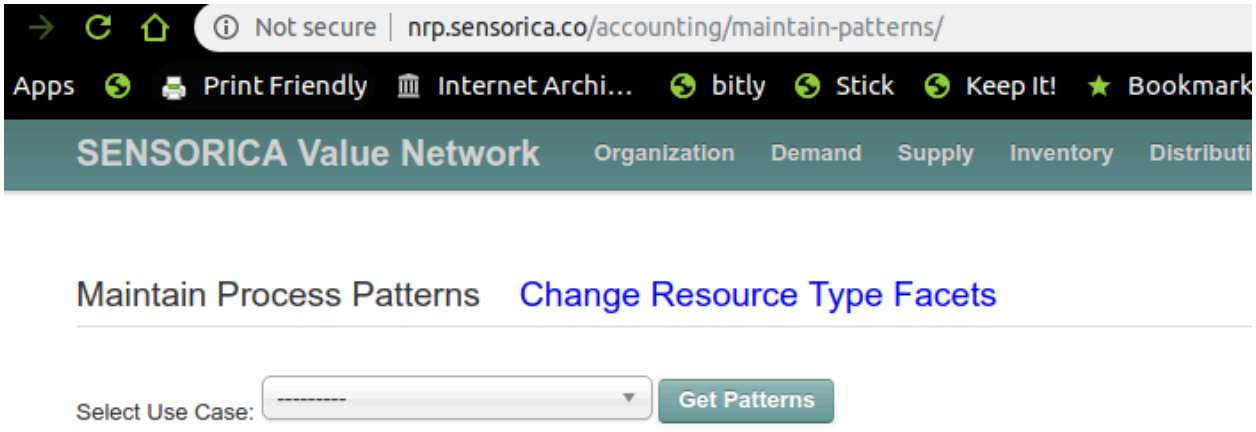
But Patterns depend on **Use cases**, which you set in Admin here.

Purchasing, in this example, relates to Incoming Exchanges, which are set up like this.

Contributors: Tibi, Emily McGill, Chris Troutner, …

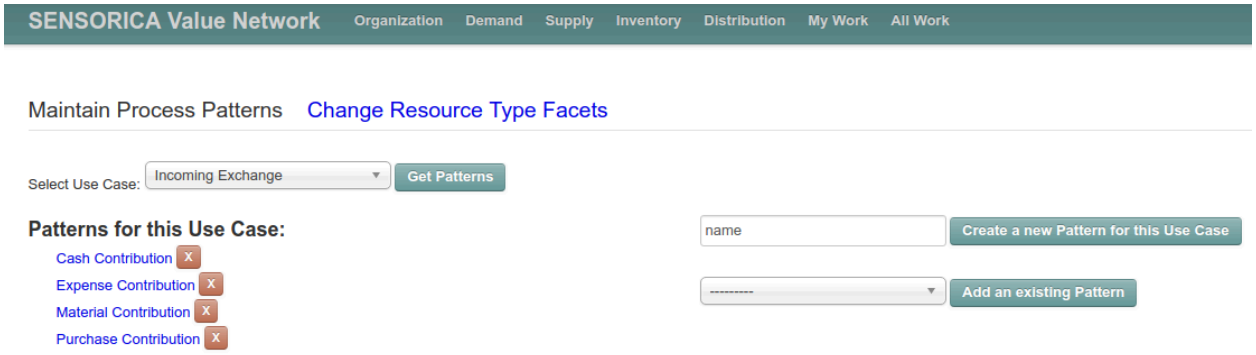


On the UI, you can tween Patterns here

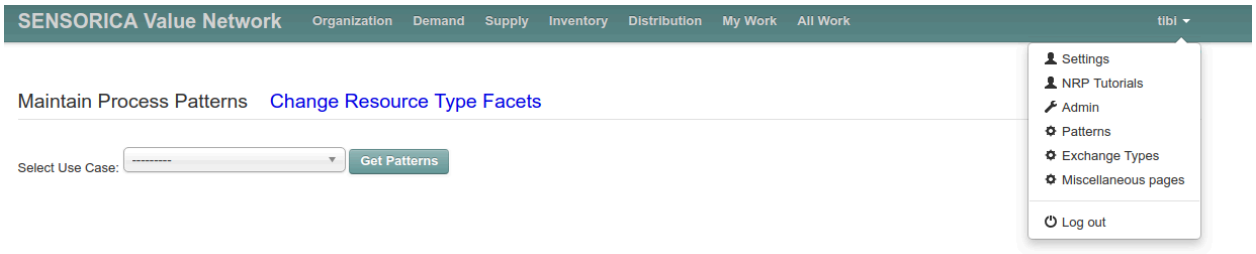


Selecting a Usecase, brings all the Patterns relevant to it. For example, selecting Incoming Exchanges you should get Purchase Contributions. If you click on Purchase Contributions, you open a page where you can tweak it, playing with Pattern Facets.

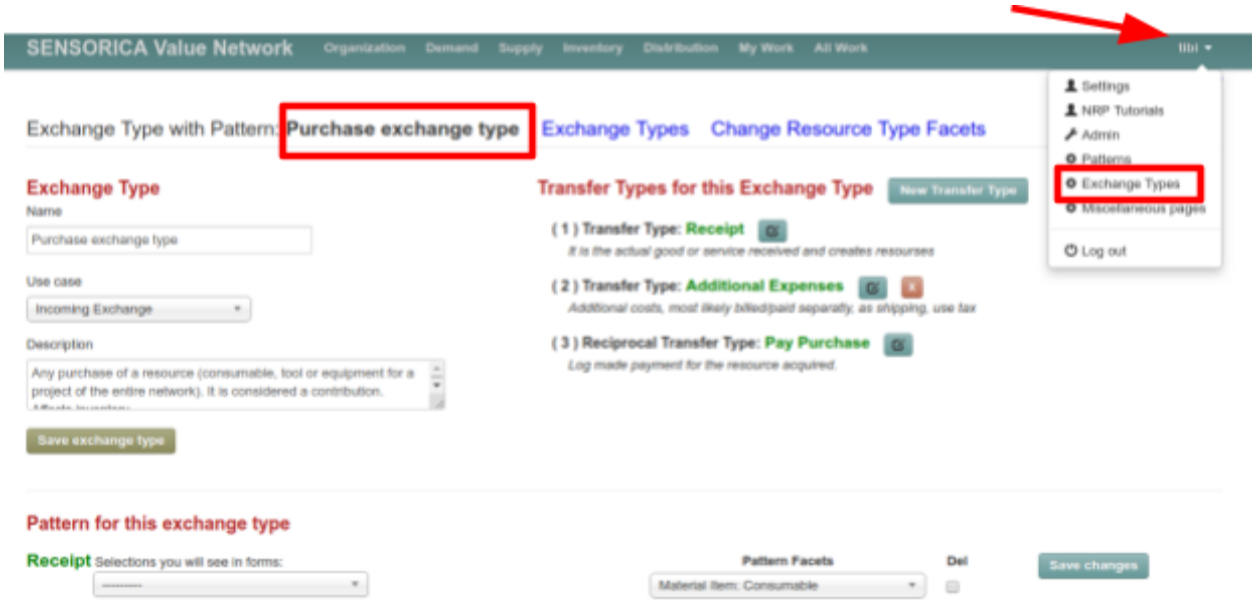
Contributors: Tibi, Emily McGill, Chris Troutner, …



You get there from your user menu, see *Patterns* on the right side of the pic below.



In the end, you can configure your Exchange Types from this page



Contributors: Tibi, Emily McGill, Chris Troutner, …