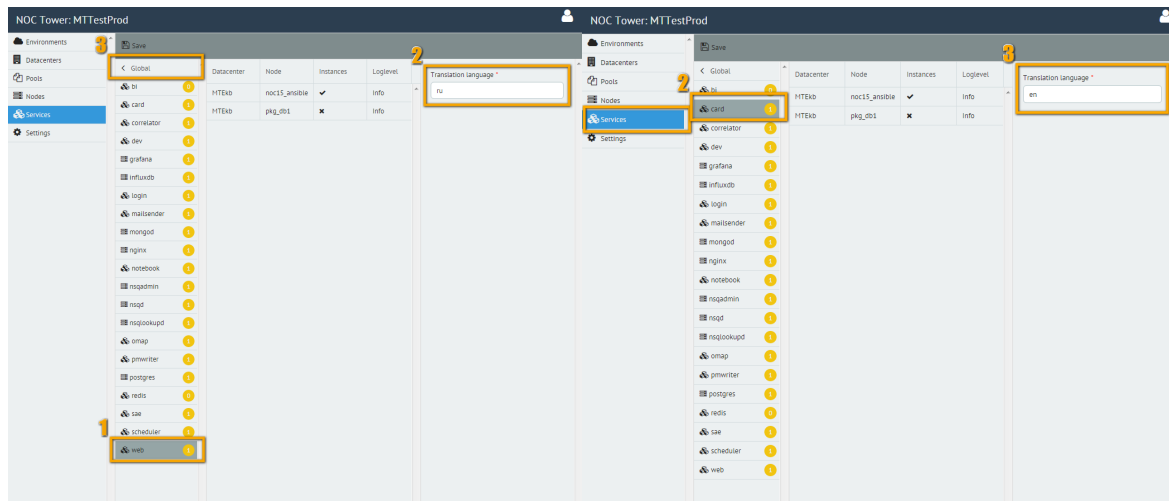


Локализуем, переводим.

Общее.

В версии Микросервисы появилась возможность локализации (перевода) НОКа на другие языки. Это осуществляется при помощи модуля GetText, который считывает подготовленные строки локализации и заменяет их на соответствующий перевод по словарю. Поскольку, работает он при запуске системы перевод не может изменяться на лету (настройкой браузера) и требует изменения настроек и перезапуска НОКа.

На текущий момент переводу поддаётся 3 сервиса: Web, Card и Login. Механизм локализации описания аварий не реализован. Перевод можно выбрать в настройках соответствующего сервиса в башне:



После этого необходимо произвести Deploy с отмеченной опцией **"Install Everything"** или **"Update config"** и **"Restart gentle"**.

Сами переводы расположены в папке **translations** соответствующего сервиса:

- service/web/translations для web
- service/card/translations для card
- service/login/translations для login

Название папки с переводом необходимо прописывать в настройках сервиса. Можно создавать свои переводы и располагать их в папках по соседству.

Также, для перевода потребуется установить пакеты для разработки:

```
./bin/pip install -r requirements/dev.txt
```

И создать файл конфигурации: **etc/babel.cfg**

```
#
# Translation extraction config
#
[python: **.py]

[javascript: **.js]
extract_messages = _, __

[jinja2: **.j2]
extensions=jinja2.ext.autoescape,jinja2.ext.with_
```

Перевод.

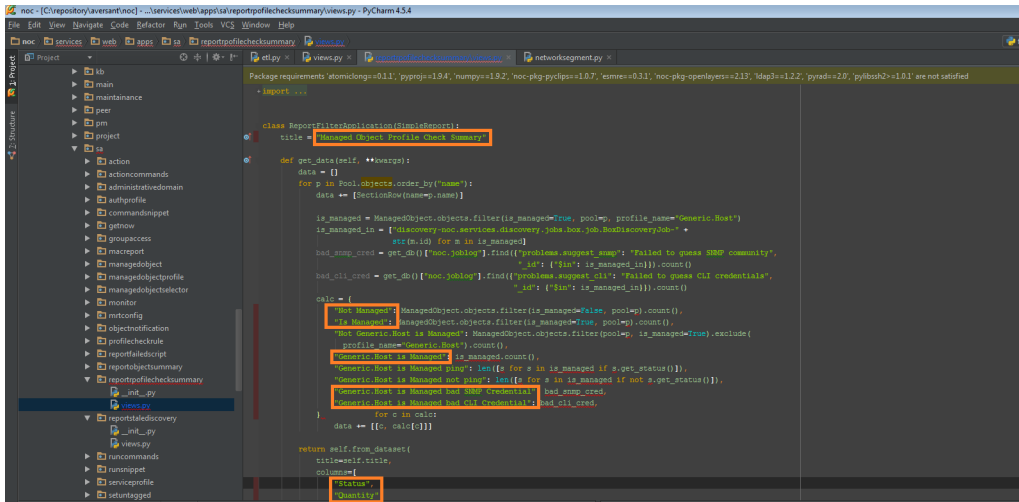
Сам по себе процесс перевода состоит из 3 шагов:

1. Разметка в исходном коде фраз, которые поддаются под перевод.
2. Создание/обновление словаря для перевода.
3. Компиляция переведённого текста.

Пройдёмся по пунктам по порядку. Если вам необходимо исправить существующий перевод, можно переходить к пункту [Создание или обновление словаря](#).

Разметка фраз для перевода

В первую очередь необходимо пометить фразы, как доступные для перевода. Это делается в исходном коде приложения и может быть в 2 вариантах: разметка для python файлов и для JS файлов. Покажем на примере, как делается разметка в Python файлах. Возьмём, для примера, подготовленный отчёт. В нём присутствуют текстовые строки (помечены оранжевым квадратом) - их необходимо пометить для перевода. Для этого необходимо импортировать пакет gettext (`from noc.core.translation import ugettext as _`) и заключить текст в круглые скобки: `_(translate_text)` - см. второй скриншот.



```
class ReportFilterApplication(SimpleReport):
    title = _("Managed Object Profile Check Summary")

    def get_data(self, **kwargs):
        data = []
        for p in Pool.objects.order_by('name'):
            data.append([SectionRow(name=p.name)

            is_managed = ManagedObject.objects.filter(is_managed=True, pool=p, profile_name="Generic Host")
            is_managed_in = [{"discovery-noc-services-discovery-job-id": "noc-job-80814109"}]
            etc(m.id) for m in is_managed

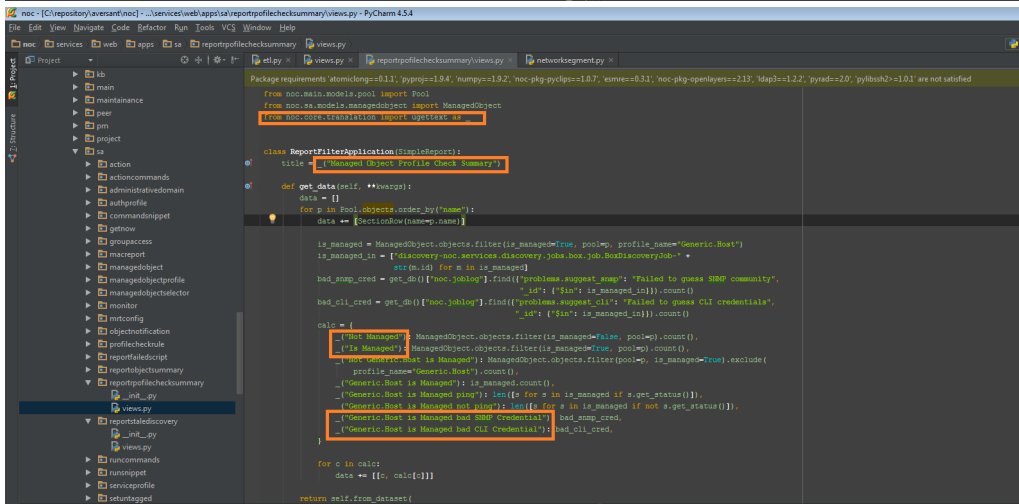
            bad_smp_cred = get_db()[["noc-joblog"].find({"problems.suggest_smp": "Failed to guess SMP community",
                                                    '_id': {'$in': is_managed_in}}).count()

            bad_cli_cred = get_db()[["noc-joblog"].find({"problems.suggest_cli": "Failed to guess CLI credentials",
                                                    '_id': {'$in': is_managed_in}}).count()

            calc = {
                "Not Managed": ManagedObject.objects.filter(is_managed=False, pool=p).count(),
                "Is Managed": ManagedObject.objects.filter(is_managed=True, pool=p).count(),
                "Not Generic Host Is Managed": ManagedObject.objects.filter(pool=p, is_managed=True).exclude(
                    profile_name="Generic Host").count(),
                "Generic Host Is Managed": is_managed.count(),
                "Generic Host Is Managed ping": len([s for s in is_managed if s.get_status()]),
                "Generic Host Is Managed bad ping": len([s for s in is_managed if not s.get_status()]),
                "Generic Host Is Managed bad smp community": bad_smp_cred,
                "Generic Host Is Managed bad CLI Credentials": bad_cli_cred,
            }

            for c in calc:
                data.append([c, calc[c]])

        return self.from_dataset(
            title=self.title,
            columns=[
                "status",
                "value"
            ]
        )
```



```
from noc.main.models.pool import Pool
from noc.sa.models.managedobject import ManagedObject
from noc.core.translation import ugettext as _

class ReportFilterApplication(SimpleReport):
    title = _("Managed Object Profile Check Summary")

    def get_data(self, **kwargs):
        data = []
        for p in Pool.objects.order_by('name'):
            data.append([SectionRow(name=p.name)

            is_managed = ManagedObject.objects.filter(is_managed=True, pool=p, profile_name="Generic Host")
            is_managed_in = [{"discovery-noc-services-discovery-job-id": "noc-job-80814109"}]
            etc(m.id) for m in is_managed

            bad_smp_cred = get_db()[["noc-joblog"].find({"problems.suggest_smp": "Failed to guess SMP community",
                                                    '_id': {'$in': is_managed_in}}).count()

            bad_cli_cred = get_db()[["noc-joblog"].find({"problems.suggest_cli": "Failed to guess CLI credentials",
                                                    '_id': {'$in': is_managed_in}}).count()

            calc = {
                "Not Managed": ManagedObject.objects.filter(is_managed=False, pool=p).count(),
                "Is Managed": ManagedObject.objects.filter(is_managed=True, pool=p).count(),
                "Not Generic Host Is Managed": ManagedObject.objects.filter(pool=p, is_managed=True).exclude(
                    profile_name="Generic Host").count(),
                "Generic Host Is Managed": is_managed.count(),
                "Generic Host Is Managed ping": len([s for s in is_managed if s.get_status()]),
                "Generic Host Is Managed bad ping": len([s for s in is_managed if not s.get_status()]),
                "Generic Host Is Managed bad smp community": bad_smp_cred,
                "Generic Host Is Managed bad CLI Credentials": bad_cli_cred,
            }

            for c in calc:
                data.append([c, calc[c]])

        return self.from_dataset(
```

Таким образом можно размечать текст в уже существующих файлах (если потребуется что-то доперевести).

В случае JS (JavaScript) файлов процедура совпадает, но необходимо использовать двойное подчёркивание перед скобками вместо одинарного: `__()`

```
test: {<code>{</code>},
dataIndex: '<code>severity</code>',
width: 100,
renderer: BOC.renderer.DateLine
},
test: {<code>{</code>},
dataIndex: '<code>severity</code>',
width: 75,
renderer: BOC.renderer.Lookup('<code>severity</code>')
},
test: {<code>{</code>},
dataIndex: '<code>alarm_class</code>',
width: 350,
renderer: BOC.renderer.Lookup('<code>alarm_class</code>')
},
test: {<code>{</code>},
dataIndex: '<code>severity</code>',
width: 1,
flex: 1
},
test: {<code>{</code>},
dataIndex: '<code>severity</code>',
width: 75,
align: 'right',
renderer: BOC.renderer.Duration
},
test: {<code>{</code>},
dataIndex: '<code>severity</code>',
width: 35,
align: 'right'
}
```

После разметки фраз необходимо закоммитить изменения (команда **hg commit**)



Обязательно выполните команду "**hg commit**" после того как разметили файлы для перевода, иначе не выполнится операция извлечения строк в словарь.

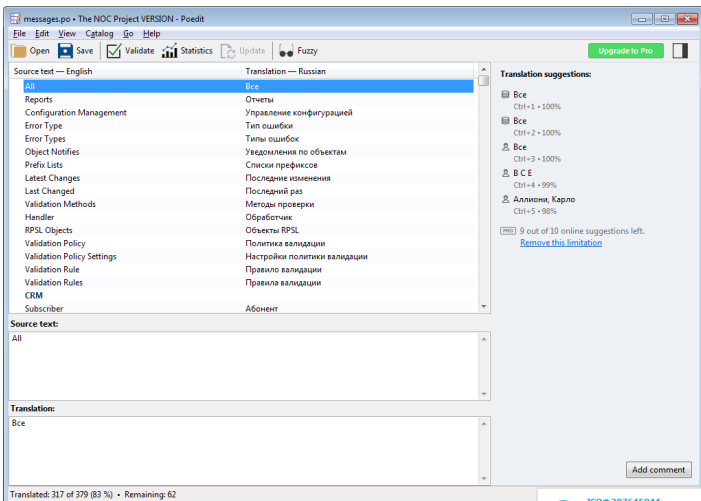
Создание или обновление словаря для перевода

После того как произведена разметка текста для перевода необходимо извлечь фразы из исходного кода и поместить их в словарь. Извлечение исходного текста делается командой **./noc translation extract <service>**. **<service>** опционален, если не указать, то извлекутся строки для всех сервисов. После извлечения необходимо обновить словарь командой **./noc translation update <service>**.


```
root@root:~# cat /etc/passwd | grep root | sed 's/://;s/ /_/' | xargs -r curl -s -u 'root:root' -H 'Content-Type: application/json' -X POST -d '{\"message\": \"\"}' --url http://10.10.10.10:8080/api/translations/messages.js
root@root:~# cd /root/.local/share/poedit/
root@root:~# find . -type f -name '*.po' | xargs -r cp -v /root/.local/share/poedit/translations/
root@root:~# cd /root/.local/share/poedit/translations/
root@root:~# ls -la
total 16
drwxr-xr-x 2 root root 4096 Oct 26 11:11 .
drwxr-xr-x 3 root root 4096 Oct 26 11:11 ..
-rw-r--r-- 1 root root 1100 Oct 26 11:11 messages.js.po
-rw-r--r-- 1 root root 1100 Oct 26 11:11 messages.py.po
root@root:~#
```

```
root@root:~# find . -type f -name '*.po' | xargs -r cp -v /root/.local/share/poedit/translations/
root@root:~# cd /root/.local/share/poedit/translations/
root@root:~# ls -la
total 16
drwxr-xr-x 2 root root 4096 Oct 26 11:11 .
drwxr-xr-x 3 root root 4096 Oct 26 11:11 ..
-rw-r--r-- 1 root root 1100 Oct 26 11:11 messages.js.po
-rw-r--r-- 1 root root 1100 Oct 26 11:11 messages.py.po
root@root:~#
```

После выполнения последней команды мы получим словарь с нашими фразами, которые мы можем переводить (**.po файл**). Эти файлы лежат в папках переводов (н-р для русского это **ru/LC_MESSAGES/messages.po** для python и **ru/LC_MESSAGES/messages_js.po** для JS). Их можно свободно скачать к себе и заняться переводом. Это можно делать в текстовом редакторе, либо, используя специальный редактор переводов (н-р **PoEdit**).



После заполнения всех полей файл необходимо вернуть обратно.

 **Внимательно!** не перепутайте сервис, для которого осуществлялся перевод!

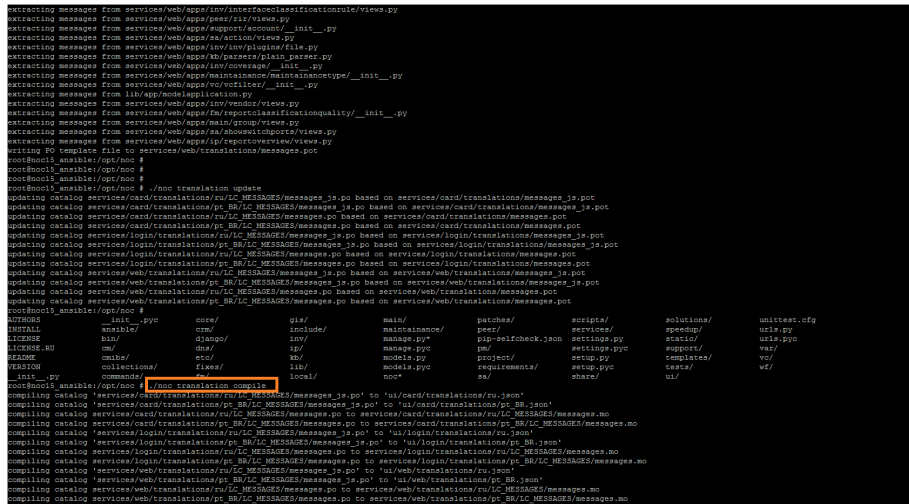
Файлки, для остальных сервисов можно не трогать (в них поменялась только дата обновления)

Итого.

```
# ./noc translation extract <service>
# ./noc translation update <service>
# ./noc translation edit <service> <lang> ( )
```

Компиляция переведённого текста для использования НОКом

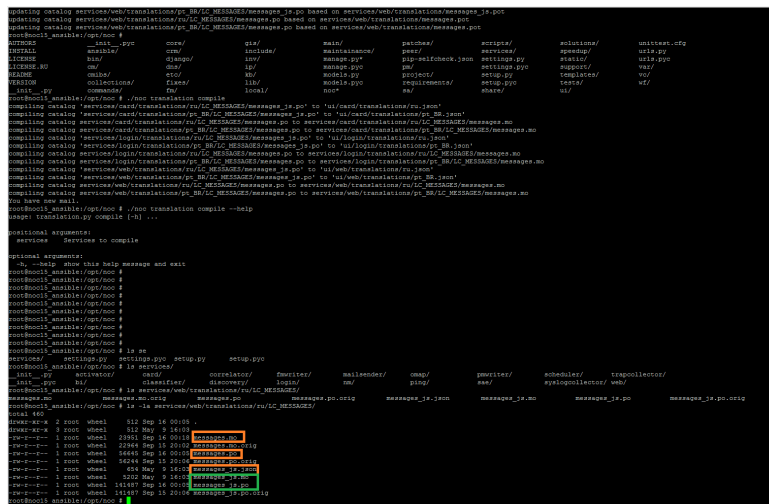
После перевода фраз, словарь необходимо вернуть в папку LC_MESSAGES для соответствующего языка и сервиса. После этого необходима компиляция: выполняется командой `./noc translation compile <service>`.



После компиляции необходим перезапуск процесса, для которого осуществлялся перевод (web, card или login).

Публикация перевода

В случае, если необходимо вместе с наработками отправить перевод или свои правки - это можно сделать отправив 3 файла для соответствующего сервиса:



- словарь: .po файл
- скомпилированный словарь: .mo файл
- json файл с описанием: messages_js.json



Отправлять необходимо файлы, в которых изменилось содержимое (добавились строки или перевод). Файлы, где поменялась только дата прикладываться смысла нет.

Проблемы

```
./noc translate extract "No module named babel.util"
```

```
]# ./noc translation extract
Traceback (most recent call last):
  File "./commands/translation.py", line 15, in <module>
    from babel.util import pathmatch
ImportError: No module named babel.util
```

Необходимо установить зависимости для разработки: `./bin/pip install -r requirements/dev.txt`