omnileads Documentation

Release develop

omnileads

Feb 07, 2020

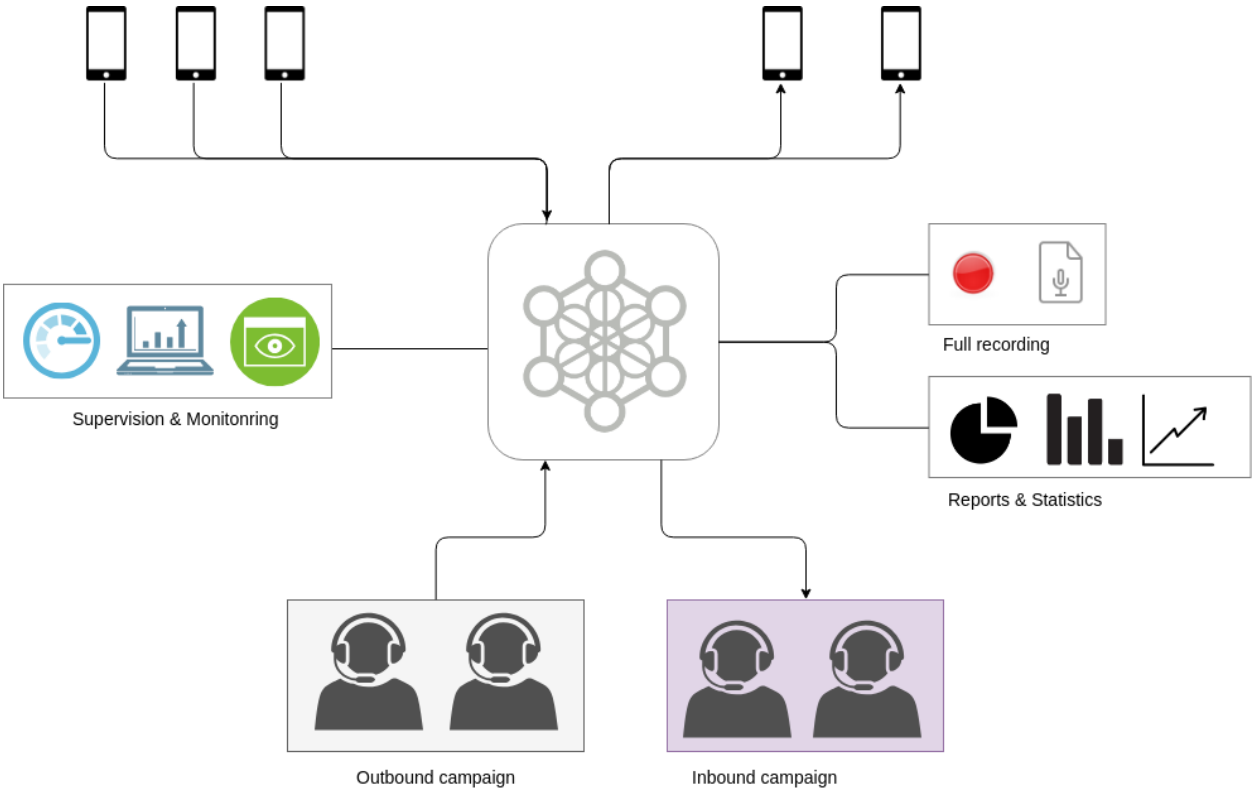
Contents

OMniLeads is a GPLV3 licenced software made for Contact Centers. Based on WebRTC , it allows to implement and manage a Contact Center, administering inbound and outbound operations, with access to metrics, reports and indicators, real-time supervision of agents and advanced QA functionalities of contacts and campaigns.

You just need to execute a simple *OMniLeads installation*, and in a few configuration steps, you will have a functional OMniLeads instance with Contact Center operations of inbound and outbound campaigns.

Our software can be adapted to a company or organization that needs to maintain its own Contact Center with integration to its PBX, also it can work as a "core" of communications to a company that provide Customer Contact (Business Outsourcing Process - BPO) services or it can be installed in VPS or Cloud instances.

OMniLeads is 100% web for the different user profiles: agents, supervisors, administrators, customers, that can access the software from every modern browser with WebRTC support. It doesn't require the use of desktop applications like softphones. This simplifies to zero the configuration to do in the workstations of agents, that will only have to go to a web URL to be online, managing communications with customers. OMniLeads is a great option to implement CCaaS "Contact Center as a Service" !



CHAPTER 1

How do i get it?

Our repository is available on Gitlab, the software is free for download, install and use

CHAPTER 2

Where can I install it ?

OMniLeads puede correr como una aplicación tradicional desplegando una instalación de todos los componentes sobre un server físico, máquina virtual o VPS. Siempre y cuando se utilice como base GNU/Linux: CentOS minimal 7.7

- CentOS 7.7.1908 minimal ISO
- Docker

OMniLeads can also be deployed using Docker containers, with this form the software can be also installed inside an Issabel-PBX or FreePBX instances, having both Contact Center and PBX solutions inside the same host.

CHAPTER 3

How do I install it ?

The section *OMniLeads installation* talk about this, exposing the steps needed to install the software.

CHAPTER 4

Omnileads's Features and funtionalities

- about_webrtc.
- about_omlfeatures.

CHAPTER 5

Where and how can I use it ?

- about_usecase_pbx.
- about_usecase_bpo.
- about_usecase_cloud.

CHAPTER 6

How do I get training?

This documentation take cover of all the facts of this product. From technical instructions oriented to IT administrators to instructions about functionality oriented to supervisors or leaders of the Contact Center.

Also a **Free Online Course** is available. In this course you can see videos that covers all thematic in this documentation.

To register **free** go to *this link <http://www.techxpert.guru/omnileads-takeoff/>*

CHAPTER 7

INSTALLATION

In this chapter is covered all the types of installation of the software

7.1 OMniLeads installation

En este capítulo se detallan los tipos de instalación de OMniLeads

- about_install_selfhosted.
- about_install_remote.
- about_install_docker_linux.
- about_install_docker_pbx.

El siguiente apartado esta dirigido a usuarios avanzados de Docker, los cuales quieran involucrarse mas con la creacion de imagenes para el proyecto:

- about_install_docker_build.

Note: Recommendations:

- Both the host and the node to be installed must have a good and stable internet connection
- That there is no network element to go out to the internet (firewall blocking 443 port, proxy)
- Use the isos that are recommended in the section Where can you install?
- In case of failure of any ansible task run the installation script again
- In case it fails again raise an issue to https://gitlab.com/omnileads/ominicontacto/issues specifying the Linux distro in which it happened and the stable version that was wanted to install

---

CHAPTER 8

INITIAL SETTINGS

In this section the essential configuration are shown once an instance of OMniLeads is installed

8.1 Initial configuration

Our platform is installed and therefore available to start setting up all needed to assemble our Contact Center operation. In this section we are going to move on with the initial steps like create our first supervisor user, generate agent groups and their pauses, to finally conclude the chapter with an agent login.

8.1.1 Instance registry

This is not mandatory, the system will work without registration. Anyway is mandatory to register the instance if you want to buy an Addon or subscribe the platform the Freetech Solutions support.

Finnaly, for that certified partners (with official OMniLeads certification done), by registrying the instance the installation is *signed* with the certification code that shows that the instance was installed by a certified person.

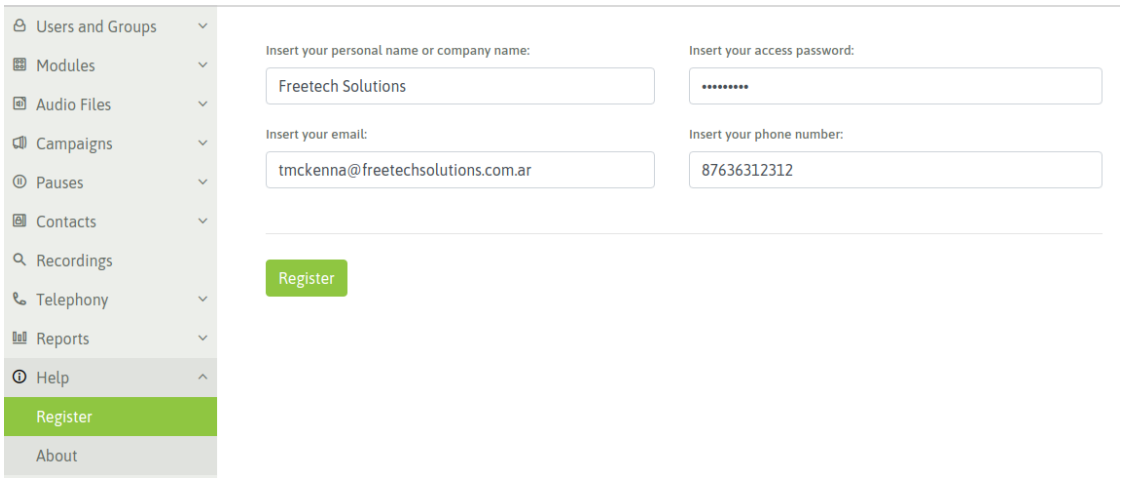


Figure 1: omnileads registration

The fields here must be filled after that you will receive an email with the instance key

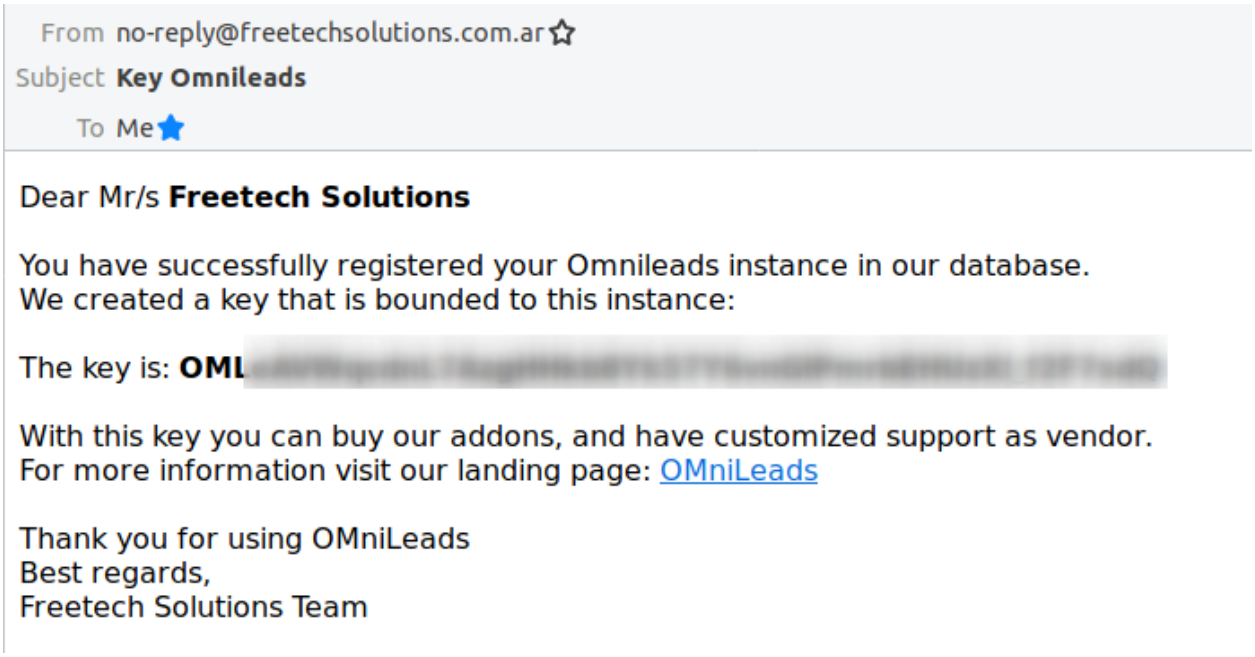


Figure 2: Email notification

After that, every time you go to Register section you will see this.

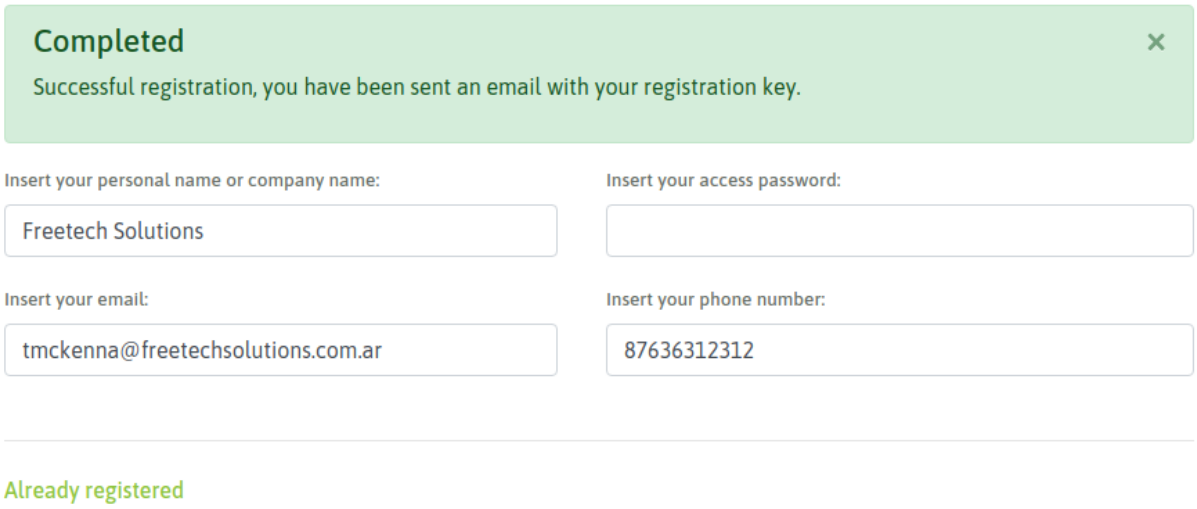


Figure 3: omnileads registered

8.1.2 Create supervisor user

The first thing we are going to do is generate a supervisor user with whom from now on we are going to do all the administrative tasks of the system, leaving the "admin" user relegated as a last access resourse and for the IT staff.

To generate a new user we must access the menu point *User list -> New user*

There we complete the requested fields as illustrated in the figure 1.

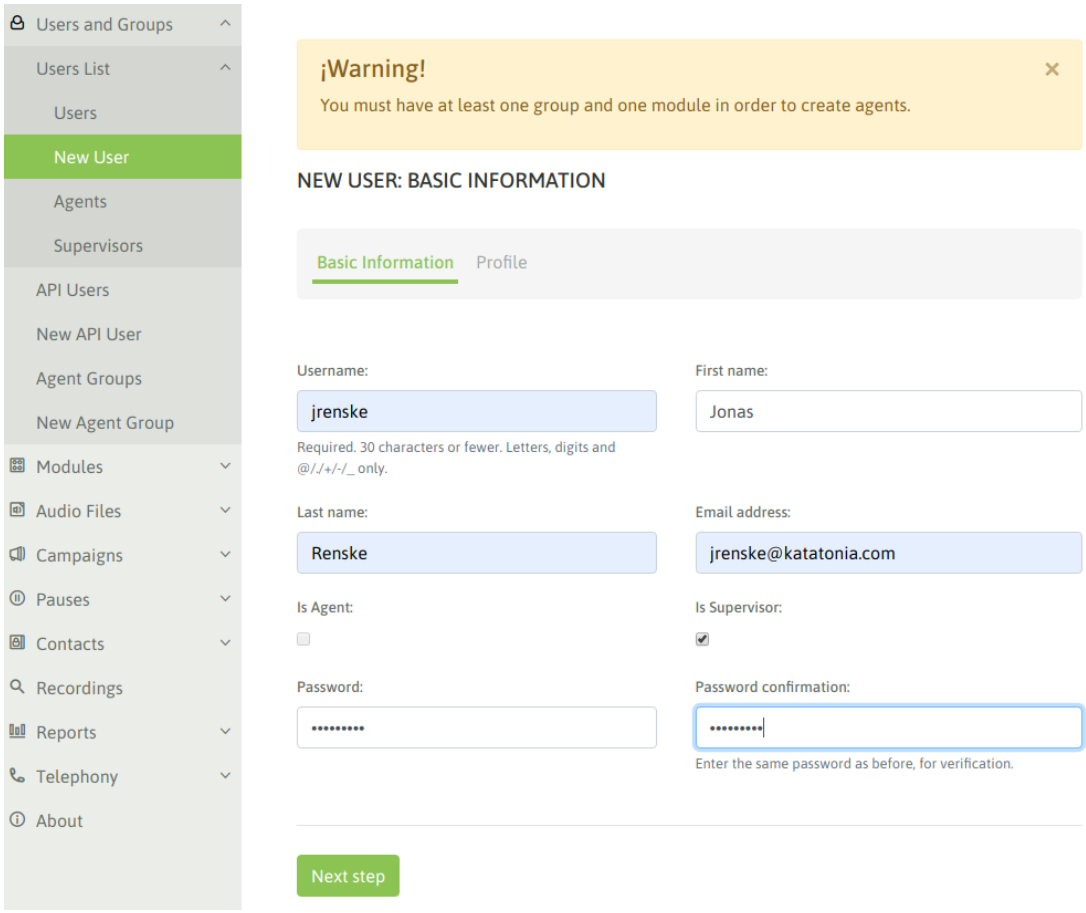


Figure 4: new supervisor user

In the next step of the wizard we must select the supervisor profile, being possible to select:

- **Supervisor manager:** this user profile can manage almost all the platform menus except for the configurations of telephony parameters like for example SIP trunks and calls routing.
- **Administrator:** is a super-user who owns the same permissions than the “admin” user
- **Customer:** is a profile that can only access to the real time supervision and call recordings about the campaings that were assigned as allowed.

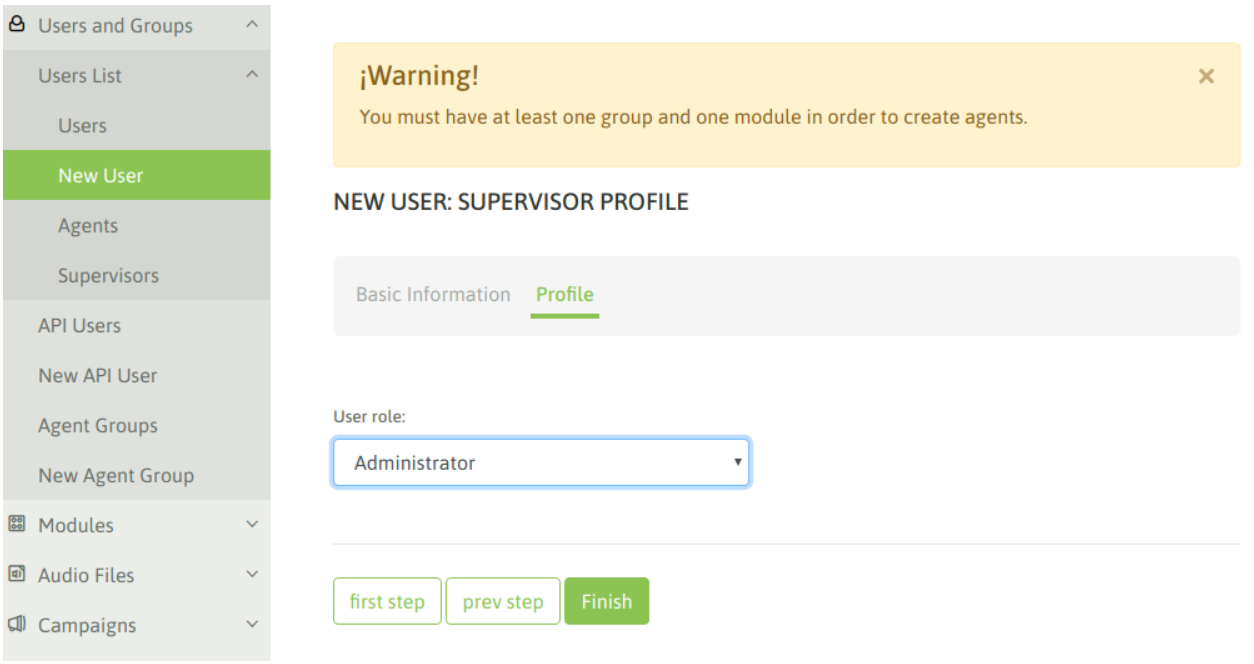


Figure 5: supervisor / admin

In our case we generate a supervisor / administrator user. Then we log in with that user to continue the managements.

8.1.3 Create telephony module

In this point each system module must be generated, telephony functionality is currently available, so we are going to add that module. However towards the future SMS and Chat would be added

To generate the telephony module we must access to *Modules -> Enable module*

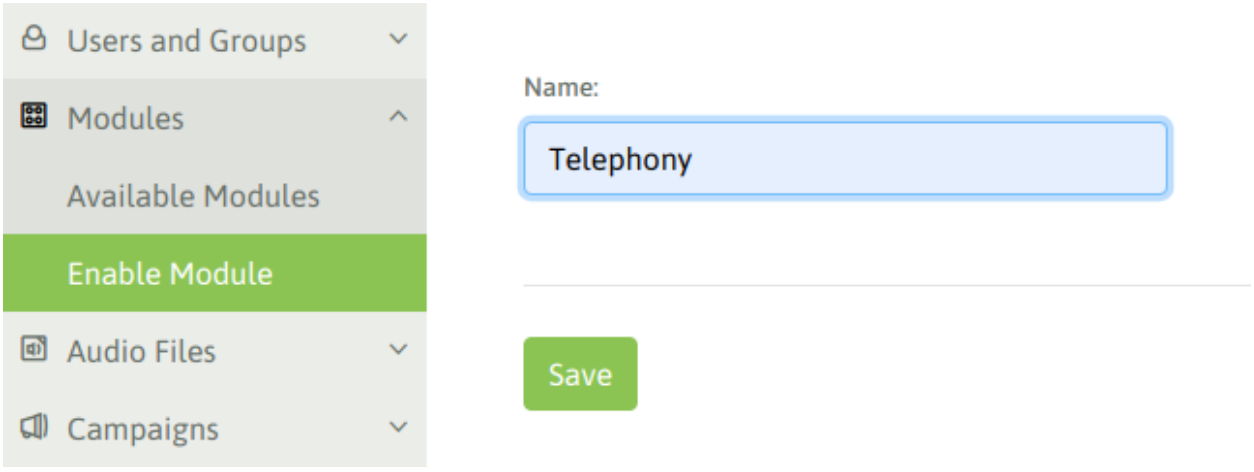


Figure 6: Enable telephony module

We save the changes and move on !

8.1.4 Extra language installation

The generic audios that the agents or external phones will hear will be in English language, these audios are configurable in the inbound or outbound routes, so that way the channel will handle the language especified in its configuration.

If the instance needs other languages, you can install the package of audios with the "Telephony-Add audio files" menu, where you can install these languages:

- Spanish
- Italian
- Swedish
- Portuguese
- French
- Russian

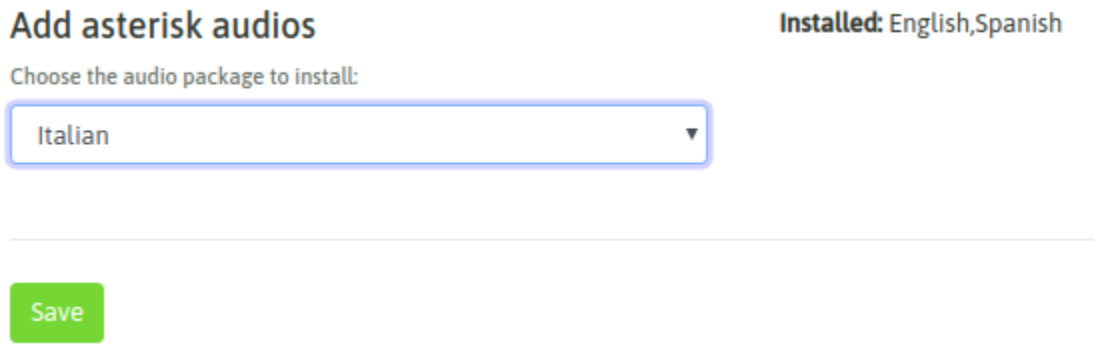


Figure 7: i18n audio promps

8.1.5 Create agents group

It is time to create the agents groups of our contact center. To generate a new group *Users and groups -> New agents group* and a form similar to the one in Figure 4 will be displayed.

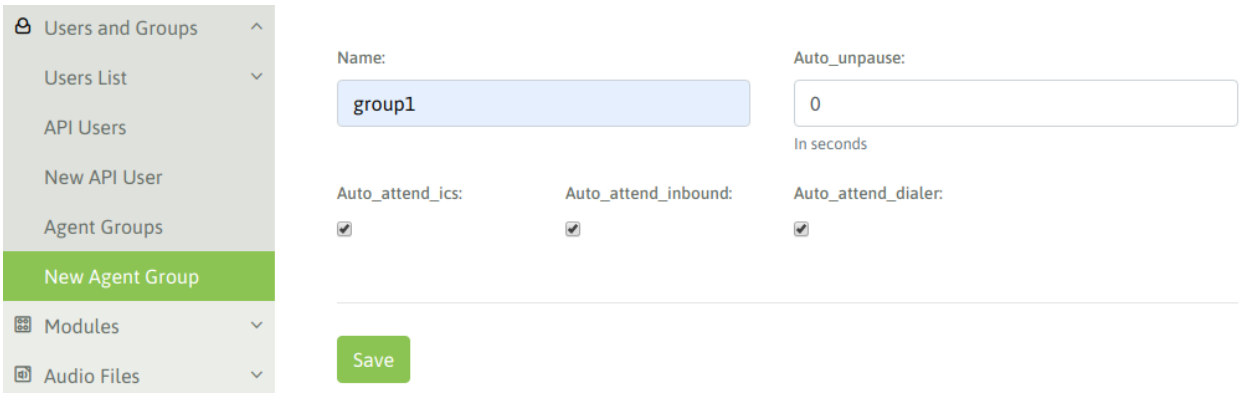


Figure 8: New agent group

The fields displayed there are:

- **Name:* is the name desired to assign to the agents group.

- **Auto_unpause:** to understand this parameter we must explain that in OMniLeads after every call processed by an agent (of any nature), the agent is forced to enter an ACW pause (after call work), in which remains inactive for the assigned campaings so he can complete the qualification of the current call and continue operating. Now, to get out of this pause induced by the system there are two possibilities and there our parameter enters, since on the one hand if we leave the value in "0" the agent must get out of the pause, now if we put a number (for example 5 seconds), this implies that the assigned agent to this group after falling into an induced pause ACW, the system will leave it online again after X seconds (as indicated in this parameter).
- **Auto_attend_ics:** if this value is checked then the calls coming from the OMniLeads ICS (Interactive Contact System) Addon, calls derived from the IVR blaster press-one will be automatically answered by the agent without providing the possibility of a ring on his webphone.
- **Auto_attend_inbound:** if this value is checked then the calls coming from incoming campaigns will be connected to the agent without providing the possibility of a ring and answer by the agent.
- **Auto_attend_dialer:** if this value is checked then the calls coming from campaigns with predictive dialer will be conected to the agent without providing the possibility of a ring and answer by the agent.

8.1.6 Create agents

Once the agents groups and their configuration are available, we can proceed with the creation of contact center agents.

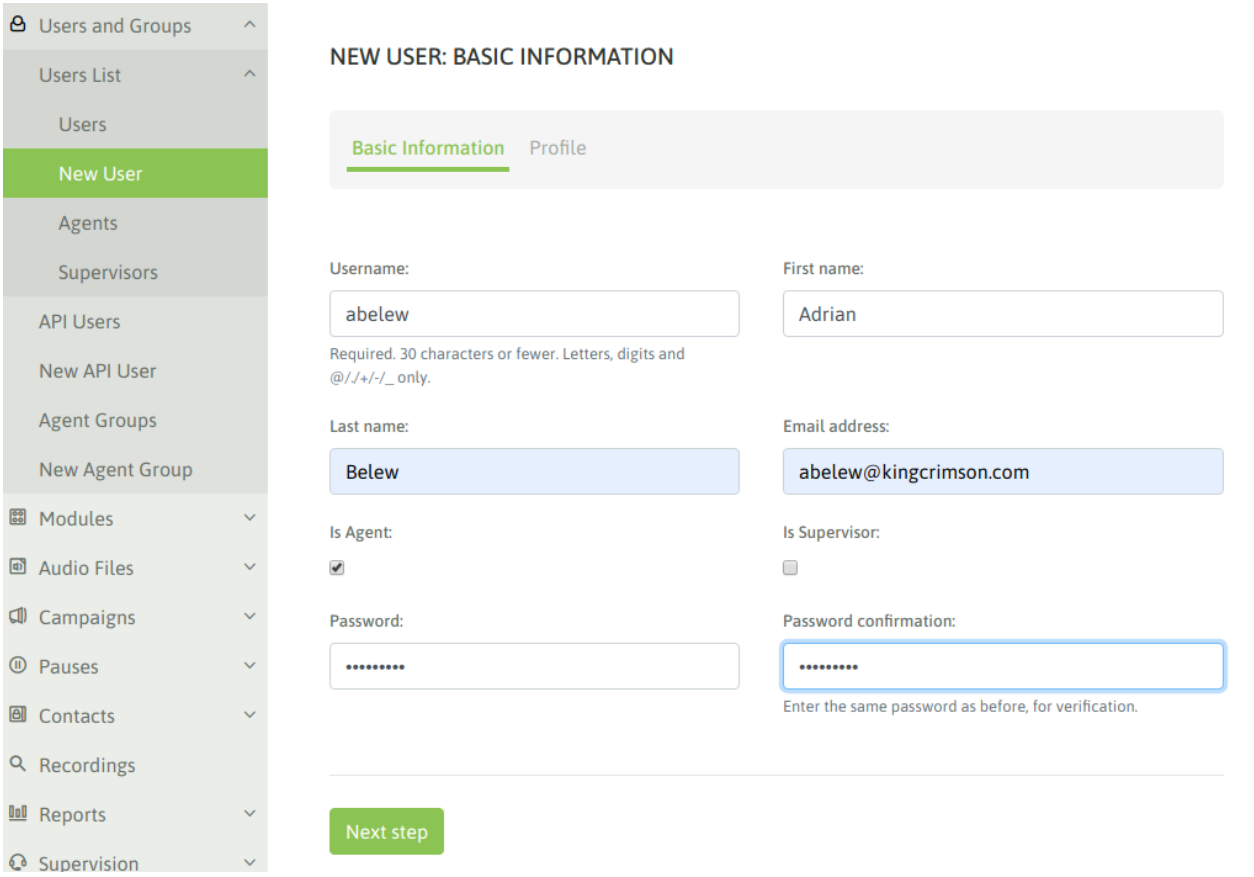


Figure 9: New agent

The agents group and available modules must be selected for our new agent.

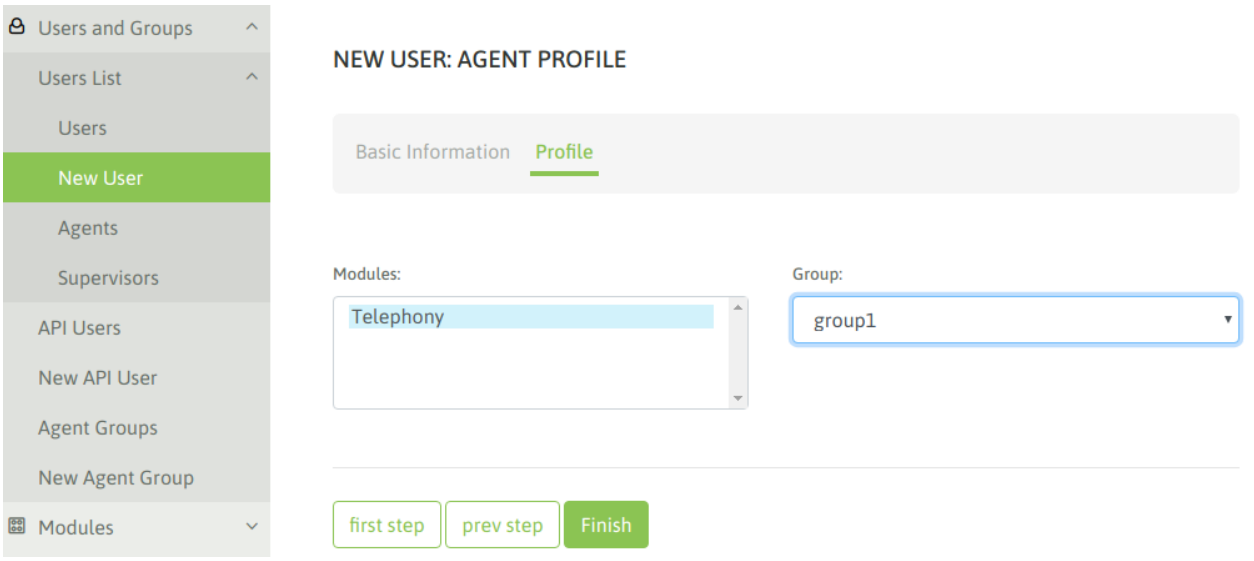


Figure 10: New agent

We can list our users, we should have at least a couple of agents.

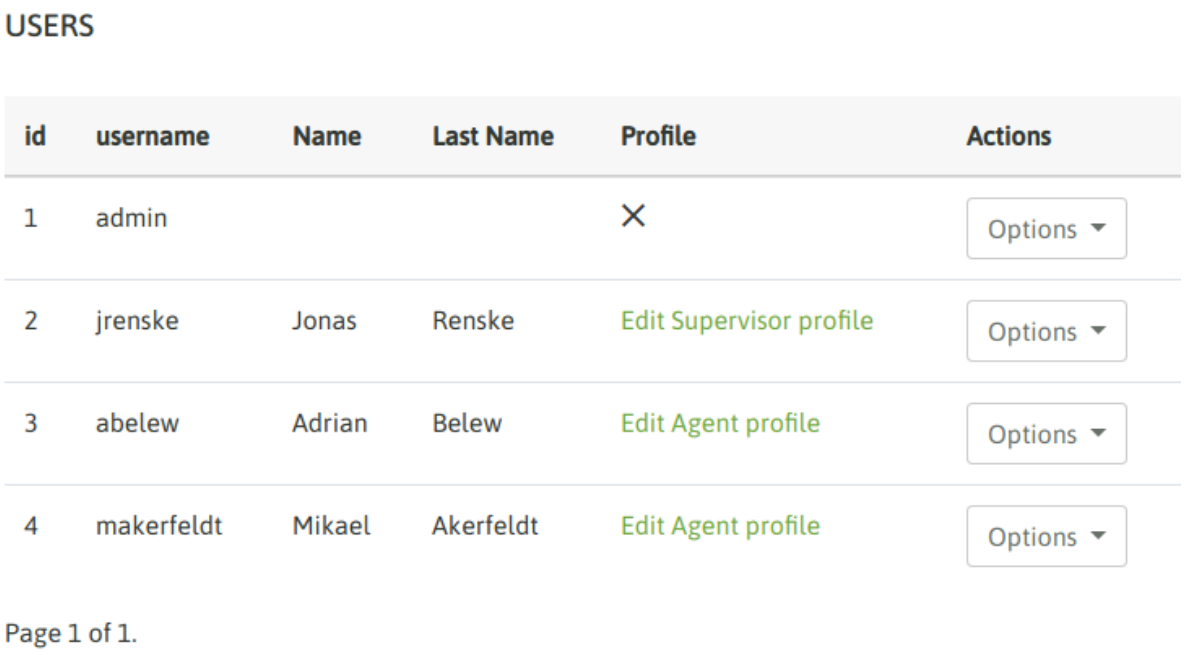


Figure 11: Users

8.1.7 Create agent pauses

The agents can enter a pause whenever they want to be unavailable for the call processing, this prevents that an incoming or dialed campaign assigns a new call. Also pause states are useful for recording productivity and measuring agent session times.

Pauses can be generated by supervisors and are classified as Recreational and Productive pauses.

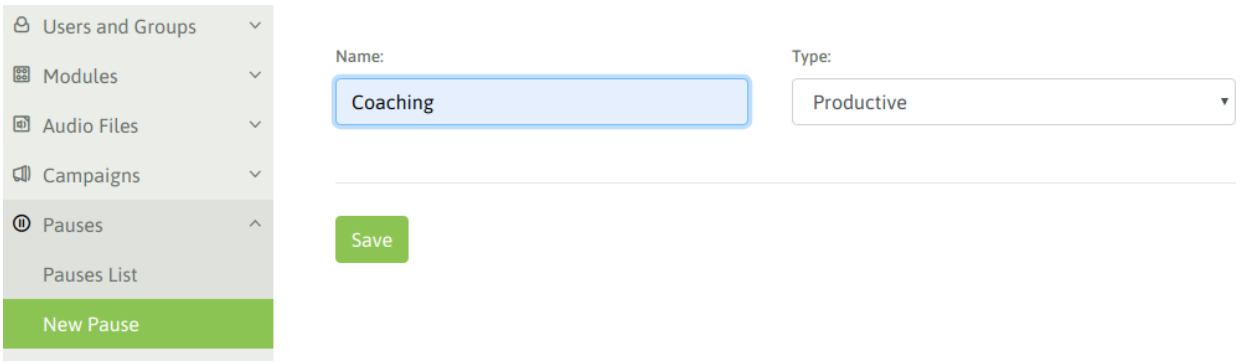


Figure 12: New pause

At the time of presenting the agent sesión reports, the totalized pauses are divided in recreative pauses and productive pauses. This allows measure the productivity of our agents in a more exact way.

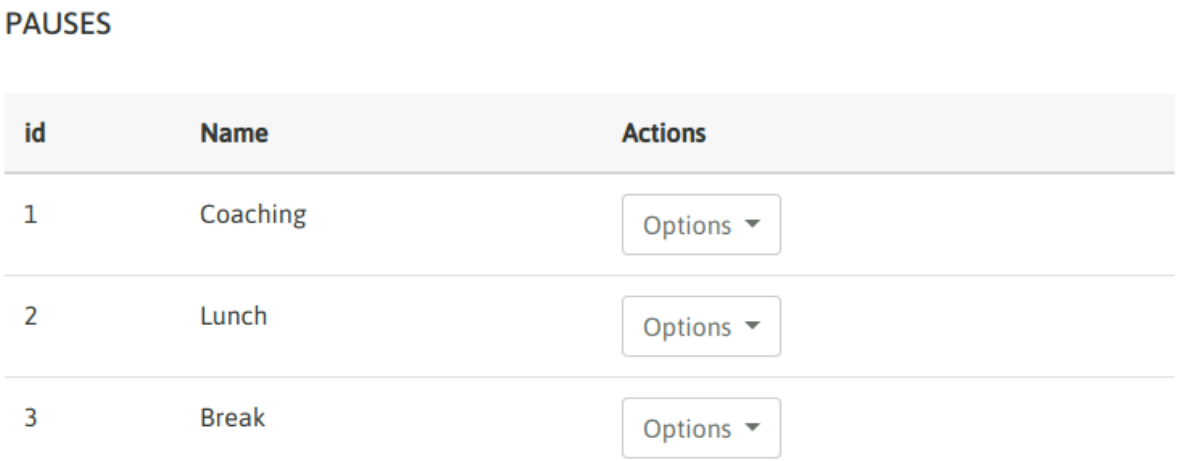


Figure 13: New pauses

Once our agents are generated, we can proceed with our first agent login !

8.1.8 First agent login

Finally we have everything ready to try our first agent.

Important: Tener en cuenta que para obtener un login exitoso debemos debemos contar con un **MICROFONO disponible:** en la estación de trabajo desde la cual se ha realizado el login de agente. Si no se cumple entonces el login será defectuoso.

Once we access with our agent, if everything goes well we should run into a popup which request the permission to take control of the microphone, ass illustrated in Figure 14.

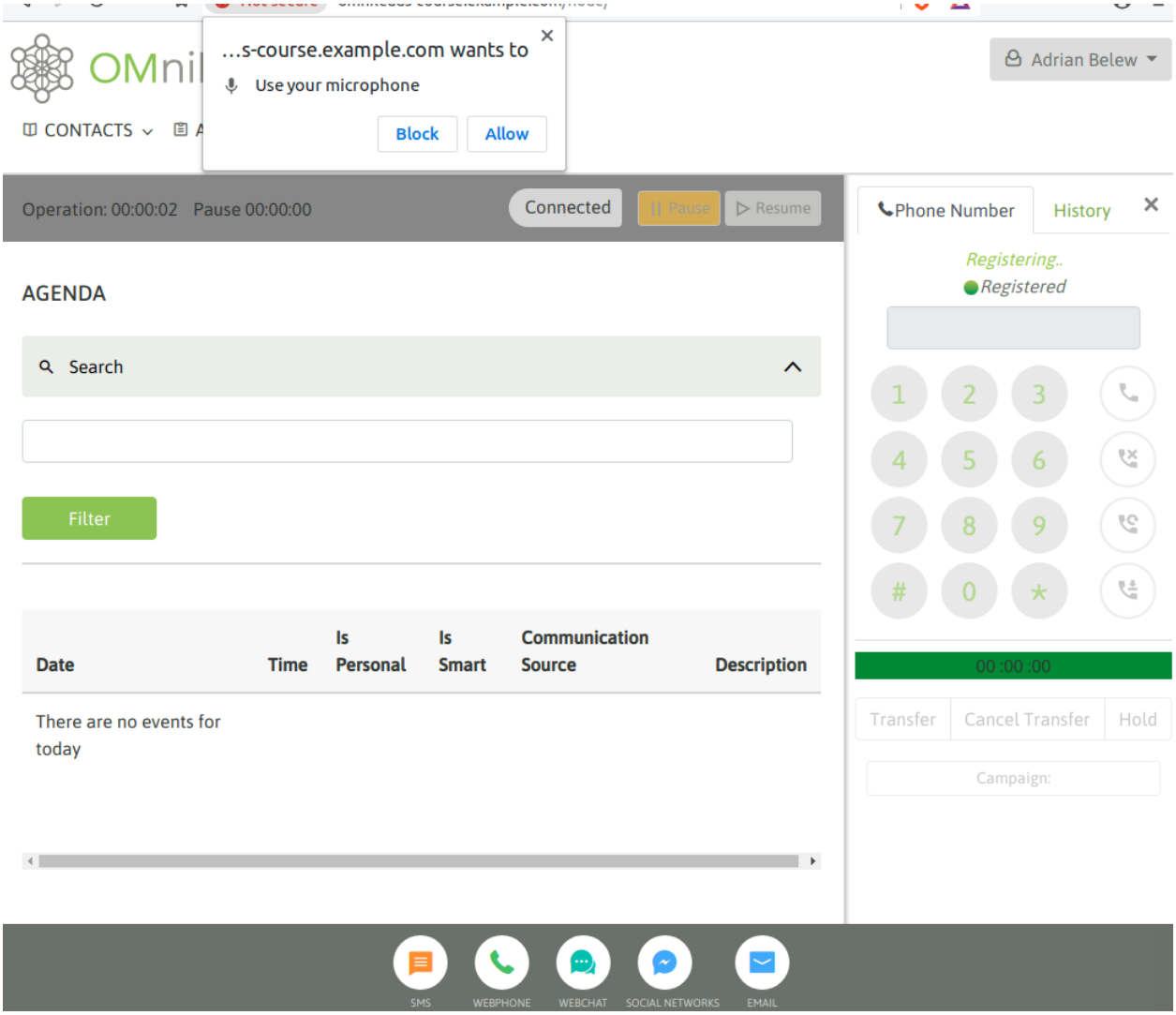


Figure 14: First login - microphone webrtc

When allowing the permission, we should listen to an audio that the system reproduces indicating the successful login and also the agent screen should look like figure 15.

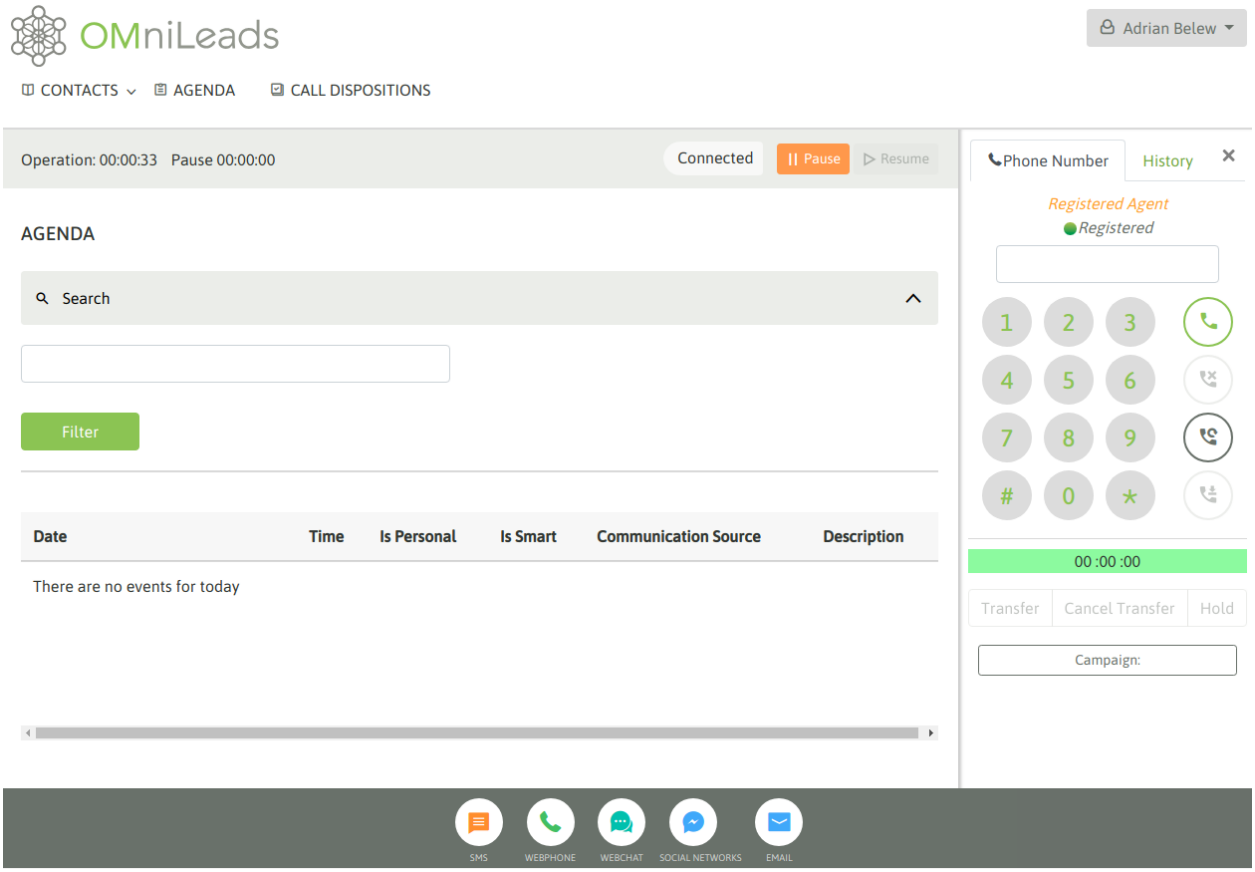


Figure 15: First login - webphone connected

CHAPTER 9

PSTN ACCESS CONFIGURATION

OMniLeads through web configuration, ease the posibility of maintain SIP trunks to access the PSTN. This trunks are invoked by the routing rules of outbound calls, where it can be specified which tye of calls are processed by each SIP trunk. Also, the trunks can be configured in *failover* mode

To go deeper it's recommended to read the rest of the chapter.

9.1 Access configuration to PSTN

En este capítulo vamos a explicar cómo dejar configurada la capa de *acceso a la PSTN* de OMniLeads para operar de acuerdo a los contextos planteados. OMniLeads admite solamente SIP como tecnología de interconexión con otros conmutadores de telefonía. Por lo tanto el integrador podrá configurar troncales SIP de proveedores ITSP, troncales SIP contra sistemas PBX y/o troncales SIP contra Gateways FXO o E1/T1.

Important: A partir del **Release-1.4.0** OMniLeads implementa y recomienda el uso de troncales SIP basados en el módulo PJSIP de Asterisk, debido a su eficiencia en términos de recursos informáticos utilizados, así como también por el estado de *deprecated* en el cual se encuentra el módulo CHAN_SIP a partir de la versión 17 de Asterisk.

Por lo tanto a la hora de plantear el uso de troncales SIP en esta documentación se utilizará PJSIP. Para disponer de la explicación de troncales utilizando chan_sip, usted podrá visitar las versiones anteriores de esta documentación.

9.1.1 Explicación del ABM para troncales PJSIP

Para acceder a la configuración debemos ingresar en el punto de menú *(Telephony -> SIP Trunks)* y allí añadir una nueva troncal PJSIP. Se va a desplegar un formulario similar al de la figura 1.

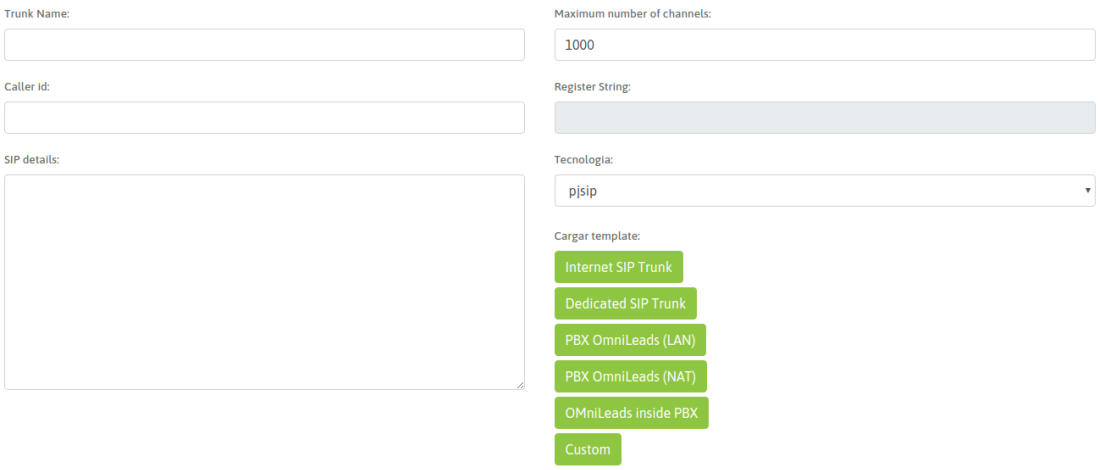


Figure 1: New SIP Trunk

The form blanks are:

- **Trunk Name**: el nombre del troncal. Debe ser alfanumperico sin espacios ni caracteres especiales (Ej: Trunk_provider_a).
- **Number of channels**: is the number of channels allowed by the link.
- **Caller id**: the number with which the calls will go through the trunk.
- **SIP details**: en este campo de texto se proporcionan los parámetros SIP usando sintaxis de PJSIP configuration Wizard de Asterisk.

Como se puede apreciar en la figura 1, existen una serie de *plantillas* de configuración en base al tipo de conexión troncal que se desea configurar, que pueden facilitar bastante el trabajo de configuración .

A continuación desplegamos las plantillas sugeridas para los tipos de escenarios planteados como casos de uso típicos de OMniLeads.

- about_telephony_pjsip_internet_provider.
- about_telephony_pjsip_lan_provider.
- about_telephony_pjsip_lan_pbx.
- about_telephony_pjsip_internet_pbx.
- about_telephony_pjsip_oml_inside_pbx.

9.1.2 Configuración para el enrutamiento de llamadas salientes

OMniLeads permite gestionar el enrutamiento de llamadas salientes sobre múltiples troncales SIP (previamente creados), de manera tal que utilizando criterios como el *largo o prefijo del número* para determinar por qué vínculo SIP encaminar la llamada. Además es posible mantener una lógica de *failover* entre los diferentes troncales SIP asignados a una ruta saliente.

Para acceder a la vista de configuración de rutas salientes, ingresar al punto de menú (*Telephony* -> *Outbound routes*)

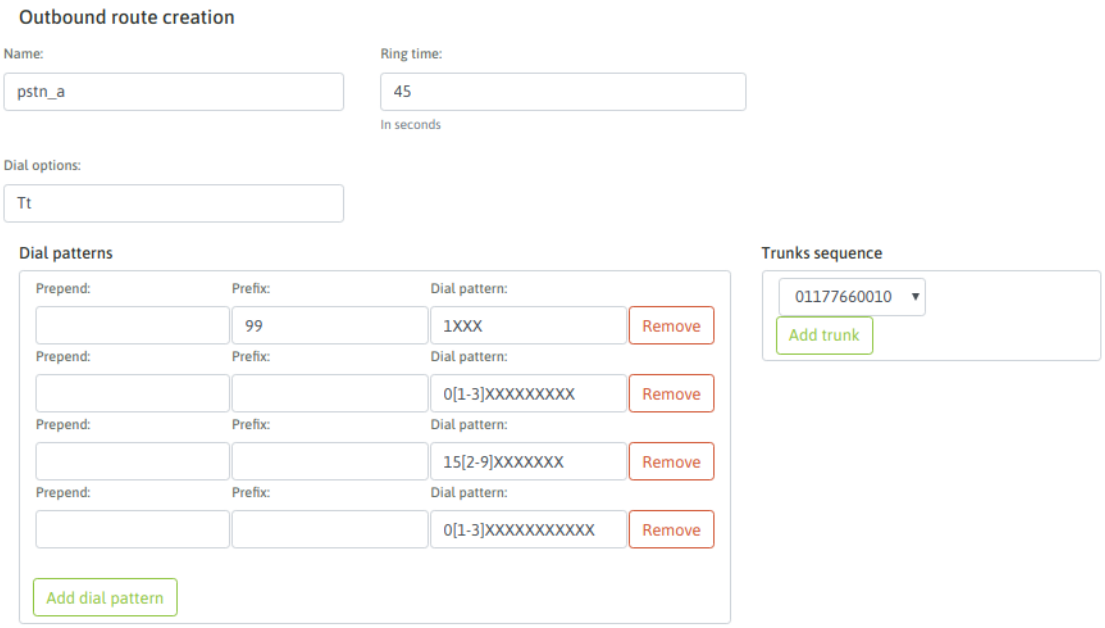


Figure 2: Outbound route

- **Nombre**: es el nombre de la ruta (alfanumérico sin espacios)
- **Ring time**: es el tiempo (en segundos) que las llamadas cursadas por esta ruta, intentarán establecer una conexión con el destino, antes de seguir intentando por el próximo troncal o bien descartar la llamada.
- **Dial options**: son las opciones de la aplicación "Dial" utilizadas por Asterisk(r) en bajo nivel.
- **Patrones de discado**: mediante patrones, se puede representar los *tipos de llamadas* que serán aceptadas por la ruta y así entonces colocadas sobre el Troncal SIP para finalmente alcanzar el destino deseado.

Para comprender cómo se representan los dígitos utilizando *patrones*, se recomienda leer éste link: https://www.voip-info.org/asterisk-extension-matching/.

Within each pattern entered there are three fields:

- **Prepend**: son los dígitos que se mandan por el trunk SIP como adicionales al número discado. Es decir llegan al Trunk posicionados delante del número marcado.
- **Prefijo**: son los dígitos que pueden venir como "prefijo" de una llamada marcada y éstos serán quitados en el momento de enviarlos por el SIP Trunk.
- **Patrón de discado**: se busca representar en este campo el patrón de dígitos autorizados que la ruta va a procesar para y enviar a un SIP Trunk para sacar la llamada hacia el exterior.
- **Secuencia de troncales**: son los troncales SIP sobre los cuales la ruta saliente va a intentar establecer la llamada discada por OML. Si la llamada falla en un troncal, se sigue intentando en el siguiente.

9.1.3 Configuración de enrutamiento de llamadas entrantes

El tratamiento de llamadas entrantes se abordará en la sección *Campañas Entrantes* de esta documentación ya que para poder operar con dicho módulo debemos al menos tener creado algún objeto (IVR, condicional de tiempo, campaña entrante, etc.) hacia a donde rutear cada DID generado.

about_inboundroutes.

CHAPTER 10

CAMPAIGNS

The processing of communications between the "outher world" and an OMniLeads's agent is encapsulated in a campaign. In this chapter all about the management of Inbound and Outbound (manual, preview and dialer) campaigns is covered.

10.1 Telephone Campaigns

A campaign represents a Contact Center operation that groups:

- An agent group processing phone calls in one direction (outbound or inbound)
- A contact base associated with the campaign.
- A dispositions list that are displayed when classifying the callprocessed by the agent.
- One or more forms to be launched in case the agent assigns a *engage disposition* associated with any of the forms on the call in progress the form is displayed by said engage disposition and the agent can complete it according to the contact details of the ongoing call.

Figure 1 shows a campaign and its components.

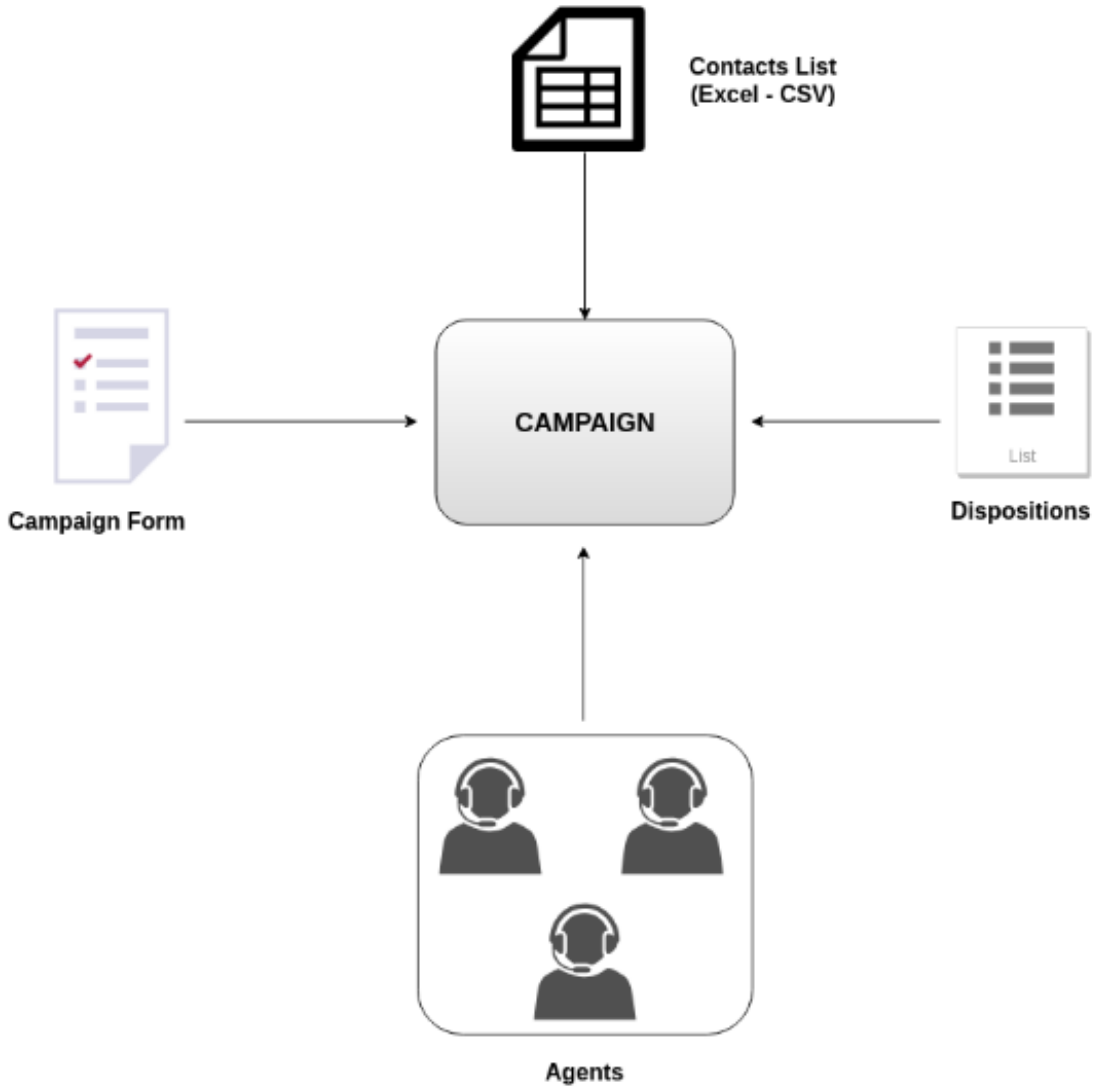


Figure 1: the campaigns elements

The fact of working with campaigns allows *Supervisor or Administrator* user profiles to focus metrics and reports, as well as real-time monitoring or search for recordings, among other actions, using campaigns as a filtering criteria. We can have different campaigns of different nature (predictive, preview, incoming or manual), living simultaneously in OMniLeads and leaving records about the calls transacted by the agents.

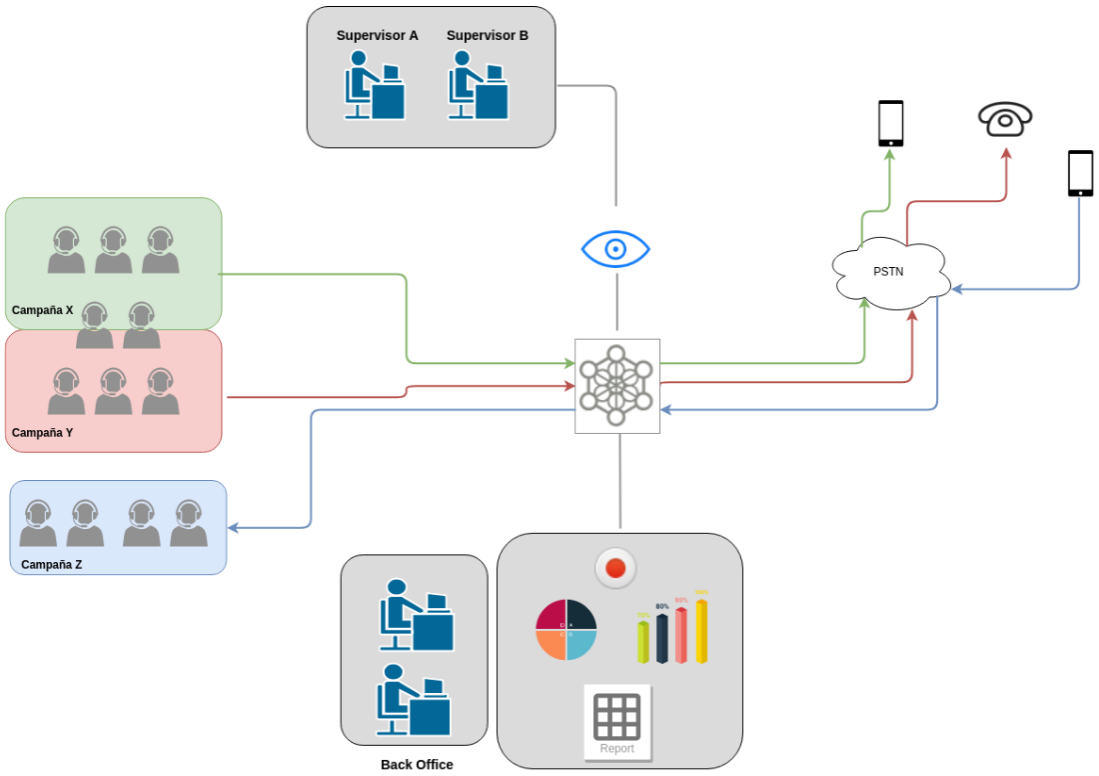


Figure 2: Campaigns, agents, supervisors and backoffice

10.1.1 Dispositions

Dispositions form a list of *tags* available to be assigned to any campaign, so that agents can classify each call within a campaign, with any of these dispositions.

Dispositions are defined by the supervisor or administrator and can be related to several aspects, for example:

- The telephone call result (busy, no answer, wrong number, voicemail, etc.)
- The result of an interaction in a call answered (sale, survey completed, schedule call, etc.)

Dispositions can be totally arbitrary and to generate them you must enter the screen*Campaigns → Call Dispositions → New Call Dispositions*.

We can list the ratings generated within *Campaigns → Call Dispositions → Call Dispositions*

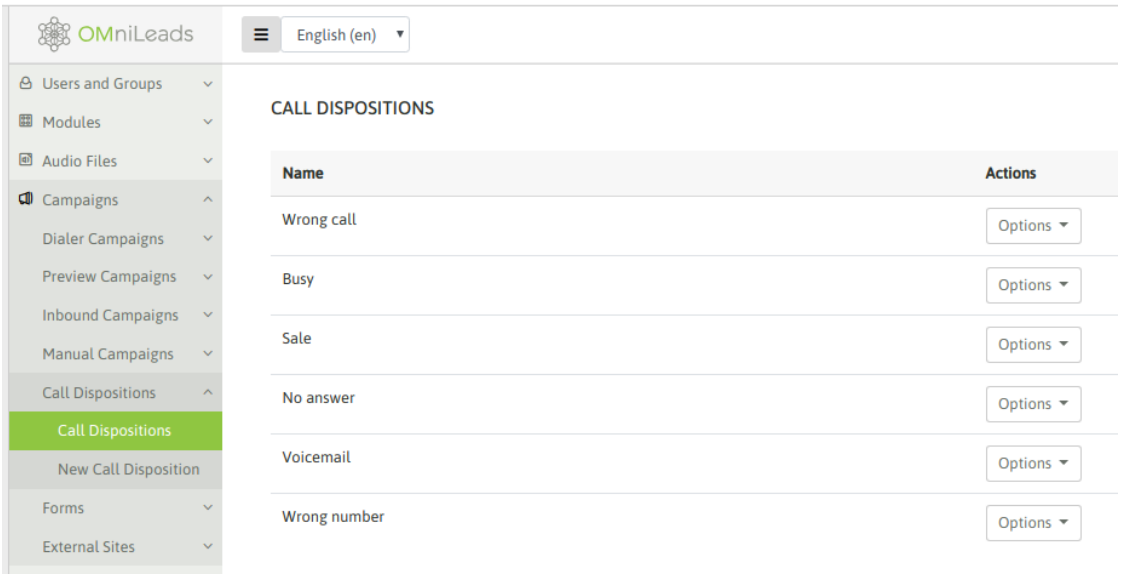


Figure 3: Call dispositions

10.1.2 Campaign contact database

Contact databases are used in outbound campaigns but can also be used by incoming campaigns. In the context of outbound campaigns, the data required by the predictive or preview dialer is provided by the contact base affected to the campaign, while in incoming campaigns they provide the complementary data to the phone number that is displayed on the agent's screen every time a call comes in.

Contact databases are provided in CSV format files with fields separated by a comma and also generated in the UTF-8 encoding (excluding requirement). There must be at least one column that contains a telephone for each contact (record) of the file, the rest of the columns can contain any content, generally each record has complementary data to the main telephone. This data is displayed on the agent screen when establishing a communication between both (agent and base contact).

```
tel1,tel2,tel3,name,surname,id
4149090,5149095,6149097,Terence,McKenna,1000001
4149014,5149015,6149017,Timothy,Leary,1000002
4149020,5149025,6149027,Albert,Hoffman,1000003
4149032,5149095,6149037,Aldous,Huxley,1000004
4149040,5149092,6149047,Herman,Hesse,1000005
4149052,5149092,6149055,Jorge Luis,Borges,1000006
4149050,5149092,6149067,Jiddu,Krishnamurti,1000007
4149050,5149090,6149077,Aleister,Crowley,1000008
4149050,5149090,6149087,Facundo,Cabral,1000009
4149052,5149090,6149090,Atahualpa,Yupanqui,1000010
4149055,5149095,6149000,Carl Gustav,Jung,1000011
4149050,5149090,6149010,Isaac,Luria,1000012
4149057,5149095,6149026,Robert Anton,Wilson,1000013
```

Figure 4: Contacts CSV file - text editor view

	A	B	C	D	E	F
1	tel1	tel2	tel3	name	surname	id
2	4149090	5149095	6149097	Terence	McKenna	1000001
3	4149014	5149015	6149017	Timothy	Leary	1000002
4	4149020	5149025	6149027	Albert	Hoffman	1000003
5	4149032	5149095	6149037	Aldous	Huxley	1000004
6	4149040	5149092	6149047	Herman	Hesse	1000005
7	4149052	5149092	6149055	Jorge Luis	Borges	1000006
8	4149050	5149092	6149067	Jiddu	Krishnamurti	1000007
9	4149050	5149090	6149077	Aleister	Crowley	1000008
10	4149050	5149090	6149087	Facundo	Cabral	1000009
11	4149052	5149090	6149090	Atahualpa	Yupanqui	1000010
12	4149055	5149095	6149000	Carl Gustav	Jung	1000011
13	4149050	5149090	6149010	Isaac	Luria	1000012
14	4149057	5149095	6149026	Robert Anton	Wilson	1000013

Figure 5: Contacts CSV file - libreoffice excel view

A contact base (csv) is available, to proceed with the the loading of the file into the system by accessing the menu item; *Contacts → New contacts database*

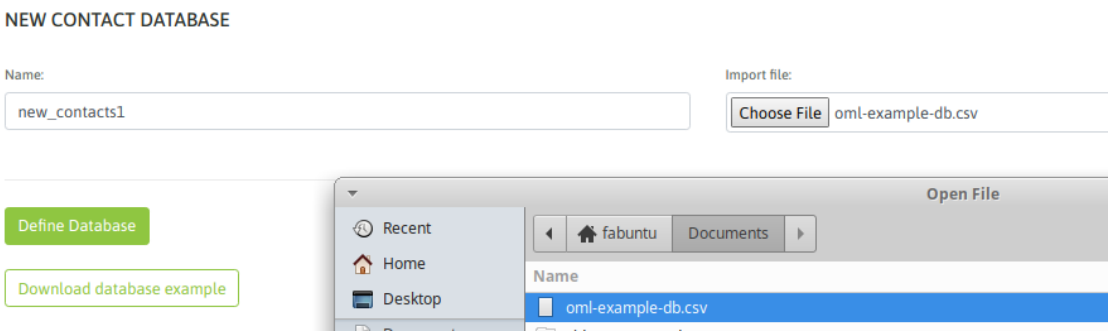


Figure 6: New contact database

It should be indicated with a check, which columns are the ones that store phones, as indicated in figure 7.

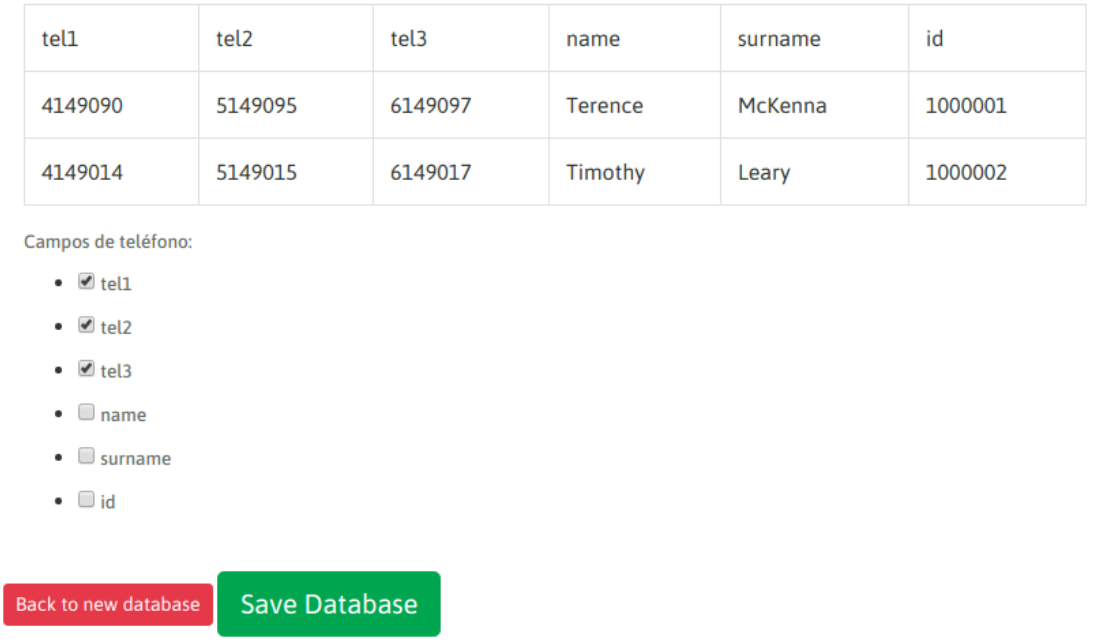


Figure 7: Tel check

Finally, the form is saved and the CSV file is now available as a system contact database available for any campaign.

10.1.3 Forms

Campaign forms constitute the default method to collect information (relevant to the campaign) in the interaction with the person behind an established communication. They are designed by the user combining in a static view different types of fields (text, date, multiple selection and text area), being able to order them according to the need of the campaign.

To create forms you must access the menu item; *Campaigns → New form*

Campaign forms may contain field types like

- **Text**
- **Date**
- **List**
- **Text area box**

Figure 8 exemplifies a campaign form creation.

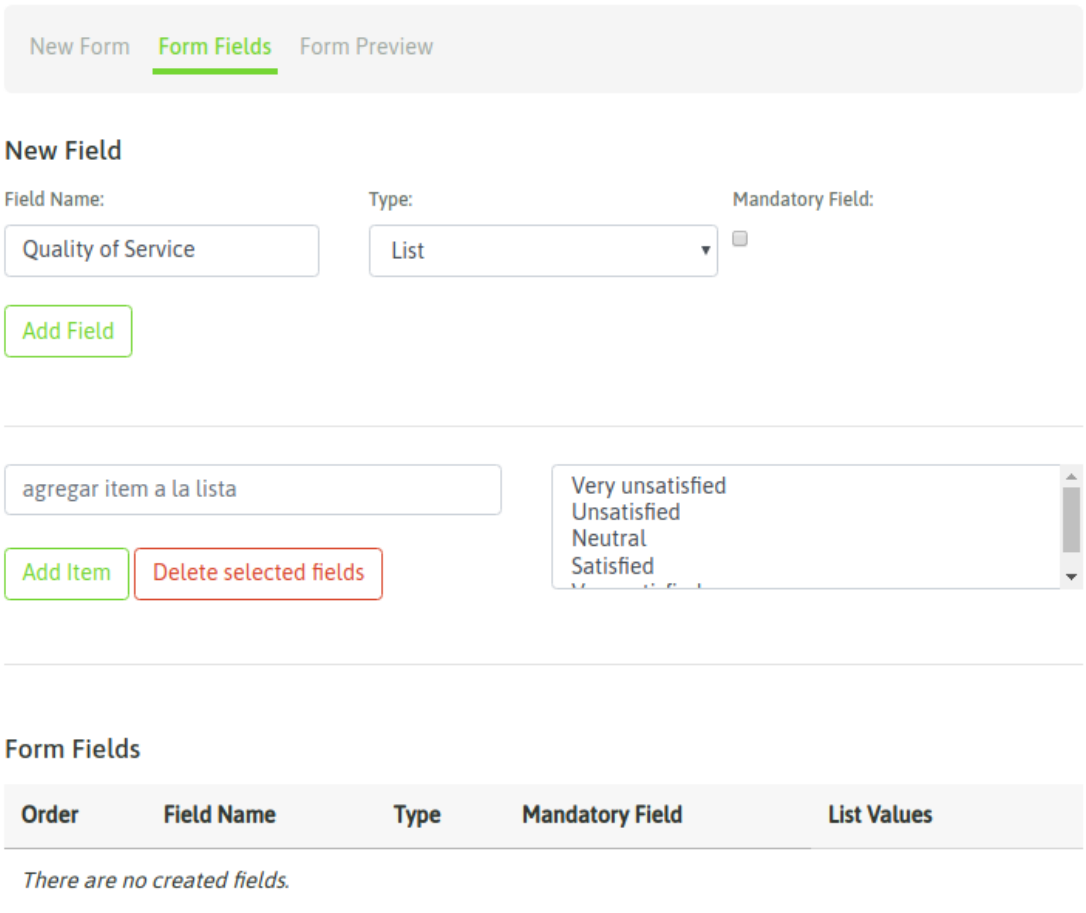


Figure 8: New campaign form

We can generate a sample satisfaction survey form with the appearance of figure 9.

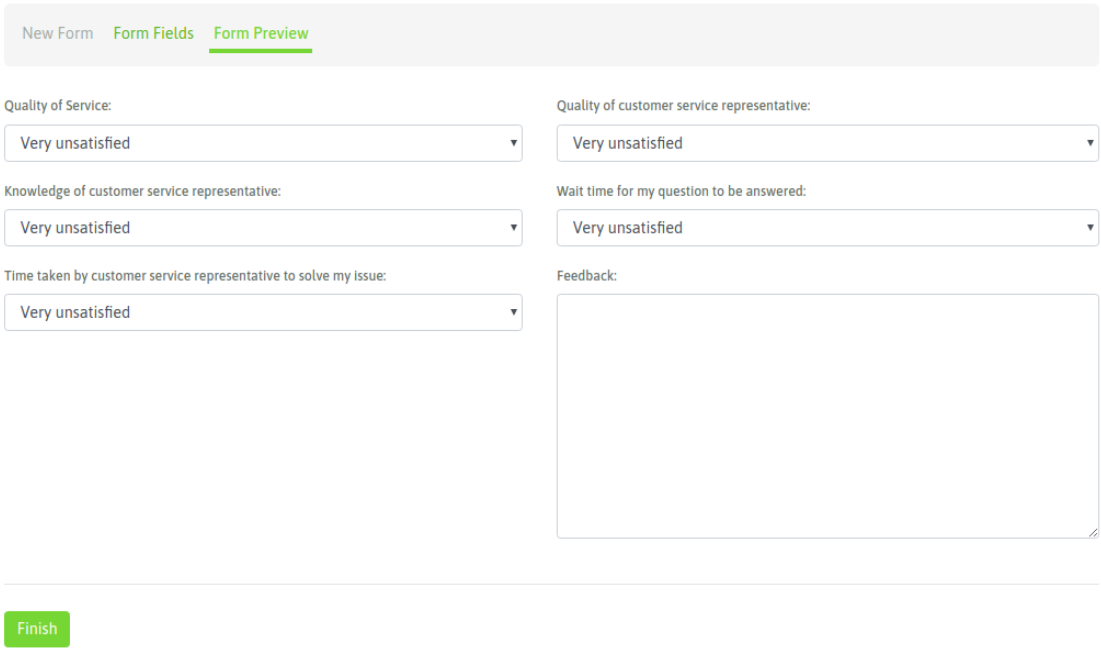


Figure 9: Survey campaign form

10.1.4 Campaigns, Dispositions & Forms

To explain the relationship between these components, we must remember that multiple forms can be assigned to a campaign. The idea is that different dispositions of a campaign can trigger different forms, thus allowing the ability to collect through previously designed forms, information associated with the interaction between the OMniLeads agent and the person at the other end of the communication within the campaign.

It is important to explain conceptually how campaign forms are used in OMniLeads. We must clarify that in the context of a campaign when assigning ratings, it will be possible to define normal and *engaged disposition*. The latter are the ones that trigger the campaign forms.

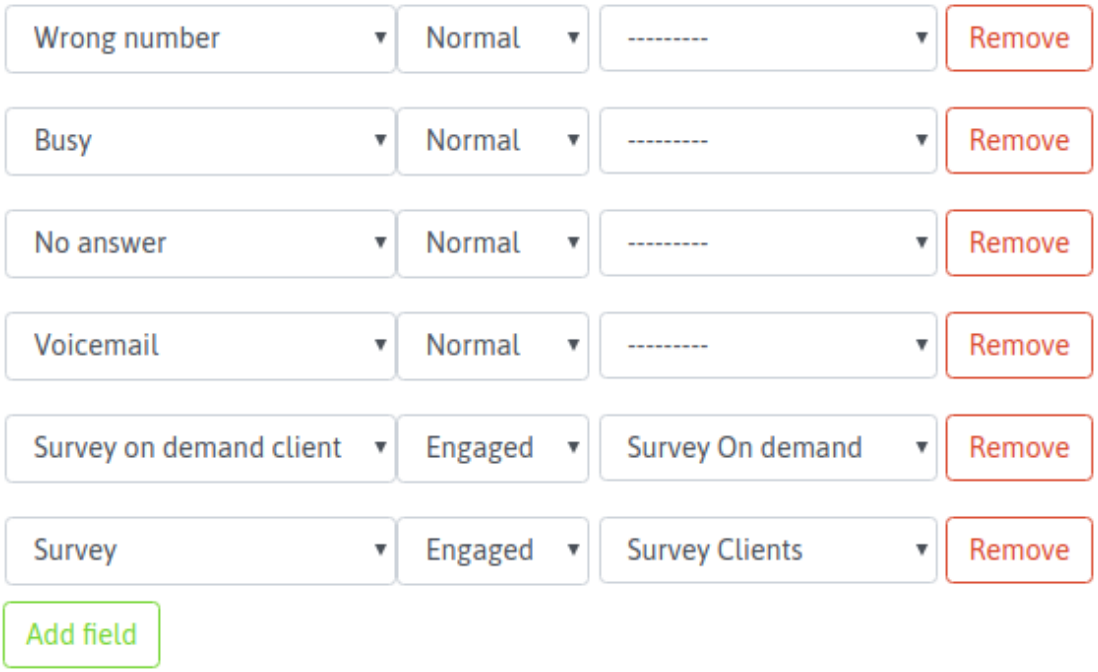


Figure 10: Call dispositions inside campaign

In the figure 10 example, we have two engaged dispositions, on the one hand the *Survey on demand client* disposition that triggers the *Survey on demand* form, and on the other the *Survey* disposition that triggers the *Survey clients* form.

Whenever there is an active call between an OMniLeads agent and a contact at thecampaign contact database, the agent has the complementary data to the contact's phone on their screen next to the disposition selection combo for the current contact. If the agent selects and saves a *engaged disposition* then the form associated with the engaged disposition within the campaign is triggered on the agent screen.

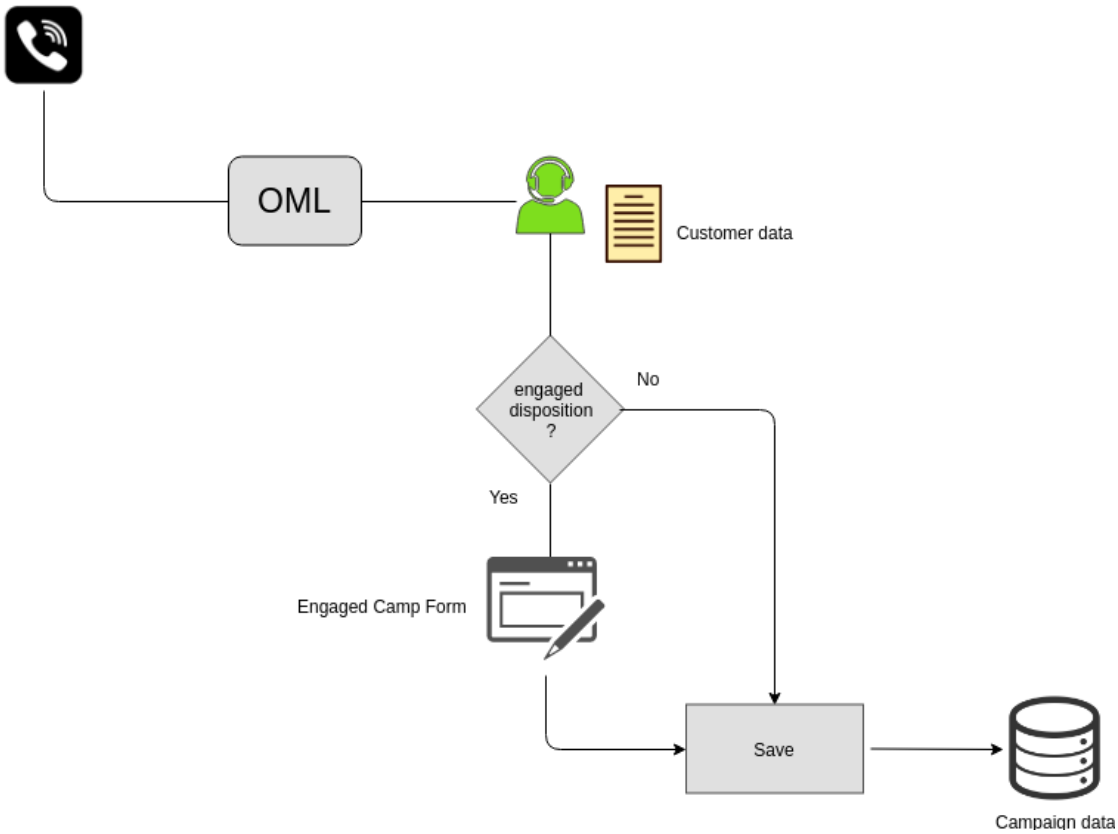


Figure 11: Engaged dispostions and forms

10.1.5 Manual Campaigns

Within this subsection the step-by-step example of how to manage Manual Campaigns is exemplified. about_manualcamp.

10.1.6 Preview Campaigns

Within this subsection the step-by-step example of how to manage Preview Campaigns is exemplified. about_previewcamp.

10.1.7 Predictive dialer campaigns

Within this subsection the step-by-step example of how to manage Predictive dialer Campaigns is exemplified. about_dialercamp.

10.1.8 Inbound Campaigns

When talking about incoming calls we have to explain each functionality applicable to the flow of a incoming call. As we know an incoming call can go through a series of "nodes" until finally connecting with an agent. Therefore we will extend the concept of "incoming campaigns" to the following configuration items.

- about_inboundcamp.
- about_inboundroutes.
- about_inboundroutespbx.
- about_timeconditions.
- about_ivr.
- about_customer_id.
- about_custom_dst.

10.1.9 Plantillas de Campaña

It is often recurring that the parameters of a "class" of campaigns (for example, survey preview campaigns) do not vary too much except perhaps by the assigned agent group, a contact database, or by assign supervisor. Therefore instead of having to create very similar campaigns always from scratch, you can generate templates and then quickly create new campaigns from cloning those templates.

This functionality is granted by the OMniLeads campaigns *Templates*.

Once you have generated a template (which is generated similarly to a campaign), you can create new campaigns simply by selecting the template and the *Create campaign from template* option. Each new campaign will be available with all the parameters specified in its parent template.

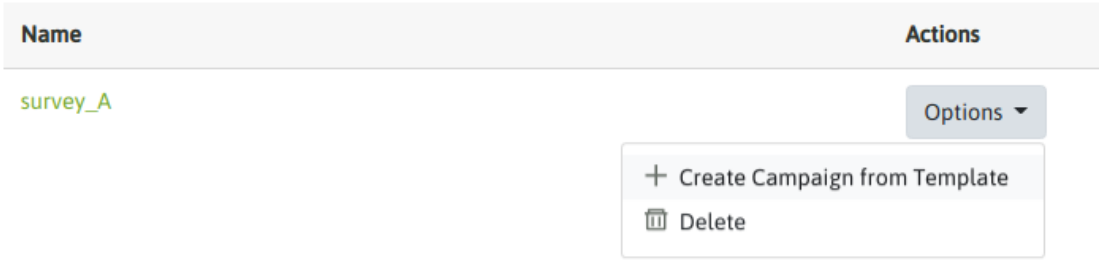


Figure 13: templates

10.1.10 Interaction with external CRMs

OMniLeads is designed from a perspective that prioritizes integration with the company's preferred CRM/ERP system. Thus providing the possibility for a company to maintain the use of its CRM/ERP system appropriate to its vertical market (health, sales, customer service, etc.).

Through its own functionalities and API methods, OMniLeads allows the following interactions:

- Open a specific view of the CRM in an incoming or outgoing communication, using communication parameters (agent id, contact id, campaign id, etc.) as dynamic data to invoke the CRM.
- Execute *click to call* from a contact view in the CRM and thus activate a call through a campaign and OMniLeads agent.
- Record the management of a CRM contact and that the disposition impacts OMniLeads, so that there is a correlation between the CRM system and the Contact Center system within each campaign.
- Request the "ID" of the person on incoming calls, send this "ID" to the CRM (via http request) and wait for the latter to respond on which OMniLeads campaign to route a call.

These concepts and configurations are expanded in the following link *CRM integration*.

CHAPTER 11

METRICS, REPORTS, RECORDINGS AND SUPERVISION

Inside this chapter, all about the metrics, statistics, reports, recordings and real time supervision, etc.

11.1 Reporting, Recordings and Supervision

11.1.1 Recordings

All campaigns that operate with the call recording option enabled generate recording files with listening and download actions. The OMniLeads *Recordings* module allows to search for recordings using filters as search criteria.

- about_recordings.

11.1.2 Inbound Campaigns output

This section describes and analyzes all staff related to the "output" generated by incoming call campaigns.

- about_inbound_camp_reports.

11.1.3 Outbound Campaigns output

This section describes and analyzes everything related to the "output" generated by outbound call campaigns.

- about_outbound_camp_reports.

11.1.4 General Call Reports

This section reviews each report generated by general call reports. *Reports - Calls*.

- about_general_reports.

11.1.5 Agent Reports

This section reviews each report generated by the general report of agent activity. *Reports - Agents*

- about_agent_reports.

11.1.6 Realtime Supervision

This section reviews the OMniLeads Supervision module.

- about_supervision.

CHAPTER 12

AGENT MANUAL

In this chapter is covered all the actions that an OMniLeads agent can do inside an operation. Refering to receive/make of calls, dispositions, scheduling, call transfers, and much more.

12.1 Agent Manual

We will split Manual Agent in the following topics stated below.

12.1.1 Agent Session

All related to agent session is decribed in this piece.

Login

Access the system using OMniLeads URL from a browser.

If there are no commercial certificates installed, a security exception takes place when trying to access for the first time. This is because the presence of a self-signed certificate under https protocol.

In this case, just accept the exception and continue, as shown in figure 1 and 2.

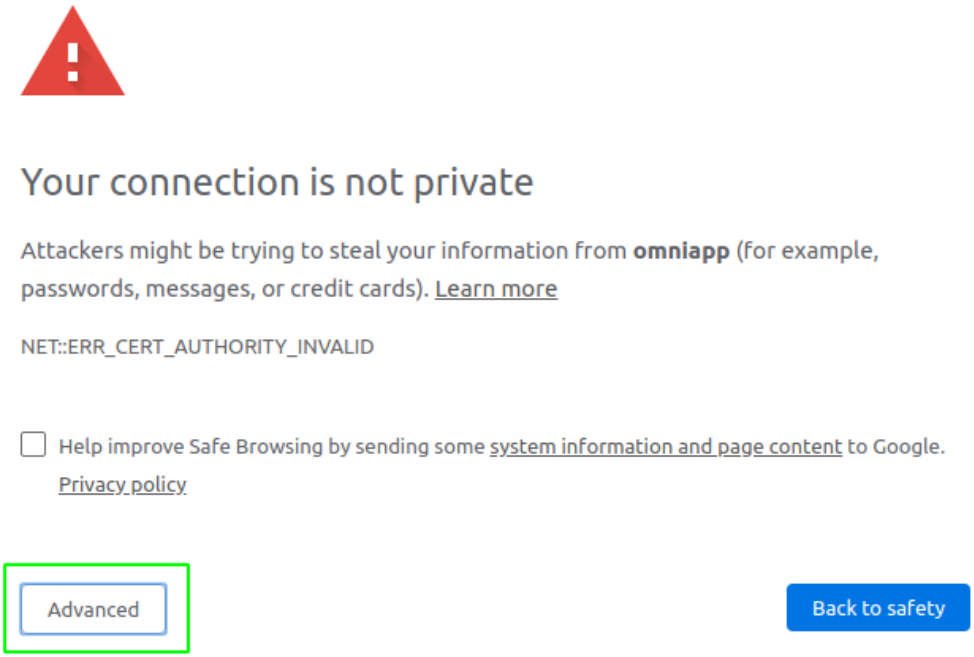


Figure 1: certificate is not trusted

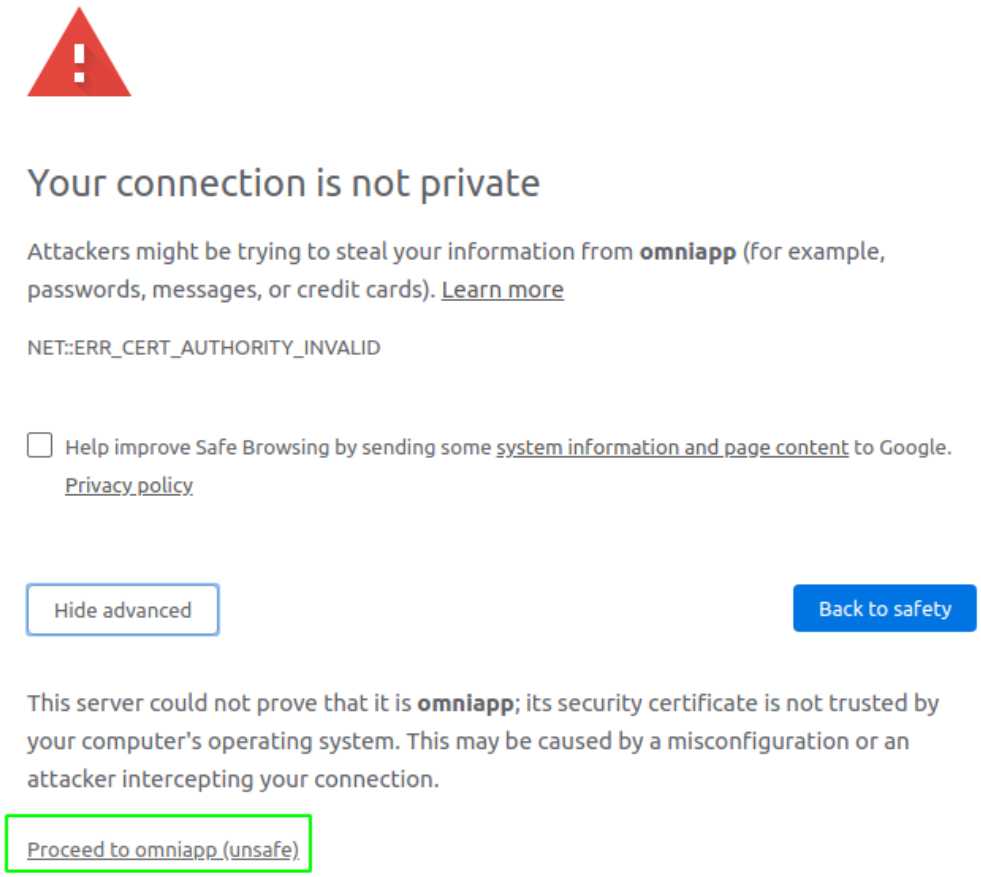


Figure 2: certificate is not trusted

After accepting security exception, agent is able to access using username and password.

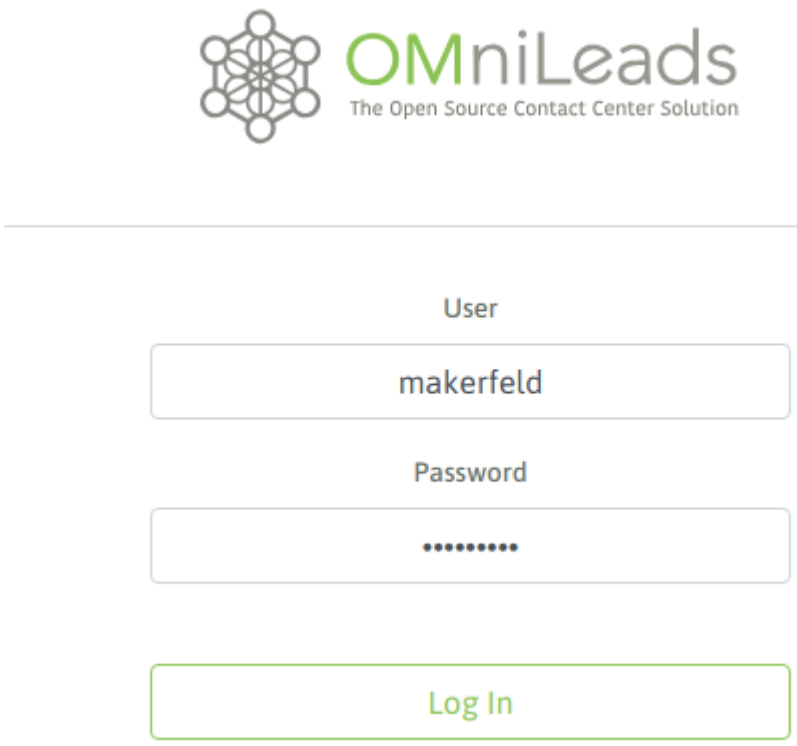


Figure 3: login screen

After successfull login, agent screen is shown. The first thing that comes up is a browser pop-up screen telling us that OML needs access to microphone. As expected, this action needs to be allowed in order for the agent to use those resources for operating.

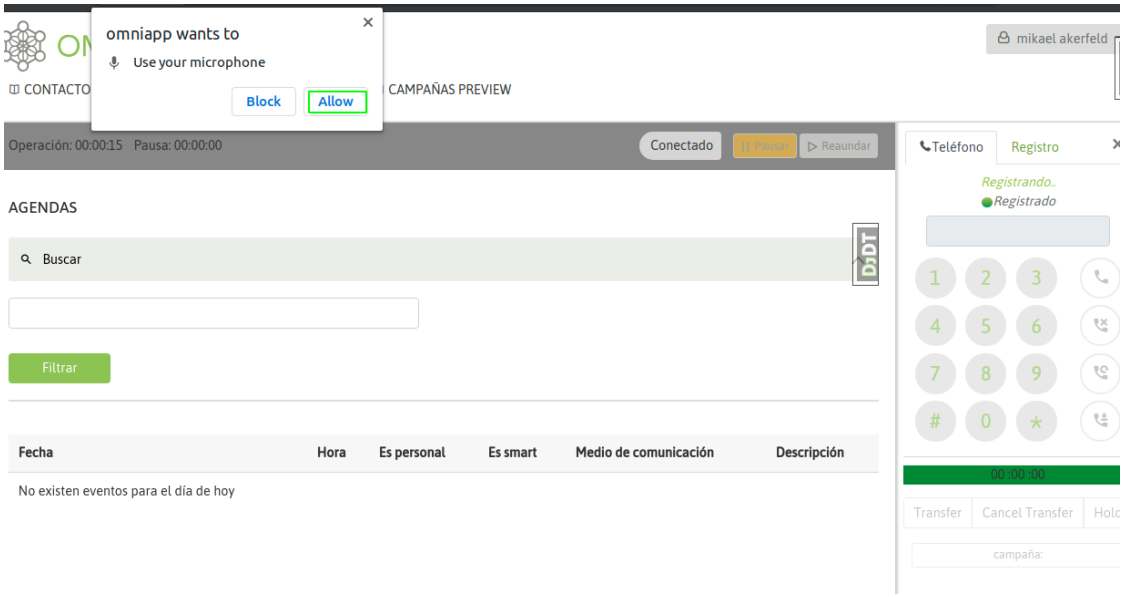


Figure 4: OMniLeads wants to use your microphone

Once this is accepted, agent is able to login successfully and can listen a brief login message over the

headsets.

Agent will see the Webphone, and if all is going on a Registered status will be displayed as it is shown in figure 5.

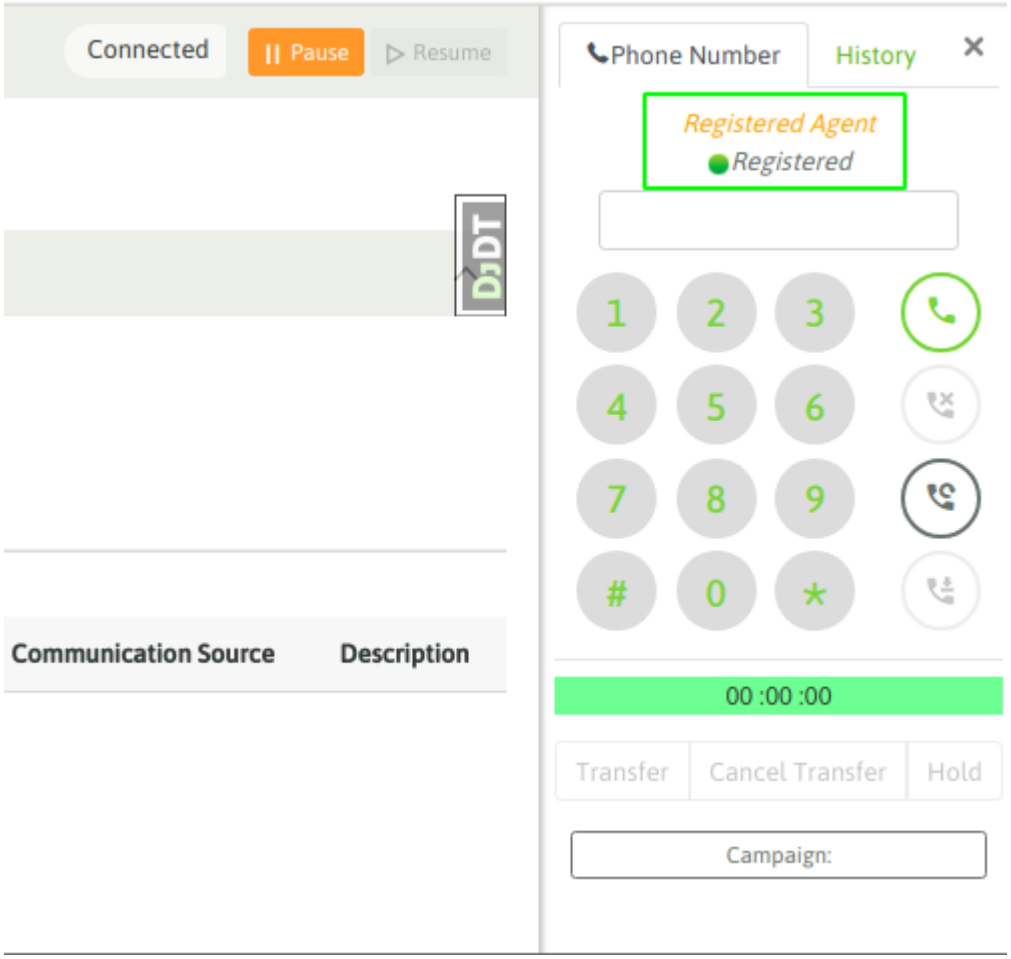


Figure 5: Succesful agent login

Agent Console

Agent console is the component where all call management flows. In the following picture, we can see the different sections that will be explained below.

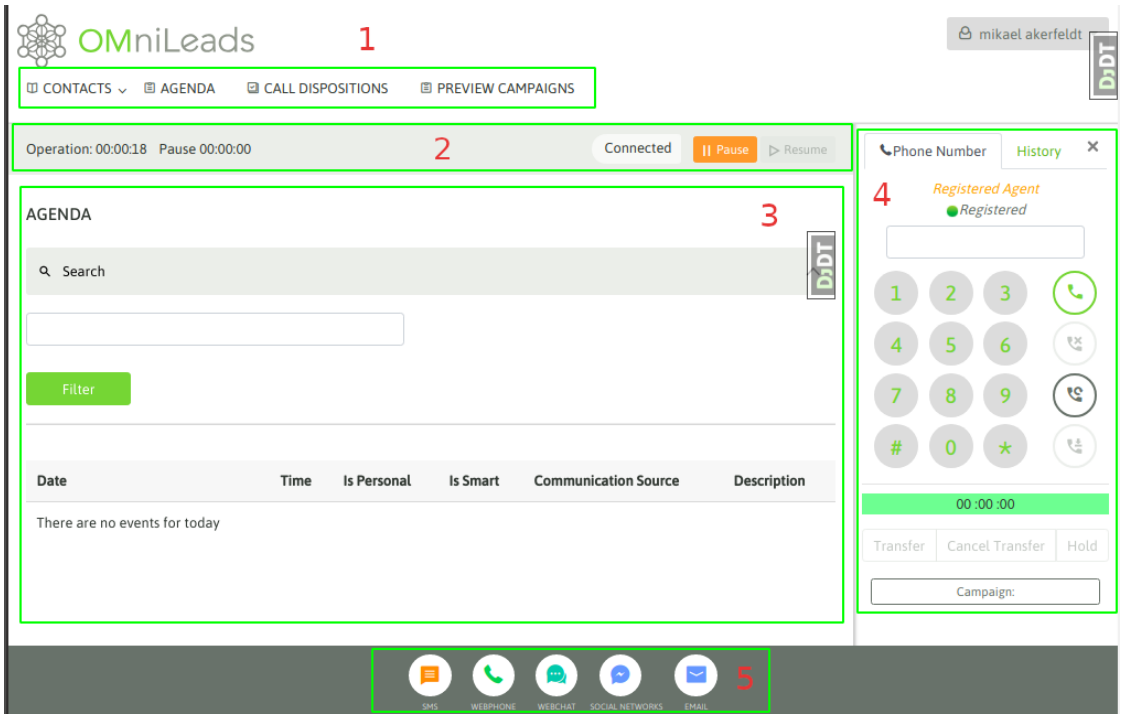


Figure 6: the OMniLeads agent console

(1) Contact Management

In this section we see the way the agent can navigate among different campaign contacts, scheduled callbacks, dispositioned calls, and also access to Preview Campaigns in order to retrieve contacts to dial.

All the above will be explained in depth in the next sections.

(2) Status Bar

In this bar, session information is shown.

Figure 7: status bar

In the status bar there are two chronometers with the amount of agent time in Pause and Ready status. Its color changes according to whether the agent is in the Ready (gray), On call (green) or Paused (yellow) status. A status legend can also be seen as a description. Finally the Pause and Un-pause buttons appear, so that the agent can go to pause and leave it.

(3) Dynamic Information Area

All the information that the agent is requesting while browsing can be seen in this section, as well as Contact Information when agent processes a new call.

(4) Webphone

OMniLeads Wephone is the main component within Telephony module. Let's review the embedded buttons.

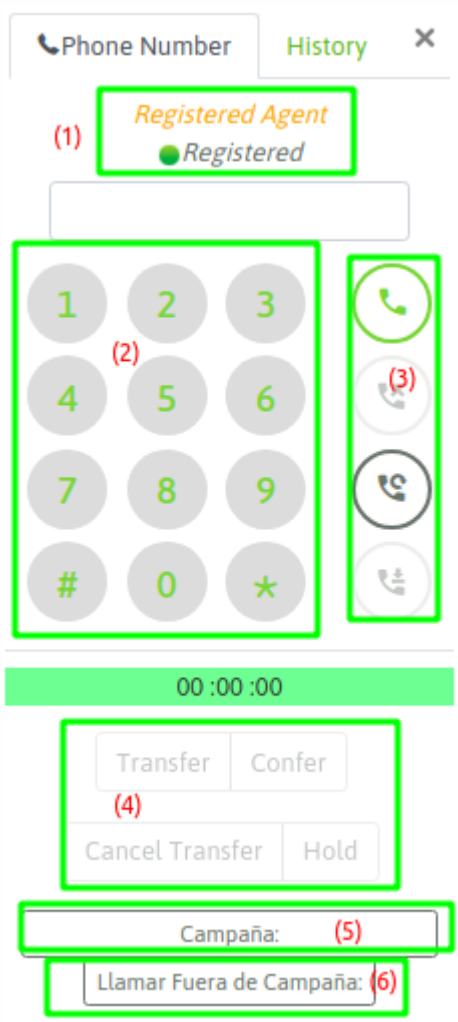


Figure 8: Webphone

Webphone can be separated in 5 pieces:

- (1) Webphone Status. It should be seen always as *Registered*. If that is not the case, please cotact the System Administrator.
- (2) Dialpad: used for dialing numbers manually or pressing DTMF options in a connected call.
- (3) These buttons allow the agent to shoot certain actions: dial, hangup, re-call, and the possibility to "leave a mark" in a conversation, i.e. a specific comment.
- (4) These buttons allow to the agent to make actions like: make a transfer call, make a conference with a third party and hold the call
- (5) This button allows to modify the campaign which is processed each manual call. More information in "manual campaign" section
- (6) This button allows to make a call to another agent, also to make an outbound call to an external number withouth association to a campaign

Almost all of the functionalities are analyzed in depth in the related sections.

Dock

In the current version, OMniLeads just supports Webphone component to make/receive calls. In future versions Product Team will be activating other modules.

Pauses

Agent has the ability to enter in Pause mode so that no incoming or predictive campaign calls are delivered. As explained in the "Initial configuration" section, there are different types of pauses the Administrator can create and maintain in the system. Therefore, when entering the pause status, the agent must select the type of pause.

For entering in Pause mode, Agent must click in the "Pause" button in the status bar of the console.

Figure 8: Pause

Pause type selection process takes place.

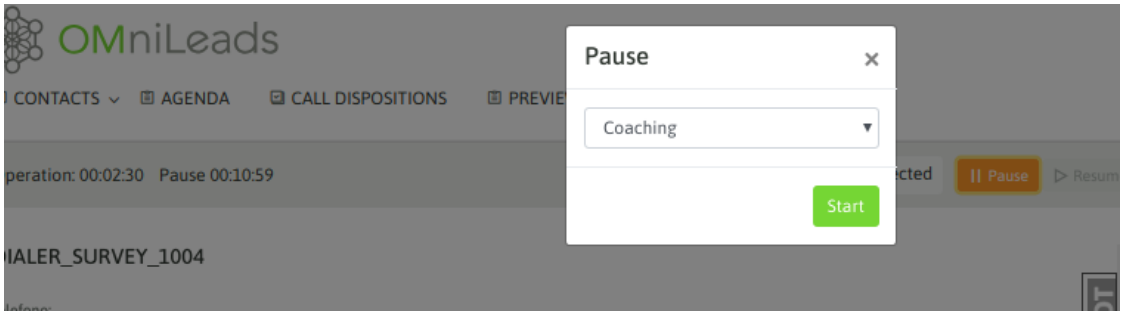


Figure 9: Pause class

Finally, the agent is paused. It should be noted how the status bar changes to "yellow" color, also the current pause type can be read in the status bar and the Pause timer starts to run. On the other hand, Operation timer stops.

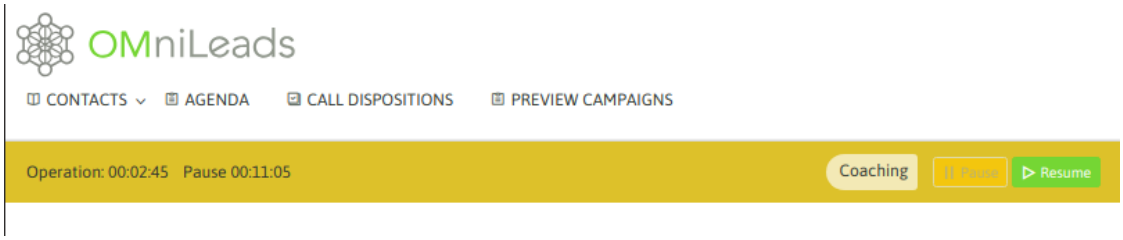


Figure 10: On Pause

Logout

In order to log out of the system, agent needs to click on the exit button as displayed below.

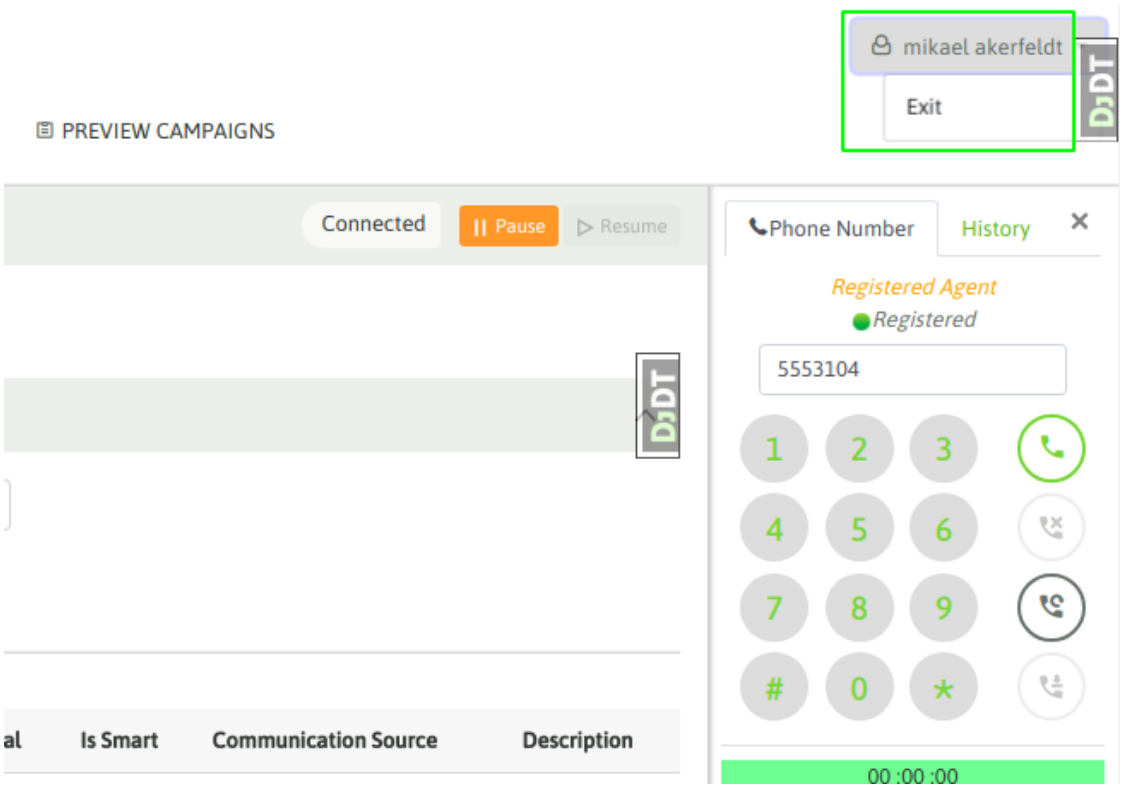


Figure 11: agent logout

12.1.2 Manual Campaigns

Agent and Operation in Manual Campaigns.

Manual Calls from Contact List

When an agent is assigned to a Manual Campaign, she/he can make calls from Contact List, this is: going to menu Contacts -> Contact List and selecting the Manual Campaign where the contact is reachable to be dialed. This can be seen in figure 1.

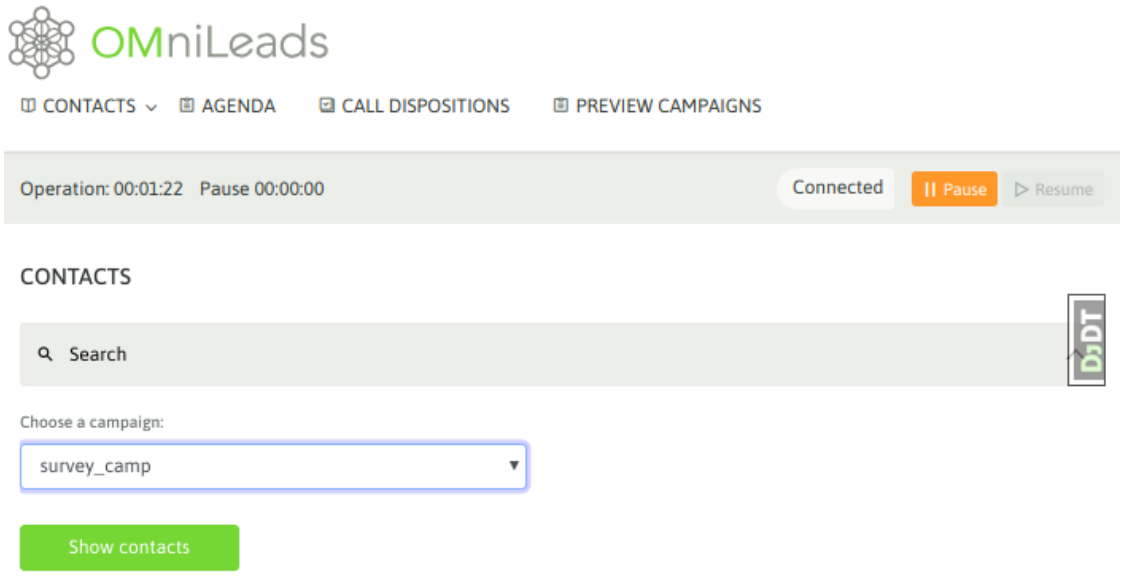


Figure 1: contact list

If agent clicks on Show Contacts, all of them appear listed as shown in figure 2.

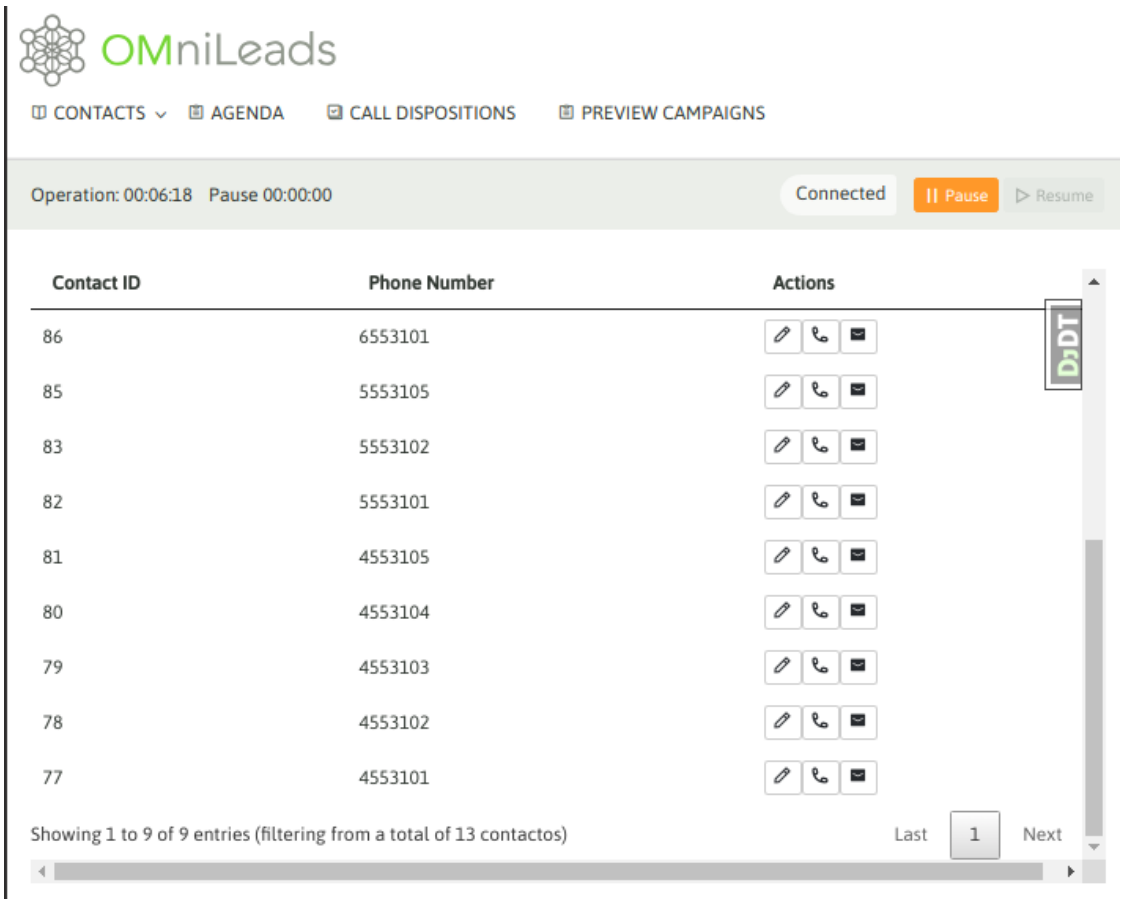


Figure 2: contact list

Then, the agent can make a call to any of the phones listed, by clicking on the phone icon. From that

moment the contact information is presented in the Agent screen and immediately the sytem attempts the dial process.

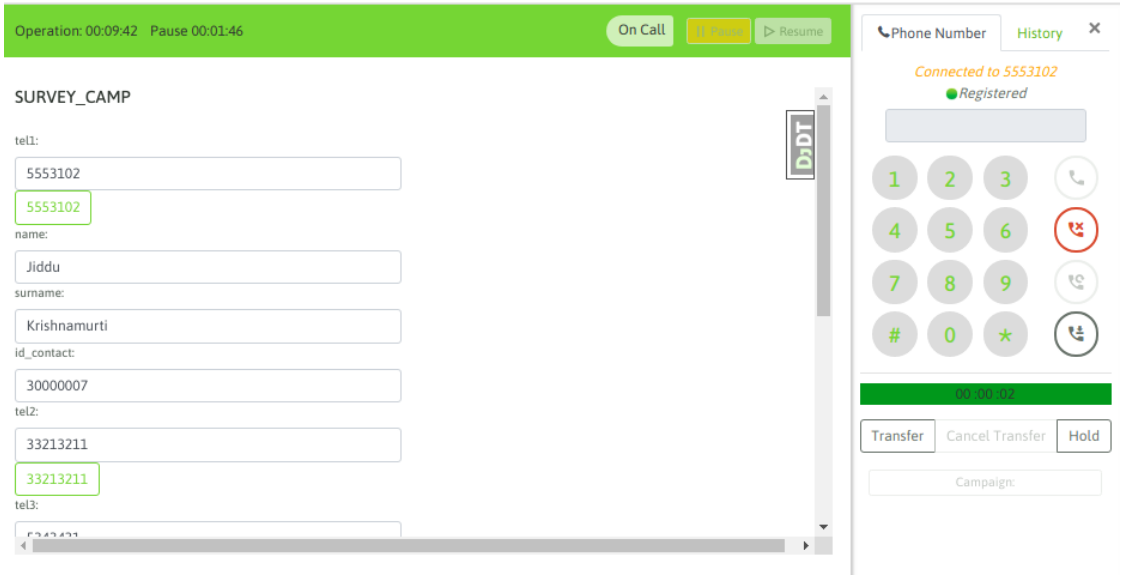


Figure 3: call contact

If the communication has ended or the phone cannot be contacted, then the agent is able to re-attempt with another contact number (if the contact has more than one phone loaded). If this is the case, then the agent can click on any of the extra telephone numbers and will automatically dial the new number.

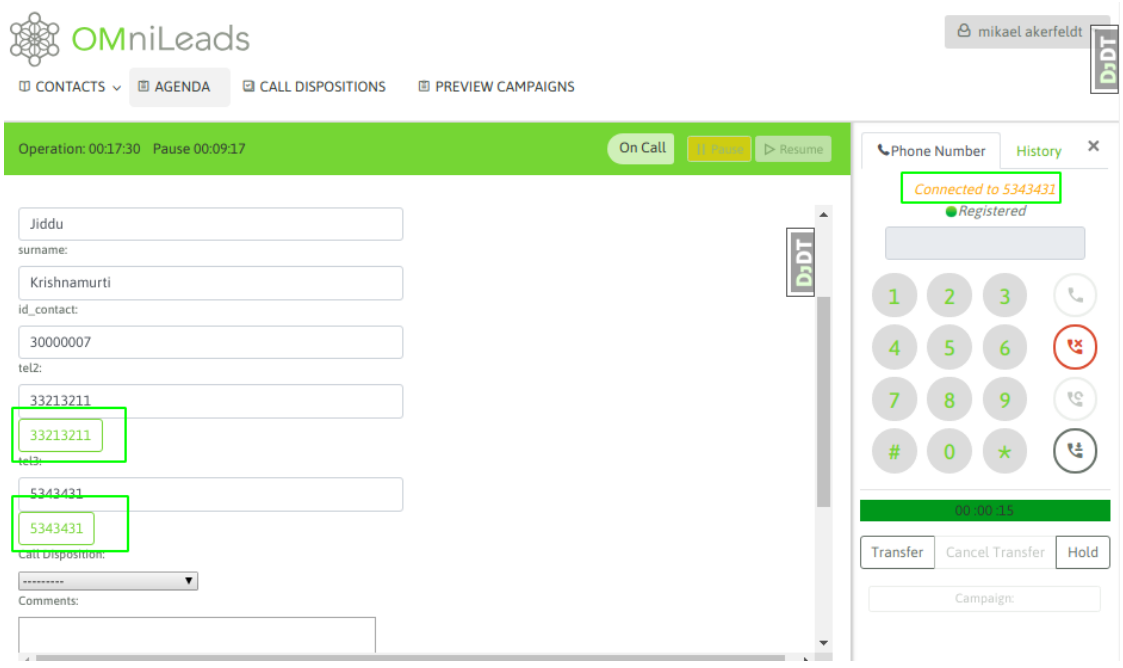


Figure 4: re-call contact

Finally, the agent is able to disposition the call through the Call Dispositions available. This list of Call Dispositions was generated by the administrator for each campaign.

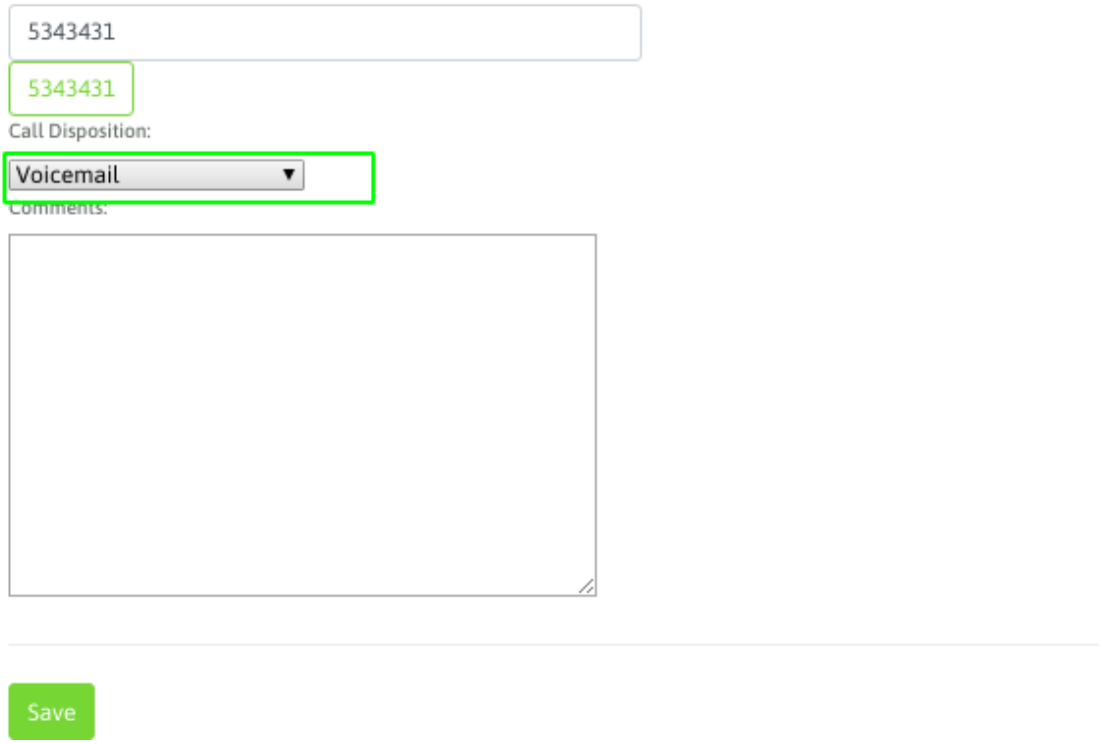


Figure 5: call disposition

Manual Calls dialing from Webphone

The agent can make calls directly from the webphone. It is typical in some Call Centers to distribute contacts between agents using a spreadsheet or looking for the data in an external CRM.

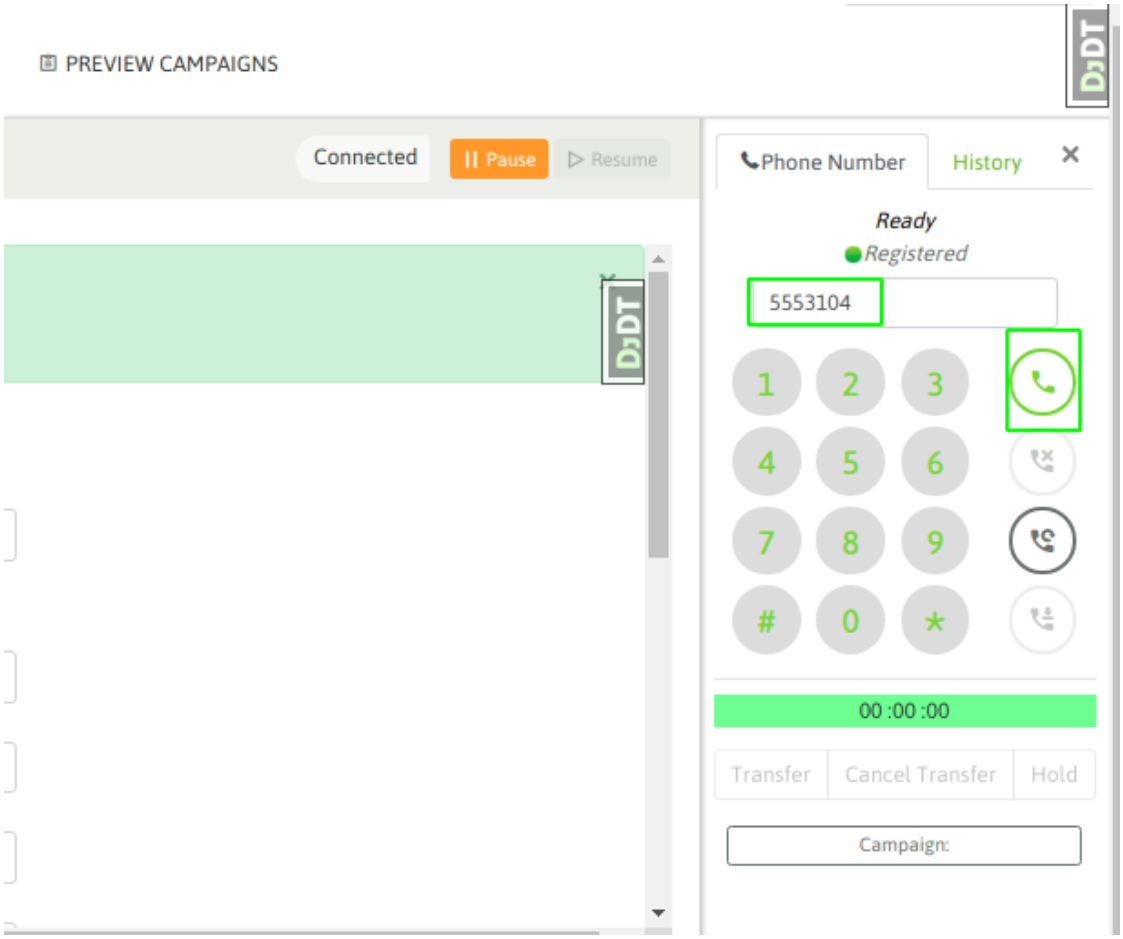


Figure 6: webphone manual call

Just pressing * enter * button or clicking on * dial * button, the call is made. If there is no pre-selected campaign, the system asks to select one prior to attempt the call.

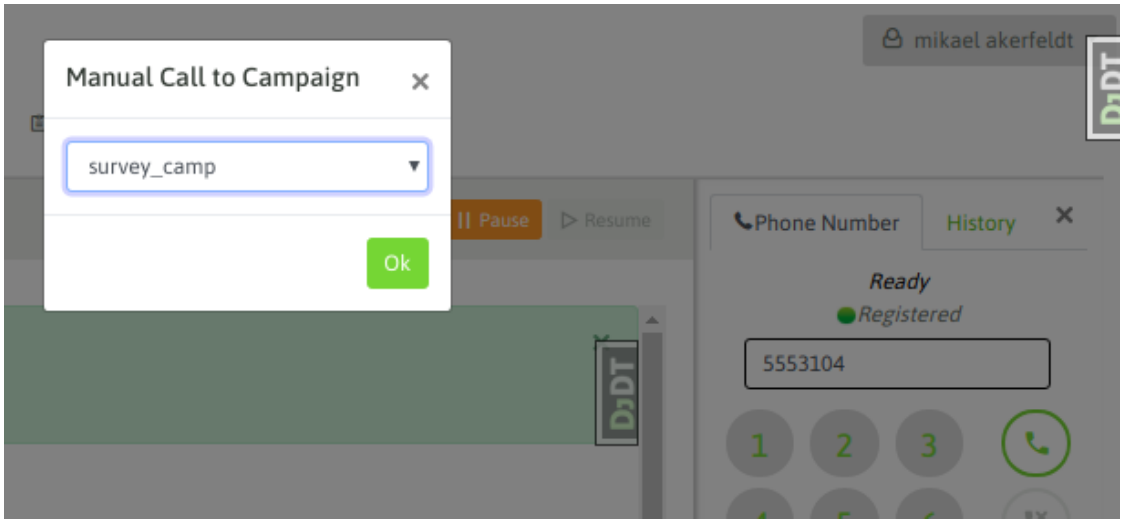


Figure 7: campaign selection

OMnileads makes a lookup of the dialed phone within the Contact Lists assigned. If it exists, all the contacts matching that number are listed. Then the agent selects the contact to dial and the call is launched

presenting Contact Information in the agent console.

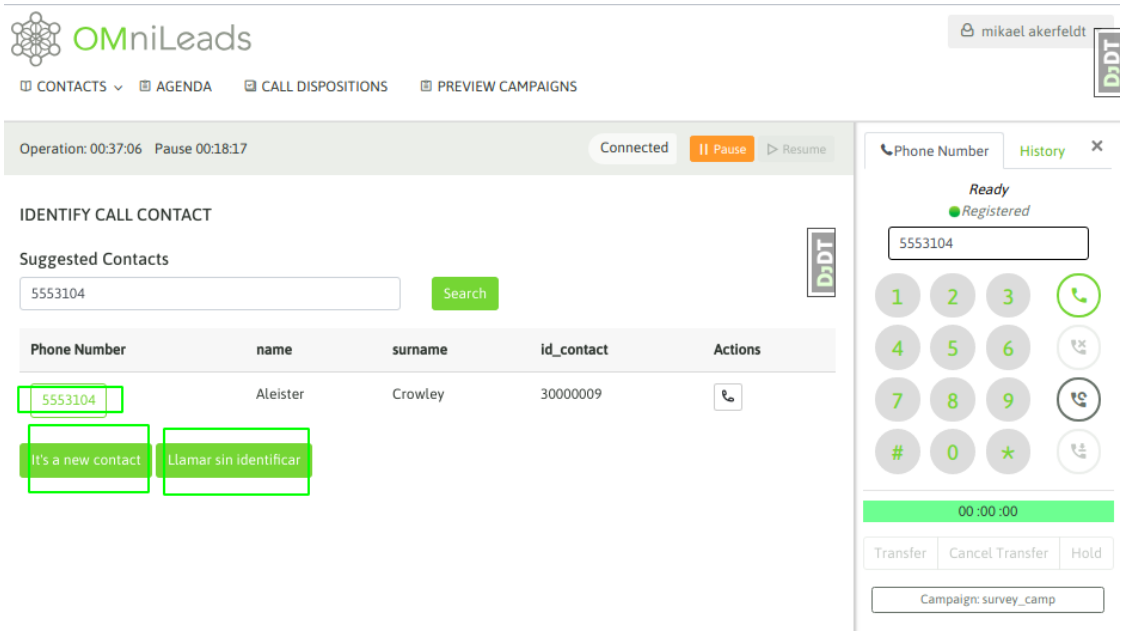


Figure 8: contact selection

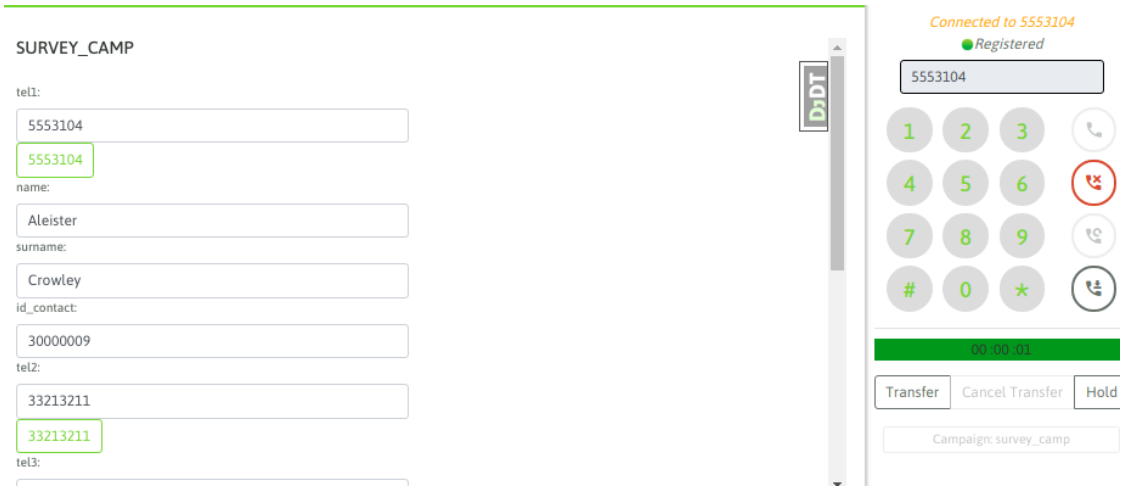


Figure 9: contact call

Manual Calls to Non-Existing contacts

It can also happen that the dialed phone number does not match any contact, as shown in Figure 10.

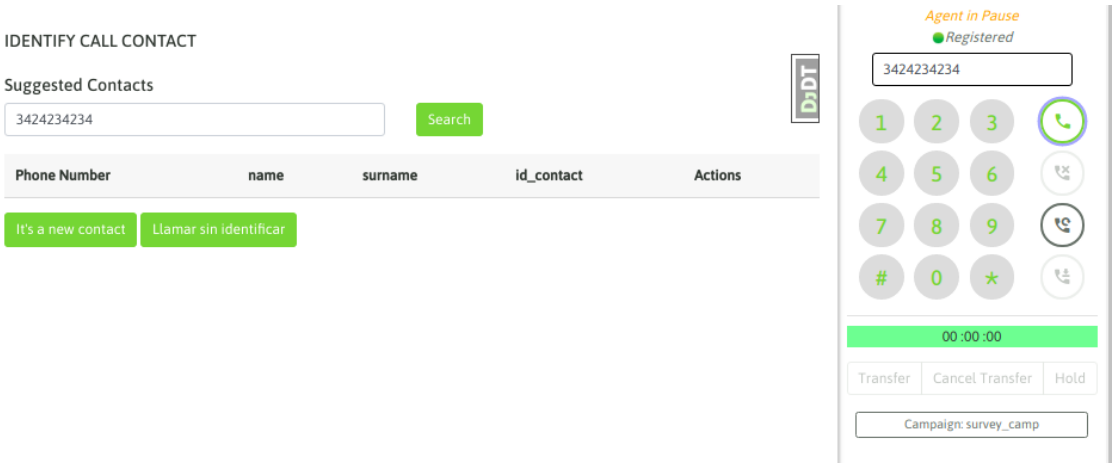


Figure 10: no contact for the number dialed

So, the agent can directly dial the phone and then save the new contact filling the campaign form, as shown in Figure 11.

Figure 11: call without load contact

Or load the new contact to the campaign before dialing the number as indicated in figure 12.

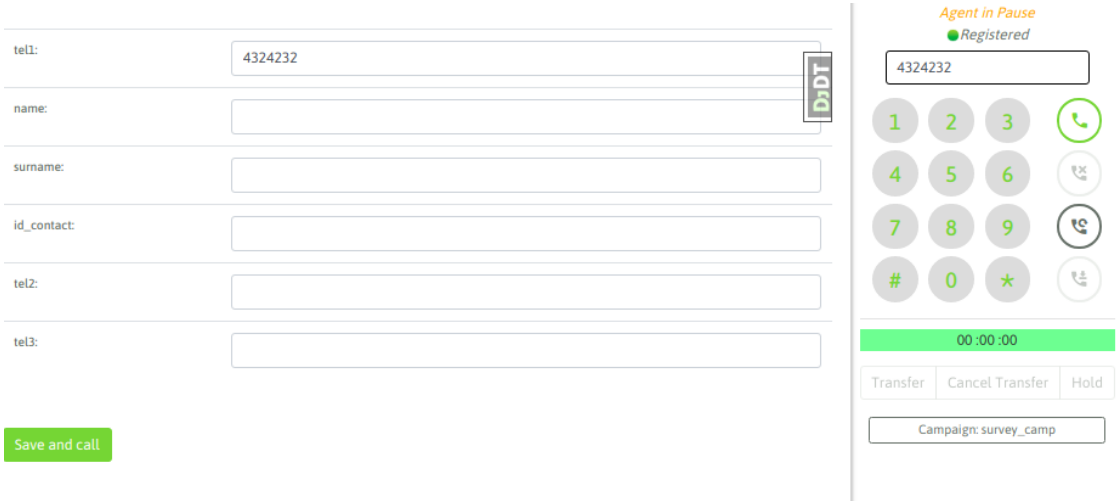


Figure 12: load contact and call

12.1.3 Preview Campaigns

Agent and Operation in Preview Campaigns.

Preview Campagins

When an agent is assigned to a Preview Campaign, Preview menu needs to be used.

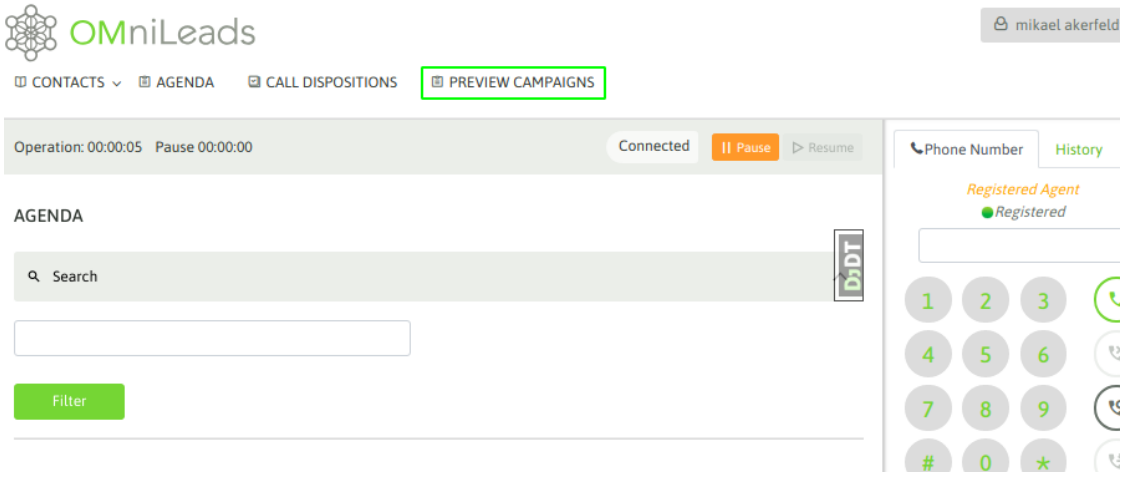


Figure 1: preview campaigns list

All assigned Preview Campaigns are listed here. Agent must select one of them and this will return a contact to manage.

Figure 2: pull new contact

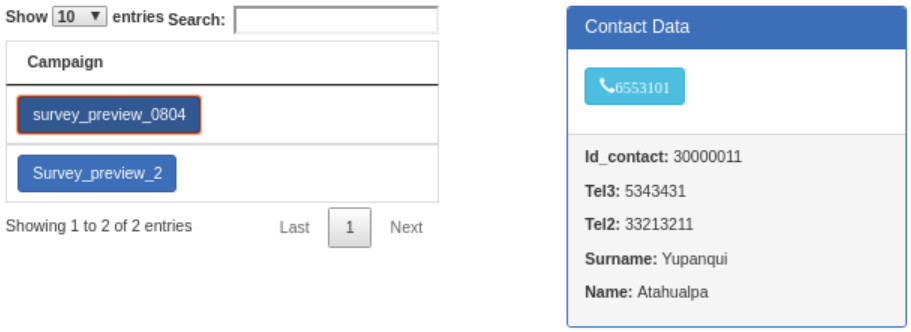


Figure 3: contact delivered

Once the contact is delivered, the agent can either dial the phone number by clicking on the number, or by clicking again on the preview campaign in order to retrieve another contact. The last one will be released.

Assuming a contact is called by clicking on it, contact information is displayed on the agent's view, as we have seen in Manual Calls section.

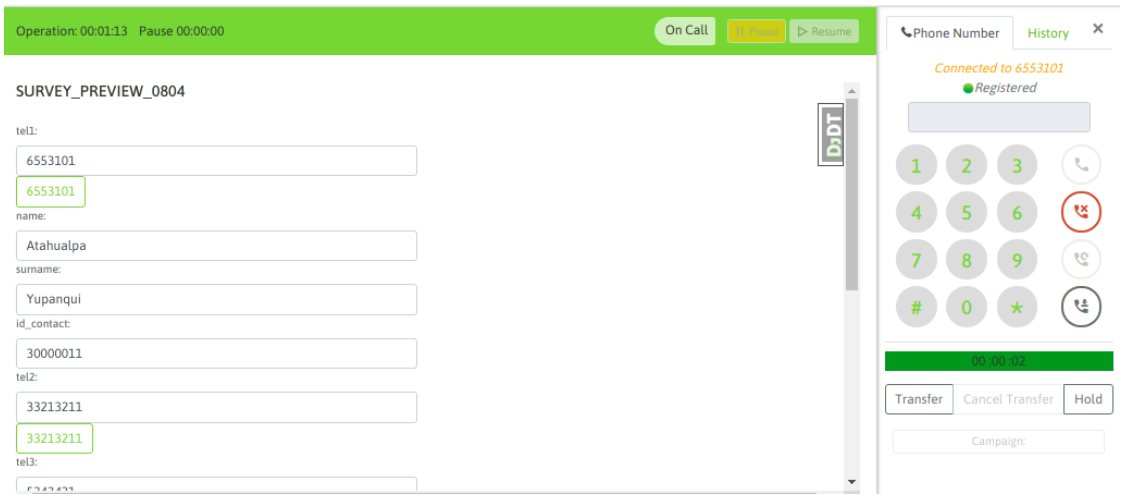


Figure 4: contact called

If the communication has ended or the phone could not be contacted, then the agent can try again with another phone number (if the contact has more than one phone loaded). If this is the case, then the agent can click on any of the extra telephones and will automatically dial the contact to the new number.

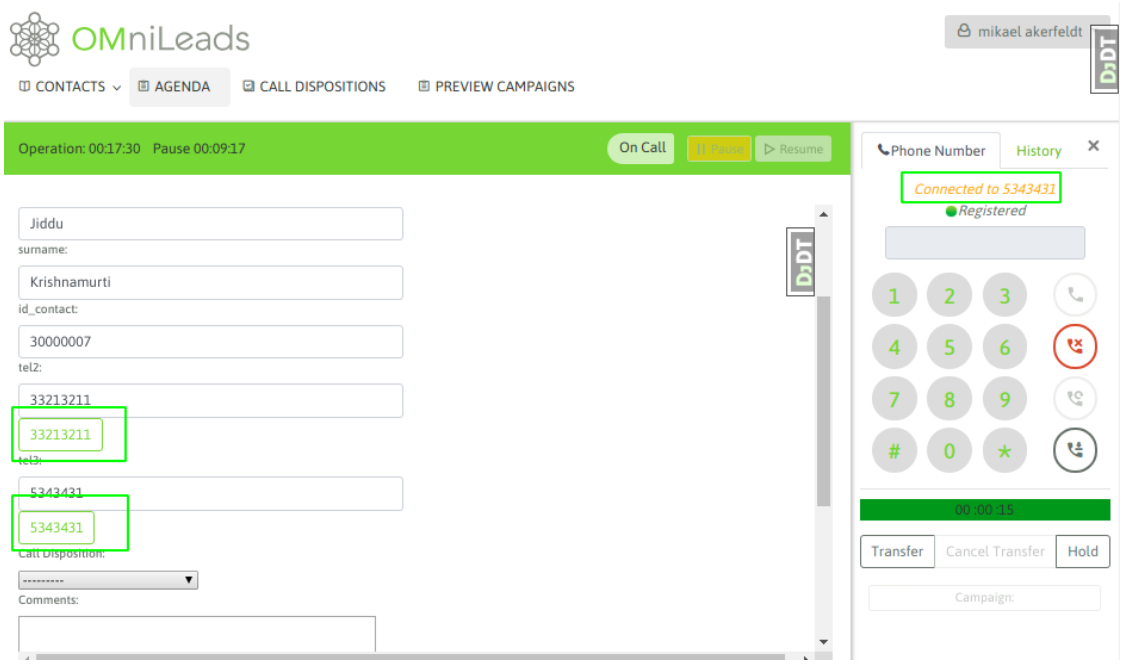


Figure 5: re-call contact

Finally, the agent is able to disposition the call through the combo of Call Dispositions. This list has been generated from the Admin view.

Figure 6: call disposition

12.1.4 Outbound Dialer Campaigns

Agent and Operation in Predictive Dialer Campaigns.

Agent in Predictive Mode

An active Agent assigned to an active Predictive Campaign, is able to receive calls from the dialer module.

The incoming calls are notified with a “beep” audio that is reproduced to the agent headset as soon as the call is delivered. Contact details are also presented to the agent once the call is answered.

Figure 1: dialer call connect

Note: It is important to note that an agent working on a Predictive Campaign needs to be very alert as the system is configured to deliver calls to agents as soon as the contact has been reached and answers the attempt.

In addition to the reproduction of audio notification (beep message), the agent can notice the change in the status bar of the screen. The bar has green color when agent is “On call” and also the name of the campaign the call belongs to can be read.

Once the call ends (regardless of which end finishes the call) , the system forces the Agent status to ACW (After Call Work), which allows the agent to finish filling the contact form with the respective Call Disposition. After that, the agent will change this ACW status to available again , so that the dialer knows she/he is ready to receive new calls.

Note: Agent is able to exit ACW pause status by inducing the agent to leave the pause or after a period of time (in seconds) defined by the administrator. In the latter case, the agent simply has that grace period for call disposition process and then automatically will be put back inReady for new calls.

How to make a Manual Call

There may be cases in which the agent assigned to a predictive campaign needs to dial a number manually. In this case the agent is suggested to enter to pause mode and then from that state, generate the manual

call. Since otherwise it may happen that at the time of being dialing the manual call, the dialer sends a predictive call and drops manual call attempt.

This manual call can be done to other phone numbers of the same contact, or external numbers out of database. The important thing is to remember to go to pause at the time of the manual attempt.

12.1.5 Inbound Campaigns

Agent and Operation in Inbound Campaigns.

Incoming Call Distribution

As we well know from the "Initial configuration" section, the system can be configured for incoming calls to generate a "Ring" on the agent's webphone, giving the possibility of deciding whether or not to answer the call. Or can be configured to directly connect incoming calls to agents (that is to say, auto-answer the call).

Behavior can be seen in figure 1.

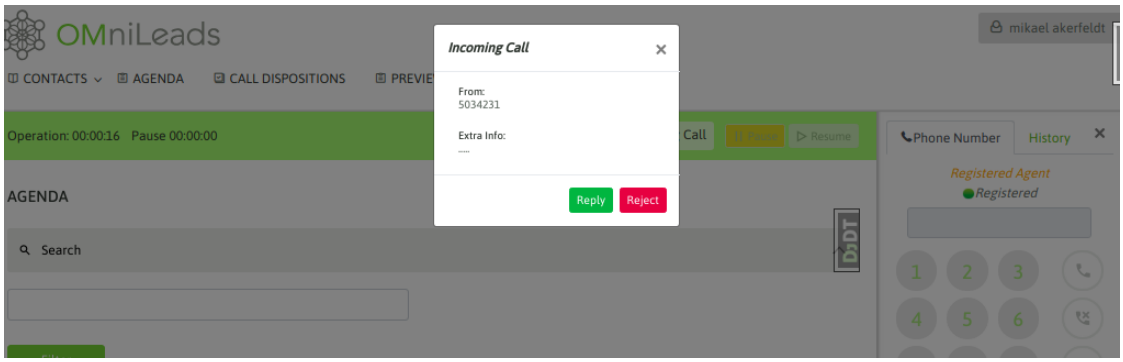


Figure 1: inbound call ring

If the configuration is in auto-answer mode (like a predictive campaign) , the agent will hear a "beep" message once the call is connected and will see the name of the campaign the call belongs to.

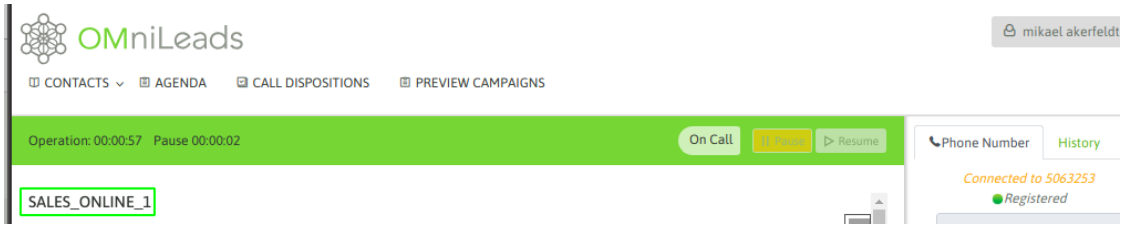


Figure 2: inbound call campaign

Contact Management

Incoming calls can be associated to contacts already available in database, or new contacts to be saved.

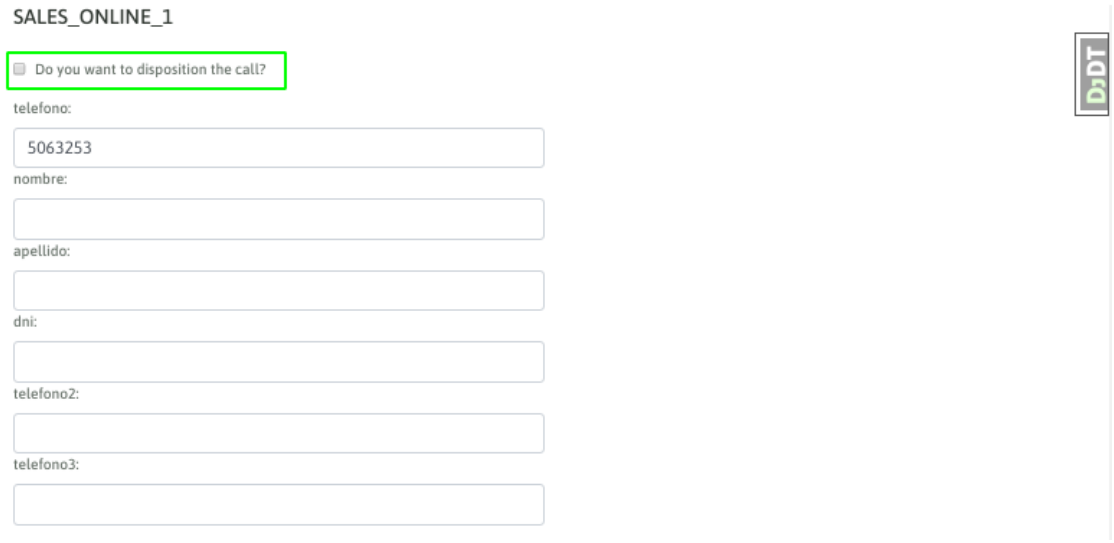


Figure 3: inbound call without contact

In this case, the agent can decide whether to disposition the call or not. In the positive case, then the agent can proceed with the contact information upload and subsequent call disposition.

12.1.6 Call Actions

When Agent is on call, it counts on with multiple features that improve the call management.

Call to an Agent

To make a call to another OMniLeads agent, we must go to the "Call out of Campaign" button available at the bottom of the webphone. When agent clicks on it, a window is displayed in order to facilitate the selection of an agent from a list available and then the call is performed (Figure 1).

Outbound Calls Out of Campaign

Sometimes it is necessary to make a call to a number (subscriber number or extension of the PBX), without the need of managing the contact as if it is part of a campaign logic. This is allowed by clicking on the "Call out of Campaign" button available at the bottom of the webphone. The displayed window has a field to enter the number to be dialed (figure 1).

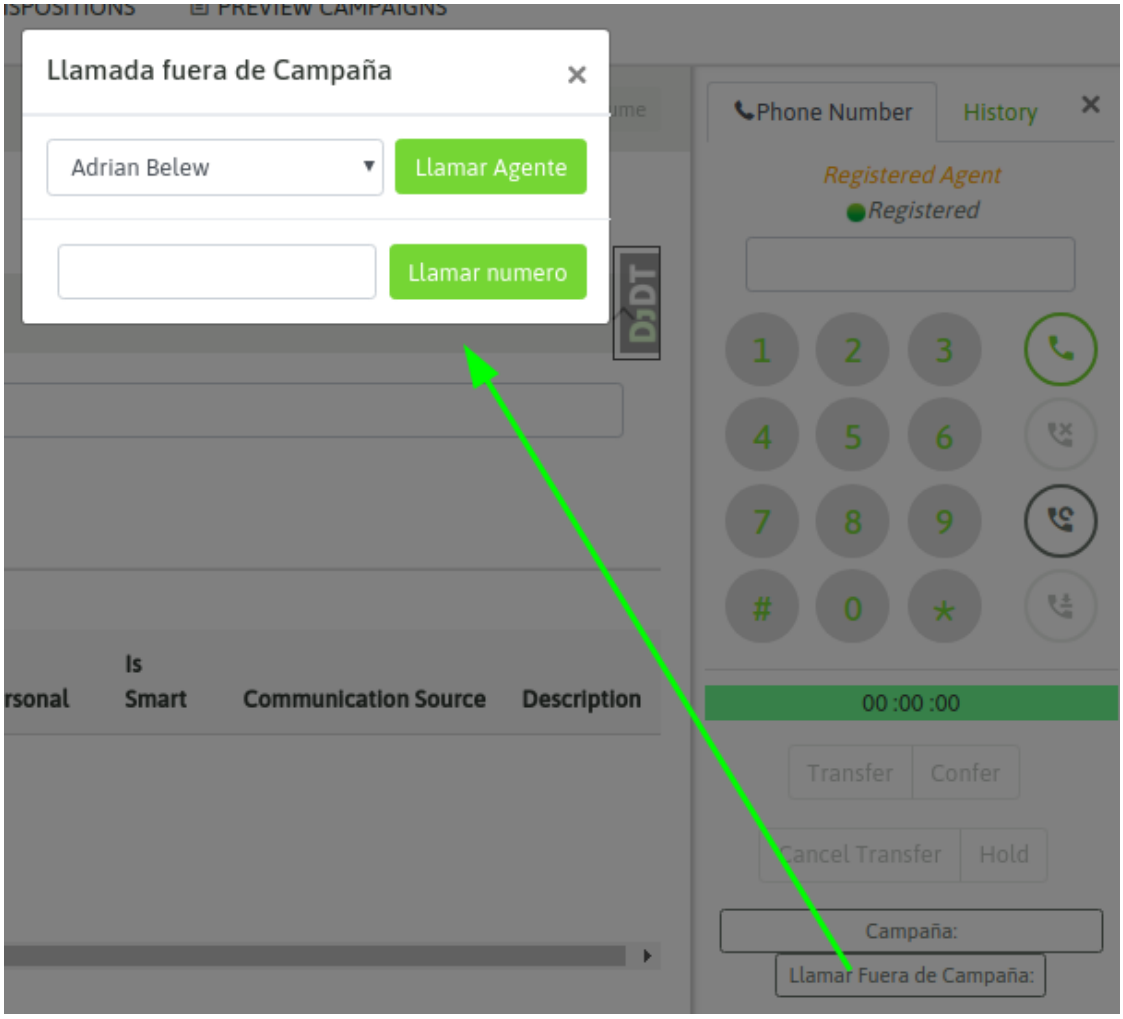


Figure 1: without camp calls

Put a call On Hold

During a call, the agent has the ability to hold the call for a while. This is possible by clicking the “hold” button in the webphone.

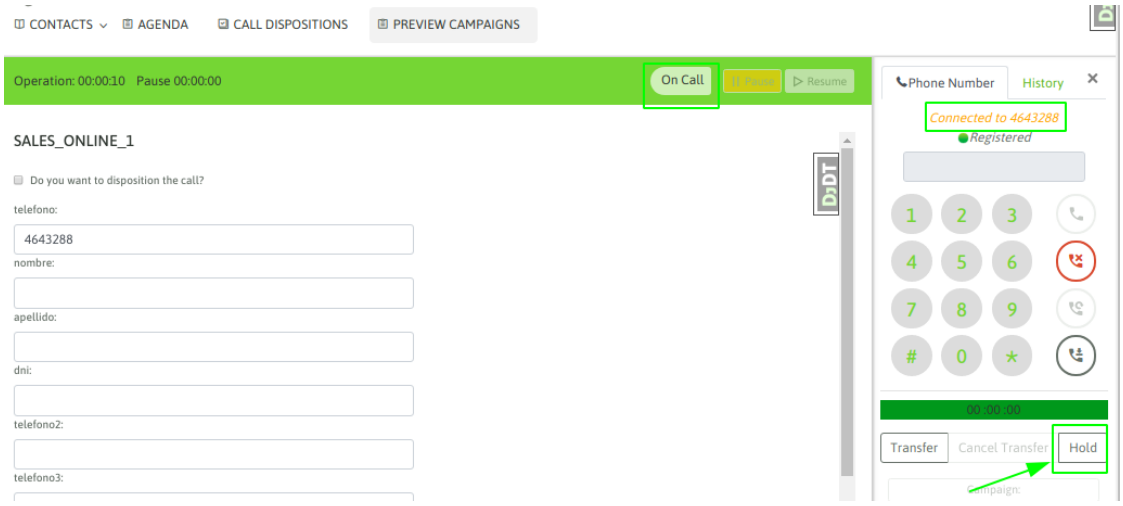


Figure 2: call on hold

When hold button is clicked, the customer will listen hold music, while the agent can unhold the call when he wishes, justo clicking in unhold, as seen in figure 2.

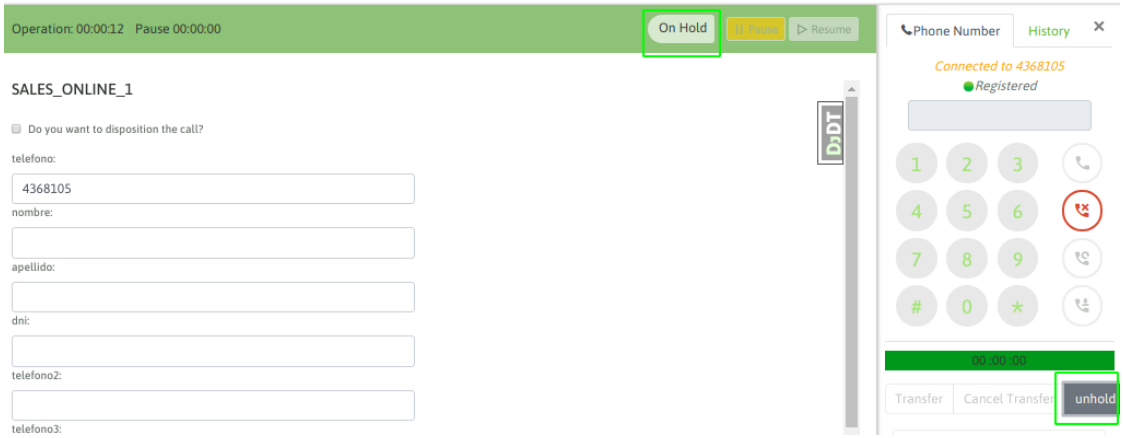


Figure 3: unhold

This functionality can be used with any type of call.

Call transfer and Conferences

Within the range of possibilities for call transfers that can be made in the system, we have the following:

Blind Transfer to another Agent

Supose that “agent A” has an active call and she/he wishes to transfer the call to “agent B”, directly. In this case agent clicks the transfer button, inside the webphone and then selects “blind transfer”. After that, she/he can select the agent.

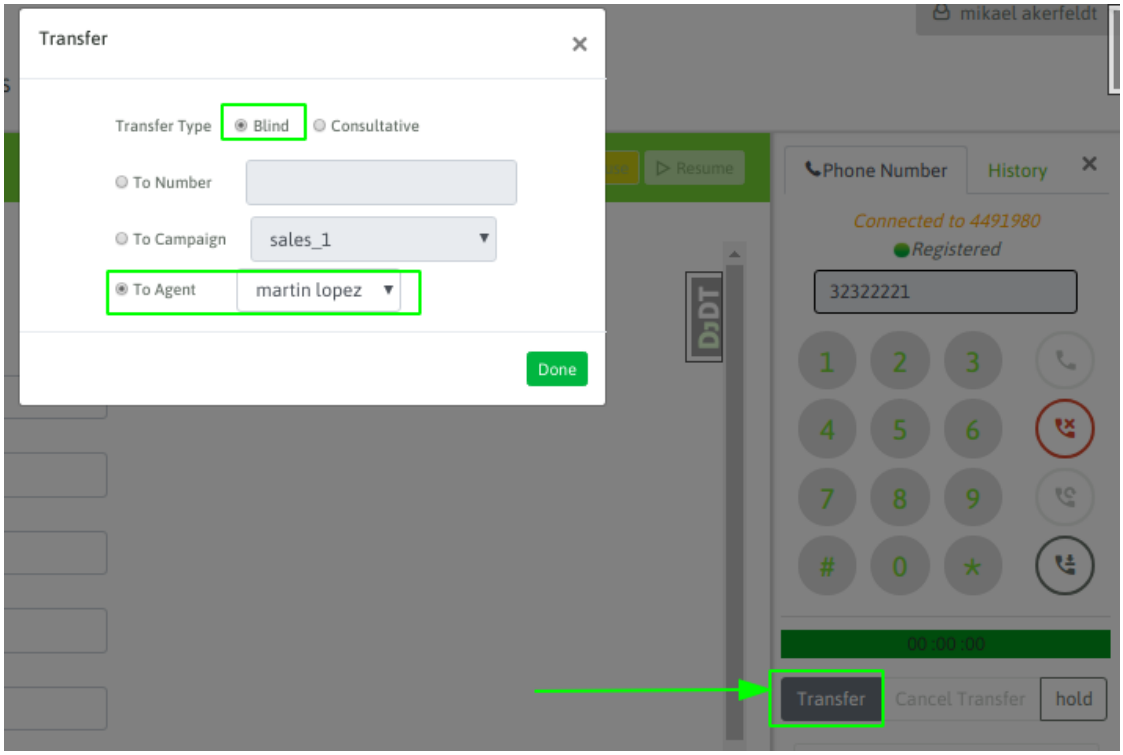


Figure 4: "Agent A" to "Agent B" blind transfer

In this case the call is automatically transferred to "agent B", leaving free "agent A". Once the transfer is made, it cannot be recovered, neither "agent A" can know if the transfer was attended by "agent B".

Blind transfer to an external number

The "Agent A" is on an active call and wishes to blind transfer the call to an "external number". When we say external, we mean a call that is generated outside the system. It can be a PBX extension within the company or an external telephone number of the PSTN.

In this case, the transfer button is clicked and then agent can select "blind transfer" as the type of transfer. You must enter the destination number in the box as indicated in the figure 5.

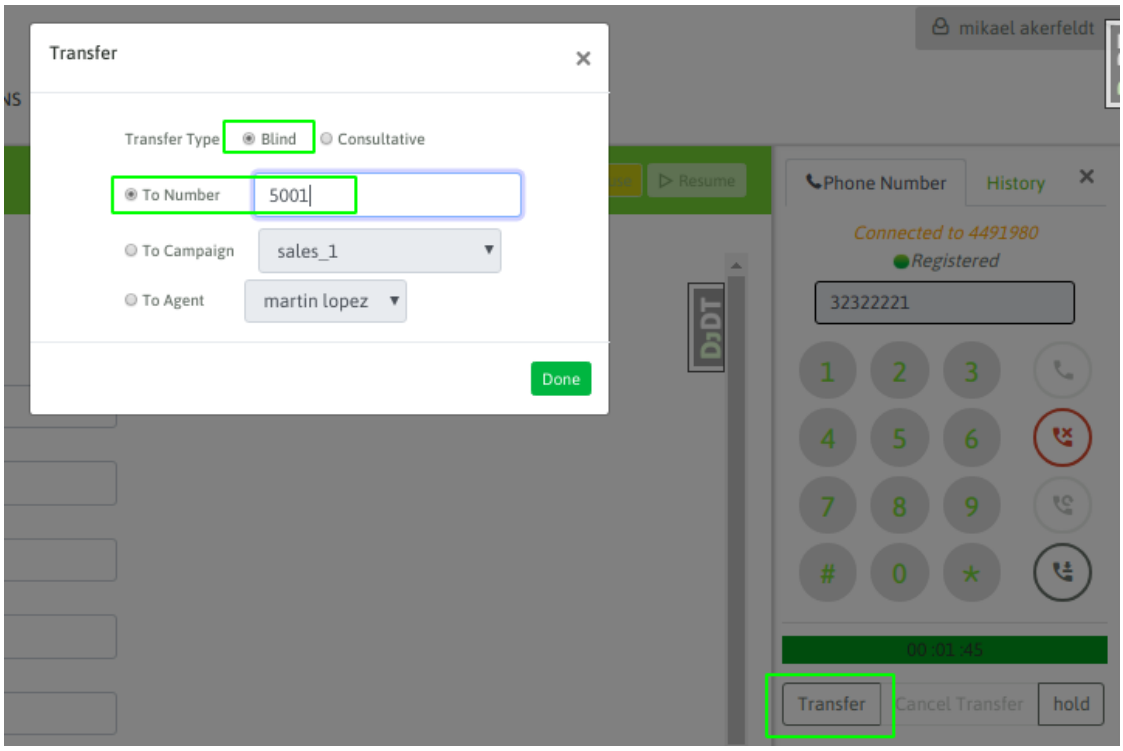


Figure 5: "Agent A" to "External Number" blind transfer

In this case, the call is automatically dispatched to the phone destination, and the webphone of the "agent A " is released. Once this transfer is performed, the call cannot be retrieved to original source, and "agent A" cannot know if the call was attended or not by the transfer destination phone.

Consultative Transfer to another Agent

The "agent A" is on an active call and wishes to transfer the call to the "agent B" in a consultative manner; that is to say that the external telephone is put on hold while the "agent A" opens a new channel to the "agent B". If the call between them is established and the "agent B" wishes to receive the transfer, then the "agent A" drops the call and automatically the external telephone is bridged with the "agent B".

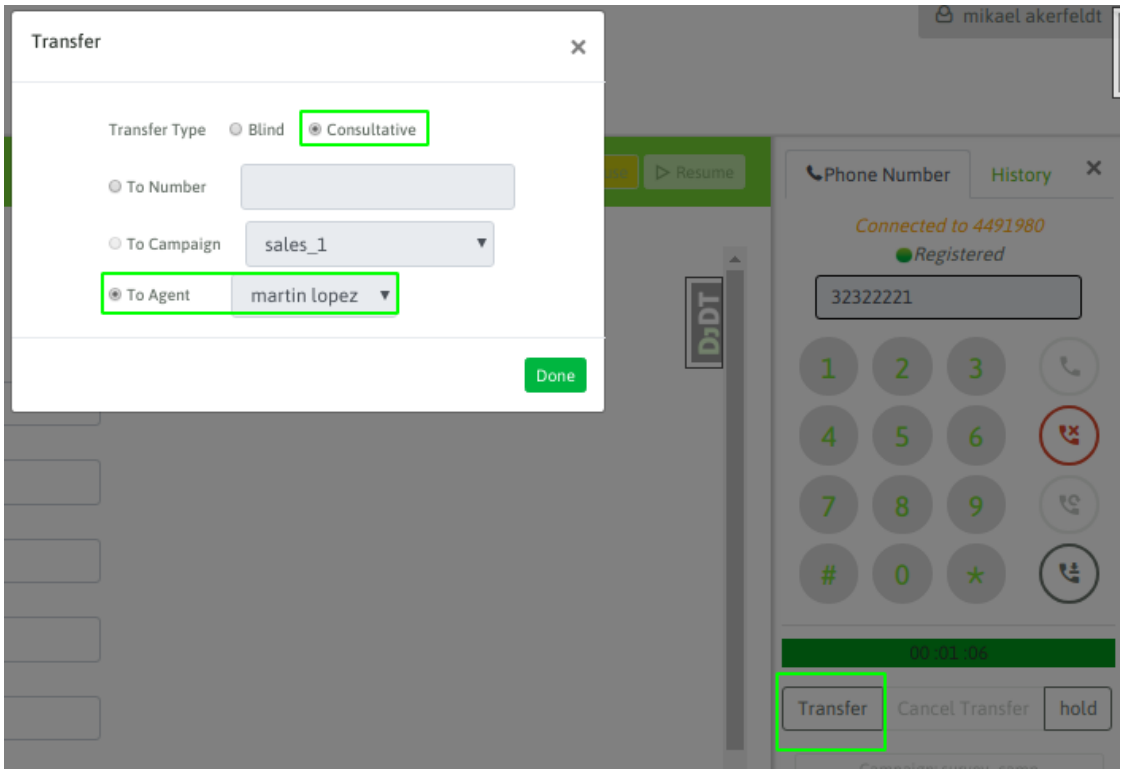


Figure 6: "Agent A" to "Agent B" consultative transfer

In this scenario, another possibility is:

- That it is not possible to contact the "agent B", then the "agent A" can cancel the transfer during the ringing phase to the "agent B" using the button "Cancel Transfer" of the webphone.
- The contact with the "agent B" is achieved but the last one cannot (or does not want to) proceed with the transfer. Therefore the "agent B" must drop the call and automically the "agent A" remains with the connected conversation.

Three Way Conference: external number, Agent A and Agent B

This is a possible scenario within a consultative transfer, since the action to be executed by the agent that drives the conference (agent A), is initially a consultative transfer. At the time of establishing conversation between "agent A" to the "agent B" (while the external person or external number is on hold) the "agent A" must click the "Confer " button available on the agent Webphone and thus remain the three parties in a conference call.

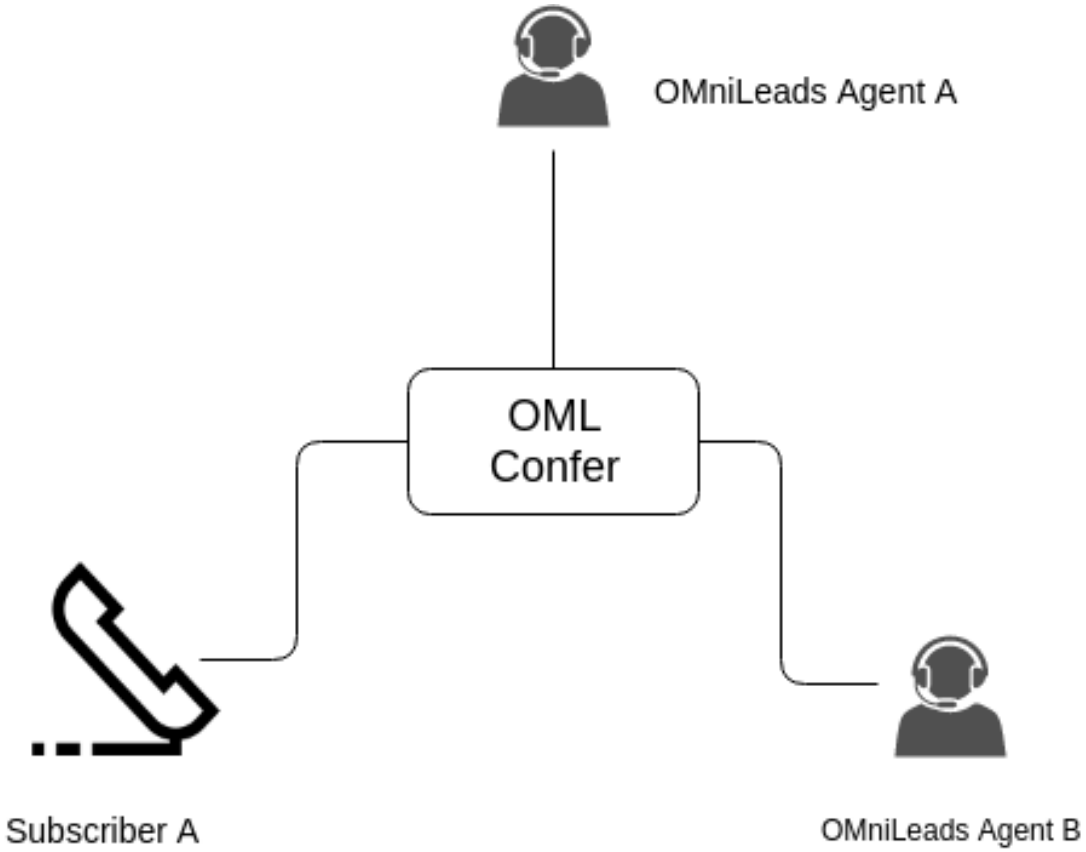


Figure 7: "Agent A", "Agent B" and External number three way conference

Consultative Transfer to external number

The "agent A" is on an active call and wishes to transfer the call to an external "telephone number" for consultation purposes, putting "external phone A" on hold while the "agent A" opens a new channel to the "external telephone B". If the call between them is established and the "external telephone B" wishes to receive the transfer, then the "agent A" drops the call and automatically the "external telephone A" is joined to the "external telephone B".

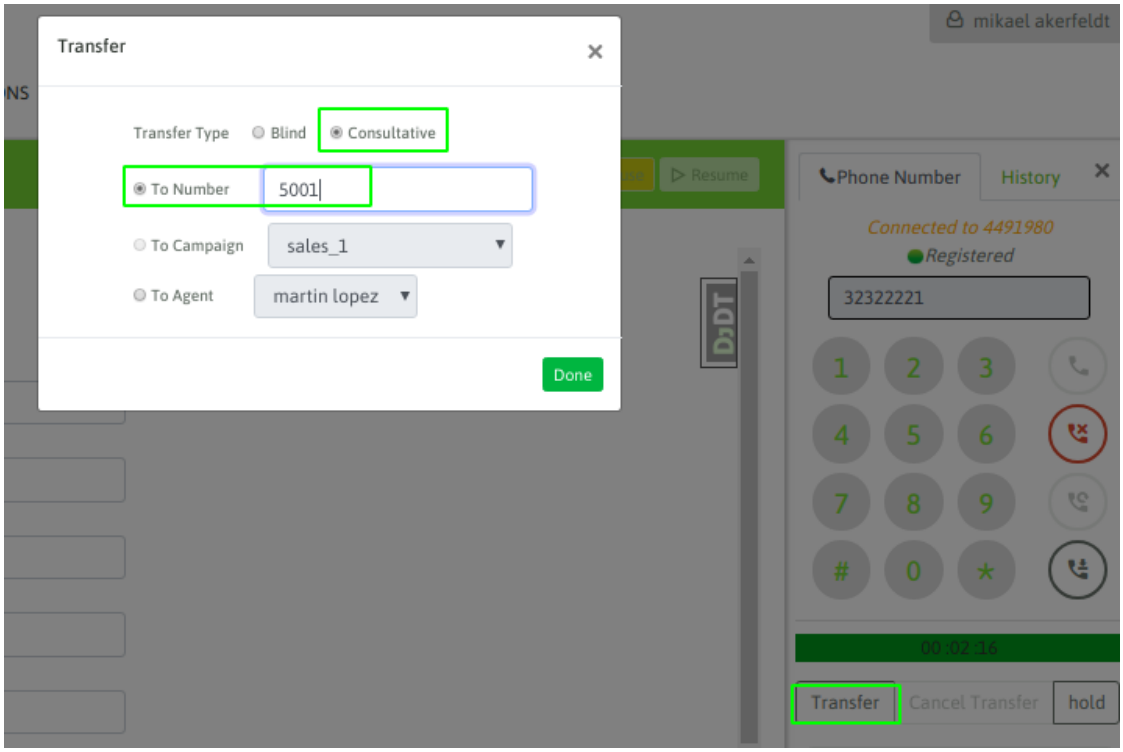


Figure 8: "Agent A" to "External telephone" consultative transfer

In this scenario, another possibility is:

- That it is not possible to contact the "external telephone B", then the "agent A" can cancel the transfer during the ringing phase to the "external phone B" using the "Cancel Transfer" button on the webphone.
- The contact with "external telephone B" is achieved but this one cannot (or does not want to) proceed with the transfer, therefore the "external telephone B" hangs up the call and the "agent A" remains connected to "external telephone A".

Three Way Conference: external telephone A, Agent, external telephone B

Under this scenario the "agent A" can set up a three-way conference between the "external number A", that is the person who initially established the call with "agent A" and an "external number B", that can be the extension of a PBX or a subscriber of the PSTN. In such a way, all the parties remain in a conference room.

In order to perform this action, the "agent A" must initiate a *consultative transfer* to the "external number B" and once in call the agent must click on the "Confer " button of the webphone (Figure 9).

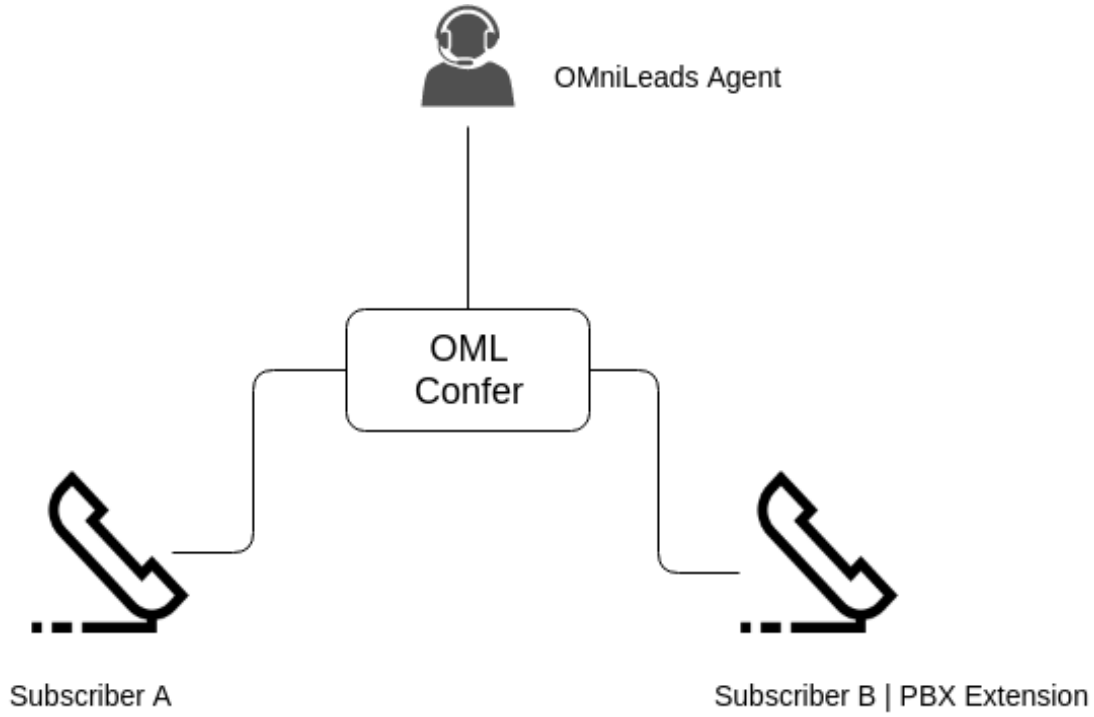


Figure 9: "Agent A", "Subscriber A" and "Subscriber B" three way conference

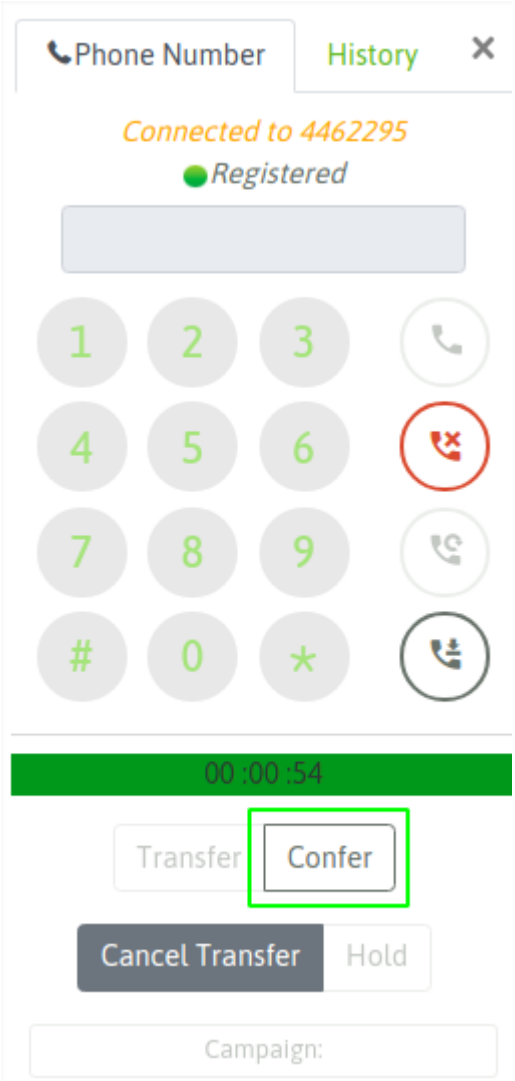


Figure 10: Webphone confer switch

Transfer to another Campaign

Under this scenario the “agent A” is in an active call and wants to transfer the call to an Inbound Campaign. Agent must select “blind transfer ” option since the call is transferred to the queue of the target campaign.

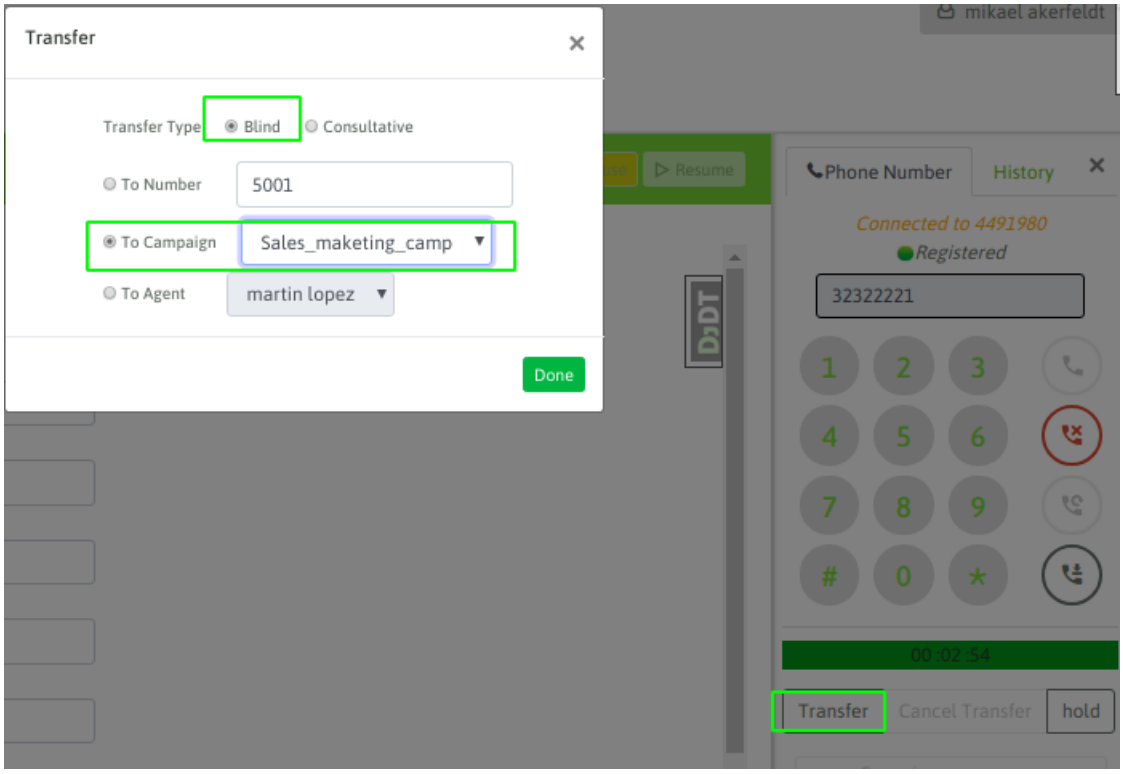


Figure 11: "Agent A" to "inbound campaign" transfer

As it is a blind transfer, the call is automatically dispatched to the destination number, while the "agent A" webphone is released. Once this transfer is triggered, it cannot be retrieved, and "agent A" cannot know if the call was really answered.

Tag a Recording Call

This functionality of the agent webphone allows to generate a mark on a recording call. The idea for this flag is to be reachable from Recording Module.

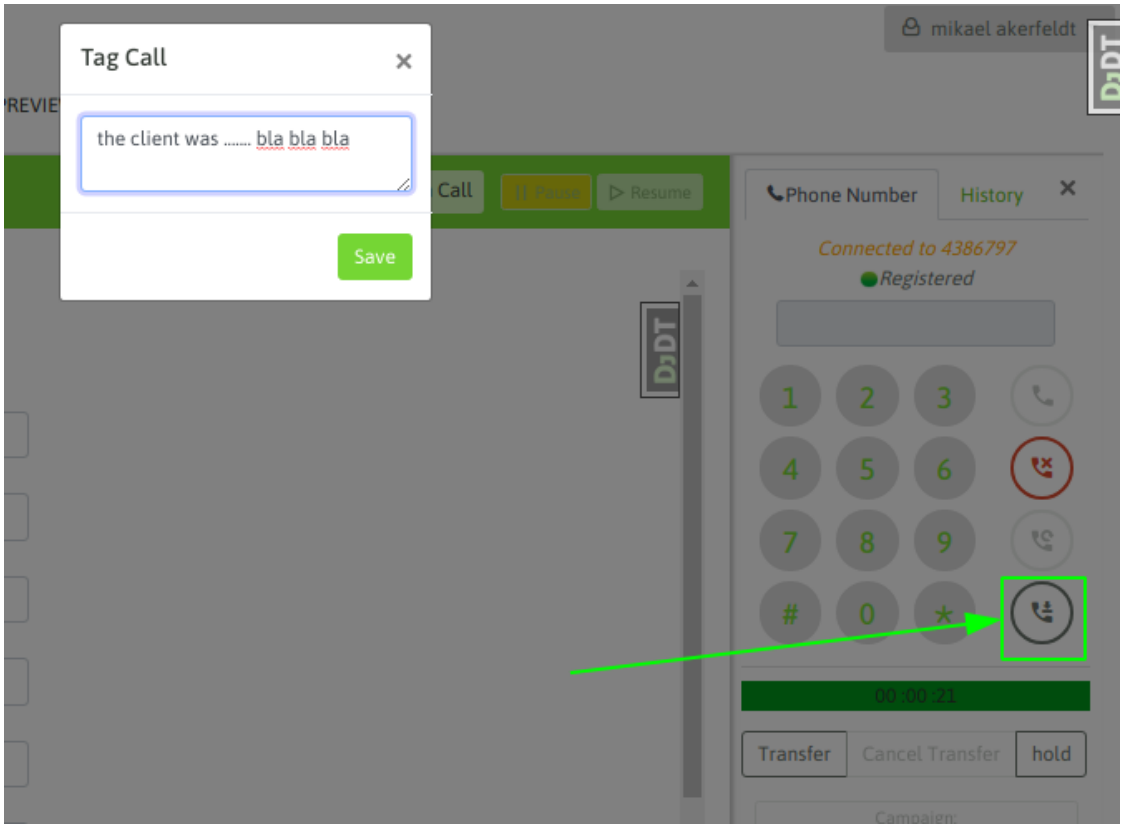


Figure 12: call recording tag

As shown in figure 12, after clicking on the button to mark the call, a text field is displayed so that the agent can describe the situation or add a specific comment.

Finally, using the OMniLeads recording module, you can recover that recording and see what the agent has written on it.

Scheduled Callbacks

Scheduled Callback functionality allows the system to call again a specific contact based on time and date. The idea is not to discard it, but keep managing it.

Personal Callback

When the agent requires calling a specific contact again, she/he is able to generate a reminder in the personal agenda, and then listing them in order to call them back.

Agenda is a default call disposition of the system.

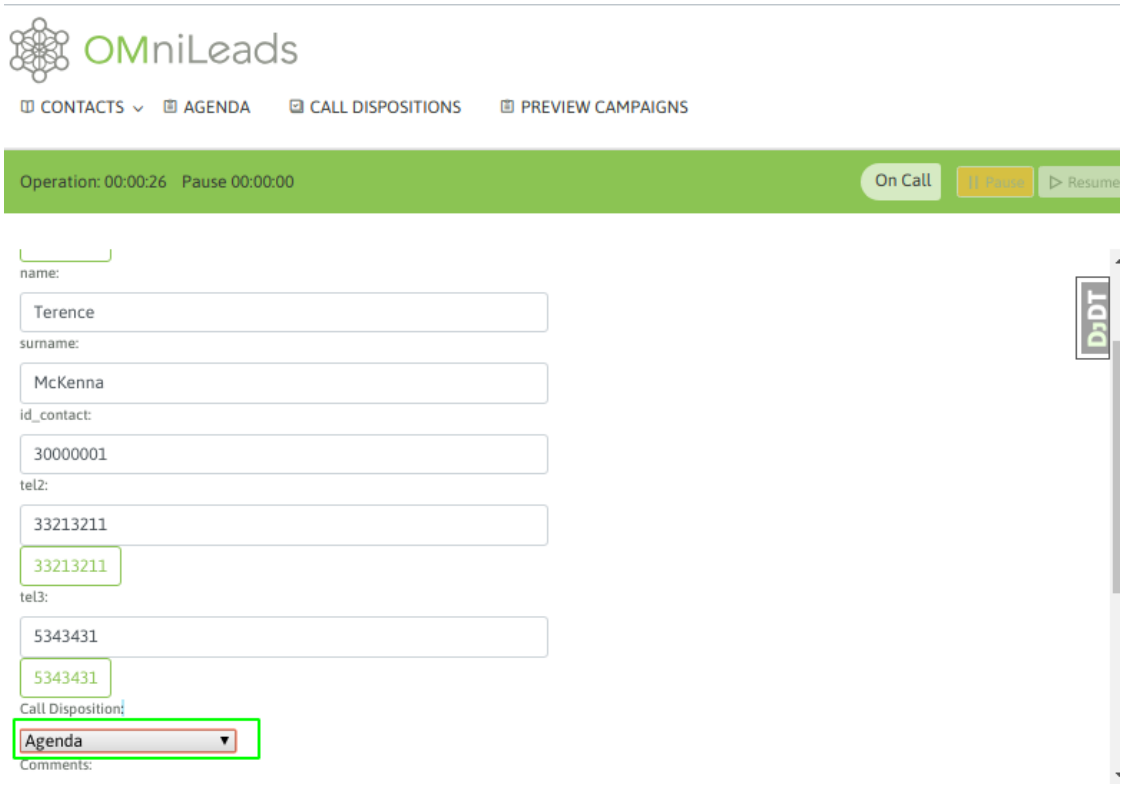


Figure 13: Personal Callback

After call is dispositioned, a form is displayed for selecting the date, time and reason of the agenda being in place.

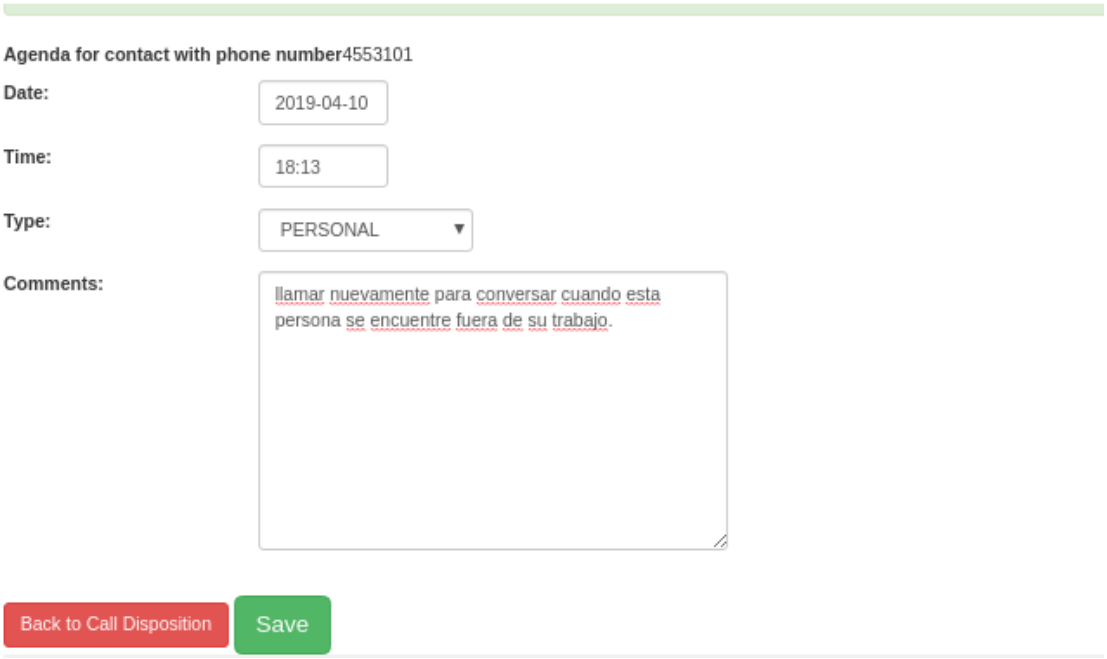


Figure 14: Personal Callback details

Finally, the entry in the Personal Agenda remains available and can be seen by clicking in the Agenda

menu.

Figure 25: Agenda details

Global Callbacks for Outbound Dialer Campaigns

Global Callback is only applicable to Outbound Dialer Campaigns as it is intended to re-insert the contact within the list of numbers to be called by the dialer. In this scenario the dialer simply calls the scheduled number.

This is a functionality that allows agents to schedule a contact, since contact wants agent to call again later, maybe because an impossibility to attend the call in that moment.

To create a Global Callback, you must disposition the call by selecting “Agenda” Call Disposition. After that, agent must select “Global” as the type of agenda.

12.1.7 Agendas, Call Dispositions and Contacts

Agent has different menues to work with campaign contacts.

Campaign Contact List

The agent can list all the contacts of each campaign to which it is assigned. This is achieved by entering the menu item “Contact - Contact list”.

A contact view is displayed, where agent is able to select the campaign to look for contacts.

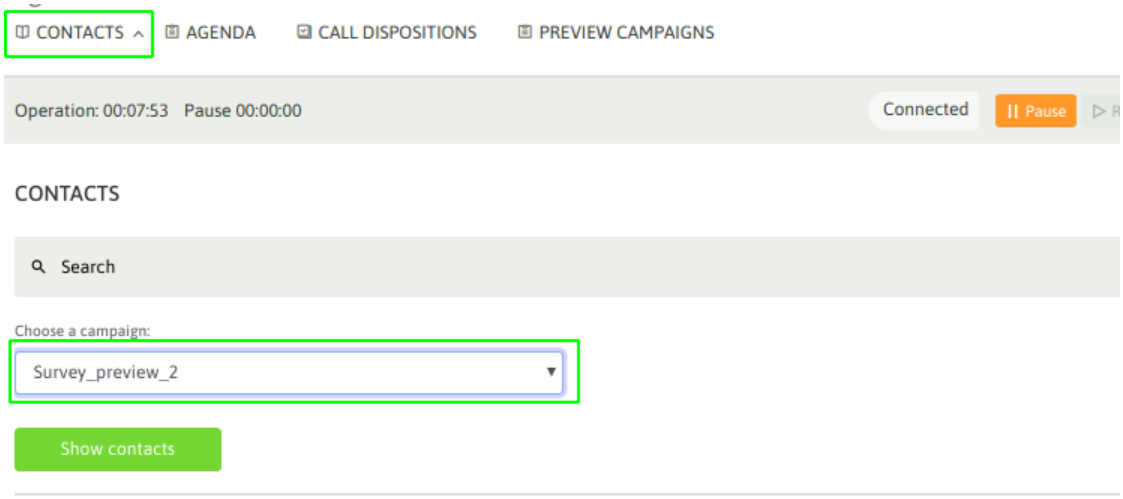


Figure 1: campaign contact list

By listing all contacts in the campaign, the agent can go through each of them or perform a search by contact id, telephone, name, surname, etc.

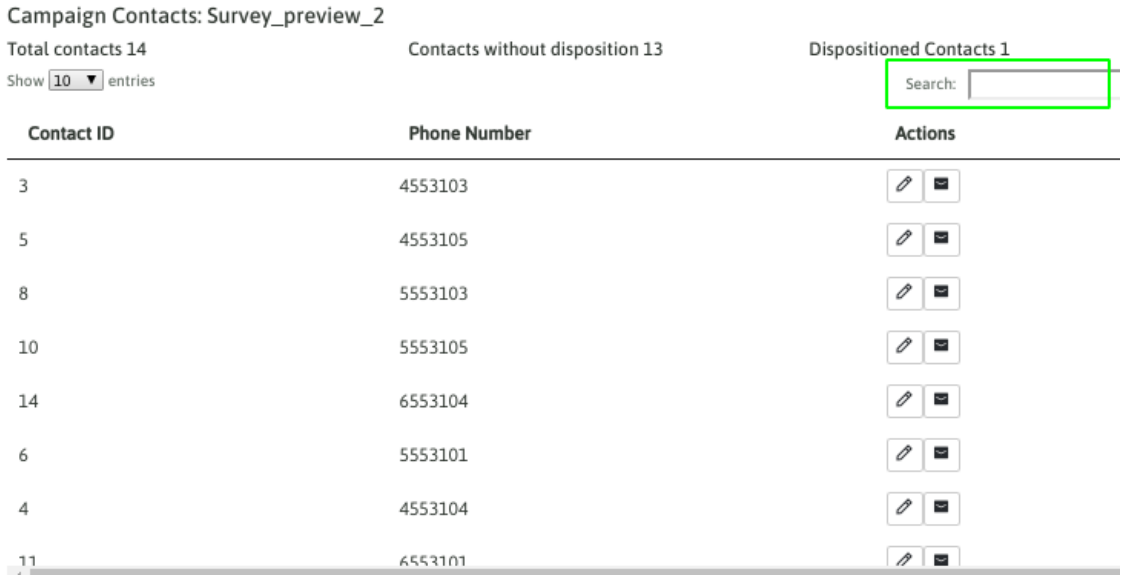


Figure 2: contact search

For example, agent can search for a phone number in the list of contacts, as shown in figure 3.

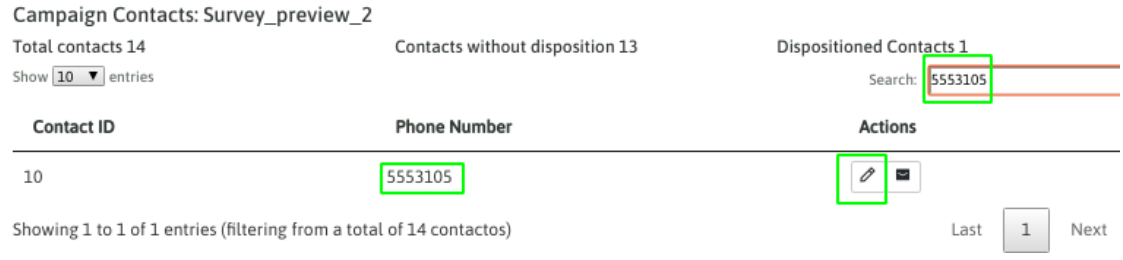


Figure 3: contact search

The view allows to edit/modify any of the contacts listed.

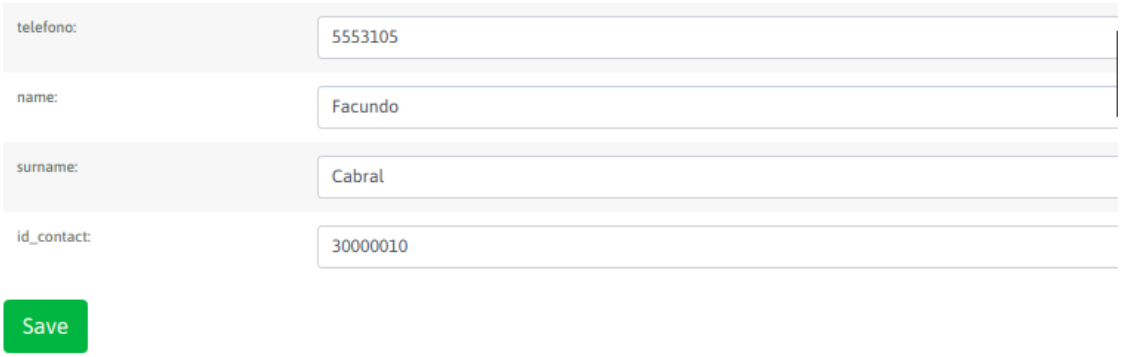


Figure 4: contact search

Pending Agendas

The agent can access her/his phonebook of pending calls. This section lists all the entries that the agent has registered during operation.

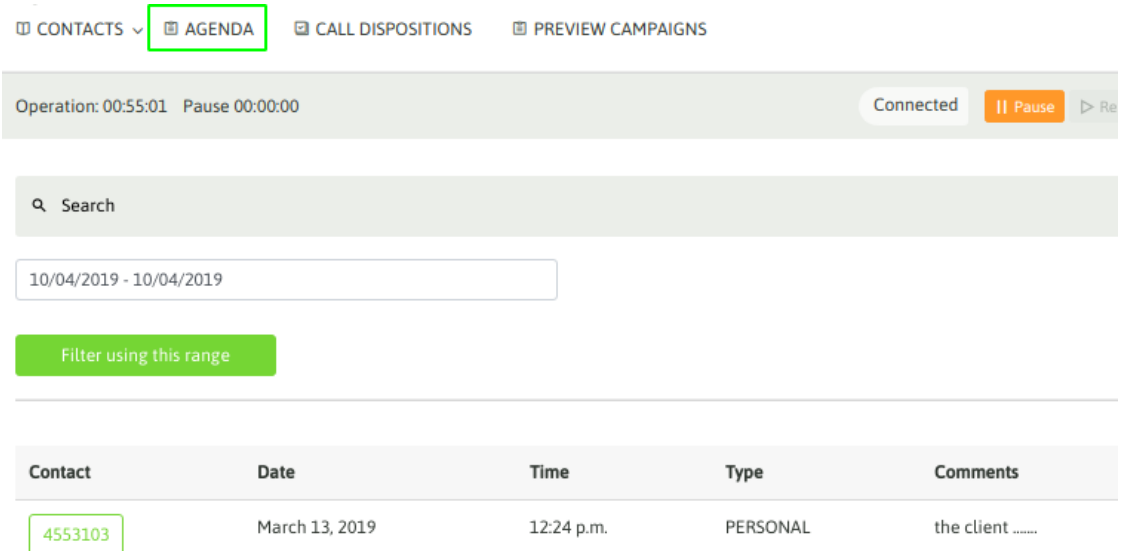


Figure 5: Agenda

The agent has each scheduled contact and its description. By clicking on the phone number, it automatically triggers a dial attempt to the contact's phone.

Agent Call Dispositions

In this menu, the agent can list all the dispositioned calls at a historical level. Therefore the agent counts on with a backward control of each managed contact.

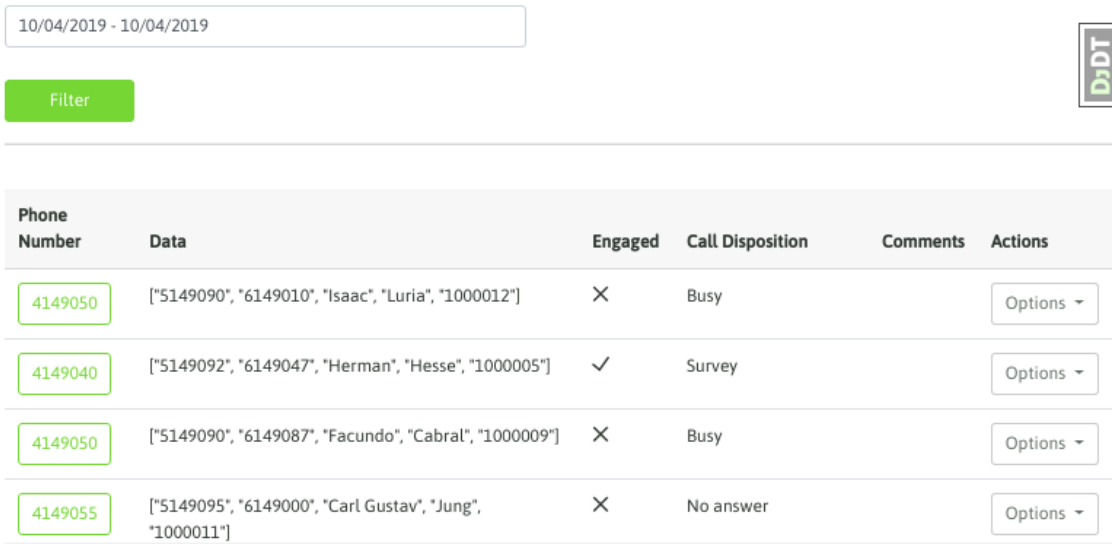


Figure 6: call disposition list

The search process can be filtered by date and the agent can also access to the selected contact to review their data or modify the call disposition.

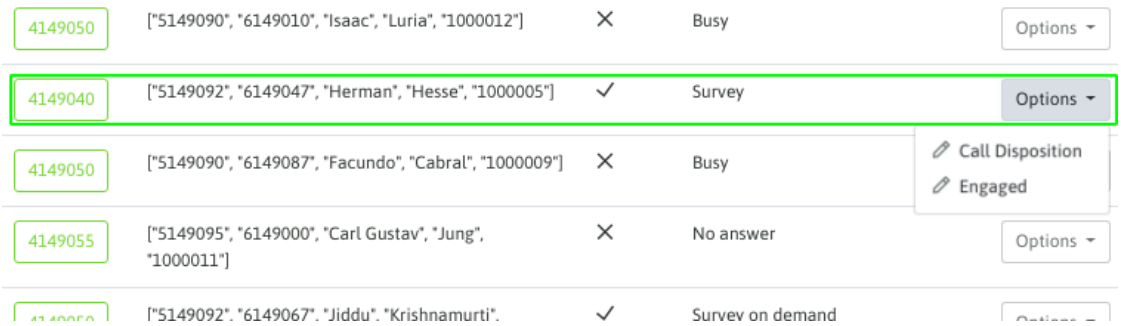


Figure 7: call disposition edit

As you can see, the agent can modify the call disposition of a contact, and the form data in the case the contact is dispositioned with a call disposition of type “Engaged”.

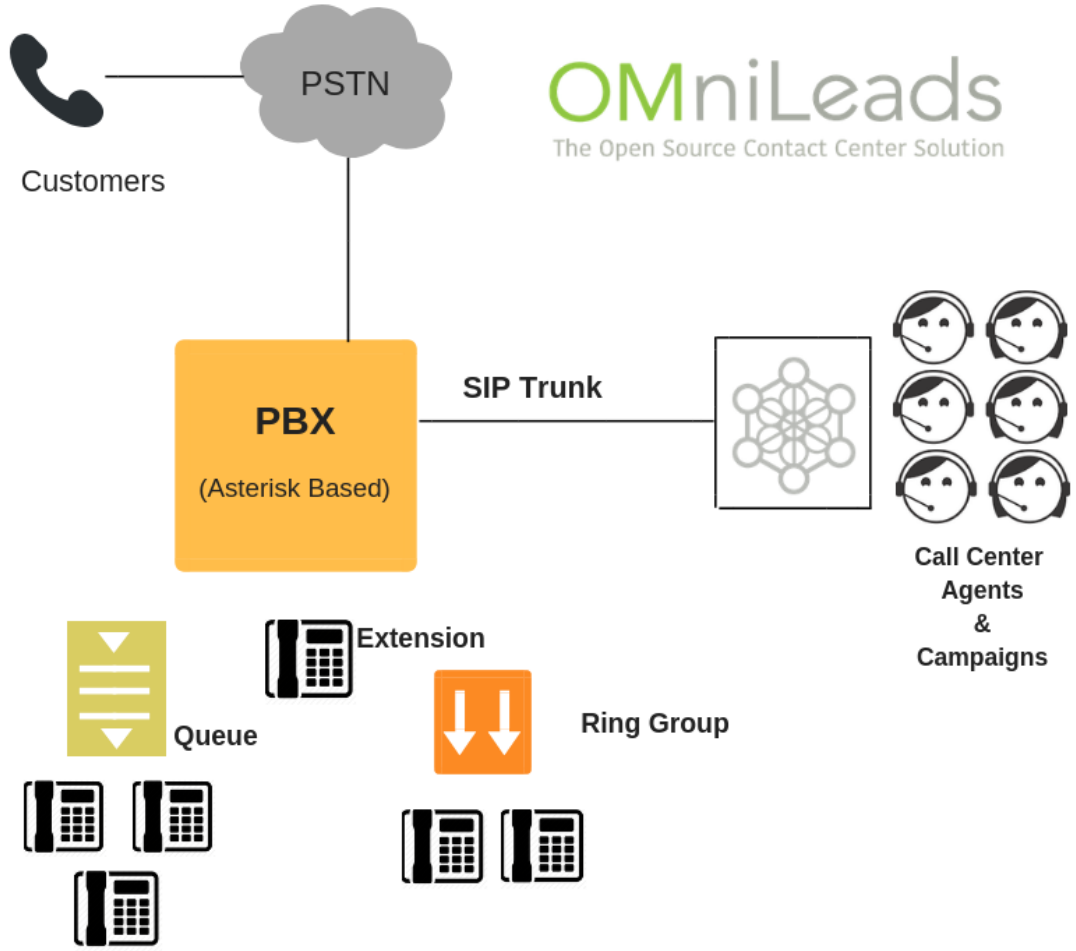
CHAPTER 13

Integration with PBX

Making a few configurations you can stablish a complete integration between OMniLeads and any PBX. In this section you can see examples of the configuration necessary to integrate the PBX and OMniLeads Contact Center that live together in same host.

13.1 Integration between OMniLeads & PBXs

We are going to approach out example using Issabel-PBX, a well known FLOSS project where, besides, every aspect exposed here can be generalized asas a configuration for every IPPBX based on Asterisk.



Note: The steps described on the section are aplicable so the scheme where OMniLeads is in an exclusive host and the IPPBX in other, as where OMniLeads is executedon Docker, living both on the same host in the IPPBX

13.1.1 SIP trunk configuration in the IPPBX

We select the creation of a new SIP trunk y complete the configuration with the following parameters

- In case we have OMniLeads in a Host and the IPPBX in other Host.-

```
type=friend
host=XXX.XXX.XXX.OML
port=5160
disallow=all
allow=alaw
qualify=yes
```

(continues on next page)

(continued from previous page)

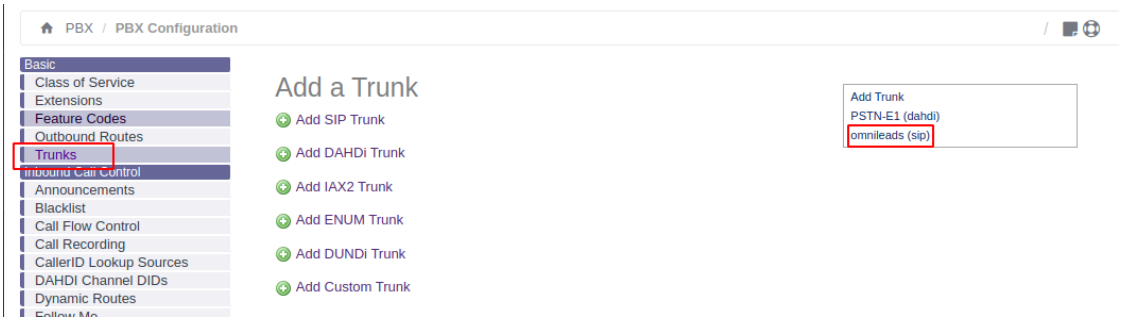
```
secret=omnileads
fromuser=issabel
context=from-internal
```

- In case of executing OMniLeads with Docker inside of IPPBX base operating system

```
type=friend
host=XXX.XXX.XXX.PBX
port=5163
disallow=all
allow=alaw
qualify=yes
secret=issabelOML
fromuser=issabel
context=from-internal
```

Note that the only thing tha changes between both posibilities is the *port* parameter, between 5160 (OMniLeads in a separate host)and 5163 (OmniLeads dockerized over the same IPPBX host)

Once our SIP trunk is available, we go to check accessibility using the IPPBX Asterisk IPPBX



We establish a bash session inside the host running Issabel and raise the command:

```
asterisk -rx 'sip show peers'
```

If all goes well, we should see OK in the output line related to the new SIP trunk, with 5160 or 5163 port

13.1.2 SIP trunk configuration on OMniLeads

Once generated the SIP trunk on IPPBX side, we proced with the generationwith its corresponding part on OMniLeads

- In case we have OMniLeads on a Host and the IPPBX in other host we use the PBX Omnileads (LAN) template.-

```
type=wizard
transport=trunk-transport
accepts_registrations=no
sends_auth=yes
sends_registrations=no
accepts_auth=yes
endpoint/rtp_symmetric=no
endpoint/force_rport=no
endpoint/rewrite_contact=no
endpoint/timers=yes
aor/qualify_frequency=60
endpoint/allow=alaw,ulaw
endpoint/dtmf_mode=rfc4733
endpoint/context=from-pbx
remote_hosts=XXX.XXX.XXX.PBX:5060
inbound_auth/username=issabel
inbound_auth/password=issabelOML
outbound_auth/username=omnileads
outbound_auth/password=issabelOML
endpoint/from_user=omnileads
```

- In case of execution of OMniLeas with Docker inside of the IPPBX base operating system, we used the Omnileads inside PBX template.-

```
type=wizard
transport=trunk-nat-docker-transport
accepts_registrations=no
sends_auth=yes
sends_registrations=no
accepts_auth=yes
endpoint/rtp_symmetric=no
endpoint/force_rport=yes
endpoint/rewrite_contact=yes
endpoint/timers=yes
aor/qualify_frequency=60
endpoint/allow=alaw,ulaw
endpoint/dtmf_mode=rfc4733
endpoint/context=from-pbx
remote_hosts=XXX.XXX.XXX.PBX:5060
inbound_auth/username=issabel
inbound_auth/password=issabelOML
outbound_auth/username=omnileads
outbound_auth/password=issabelOML
endpoint/from_user=omnileads
```

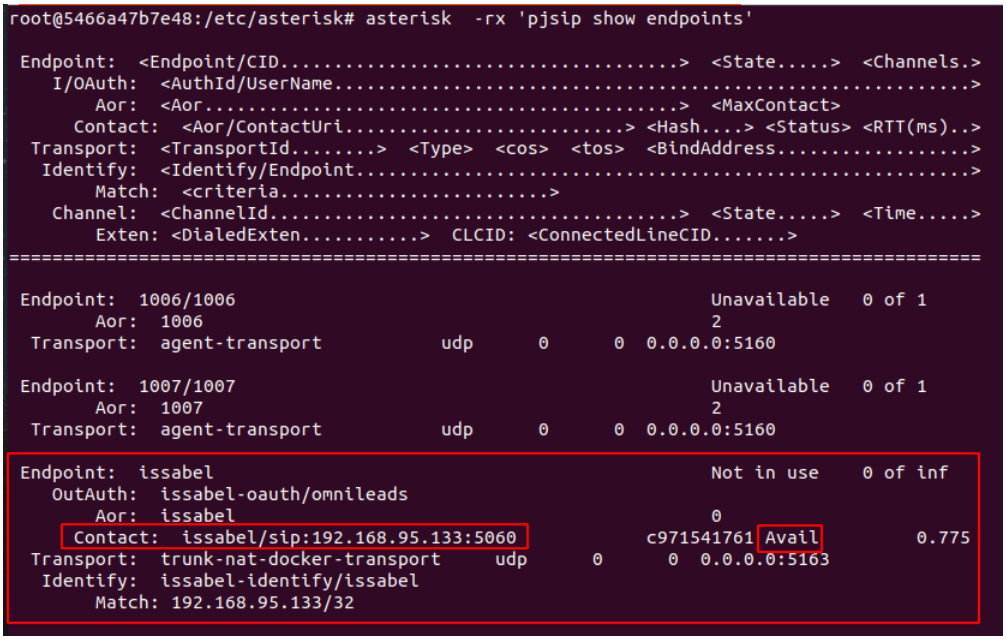
Once efective our trunk, we pass to check if Issable is accessible from OMniLeads, using OMniLeads Asterisk CLI.

Note: If we are executing OMniLeads in Docker, for access the container executing we must execute the

following command: **docker exec -it oml-asterisk-prodenv**, then we can start the CLI

```
asterisk -rx 'pjsip show endpoints'
```

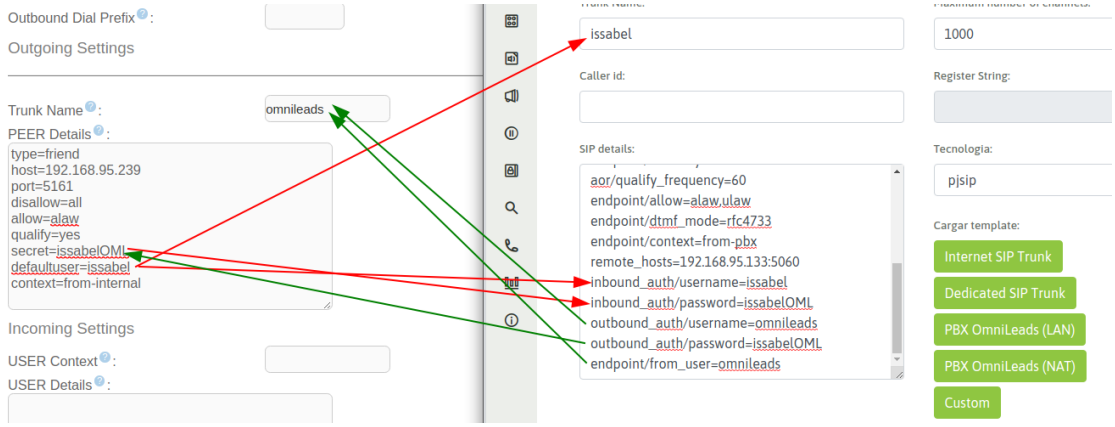
The command output should be similar to the figure:



At this point, exists a SIP trunk between bot phone systems, pending the calls routing configuration between both systems

Finally we emphasize on make relations between parameters onIssabel and OMniLeads SIP trunks

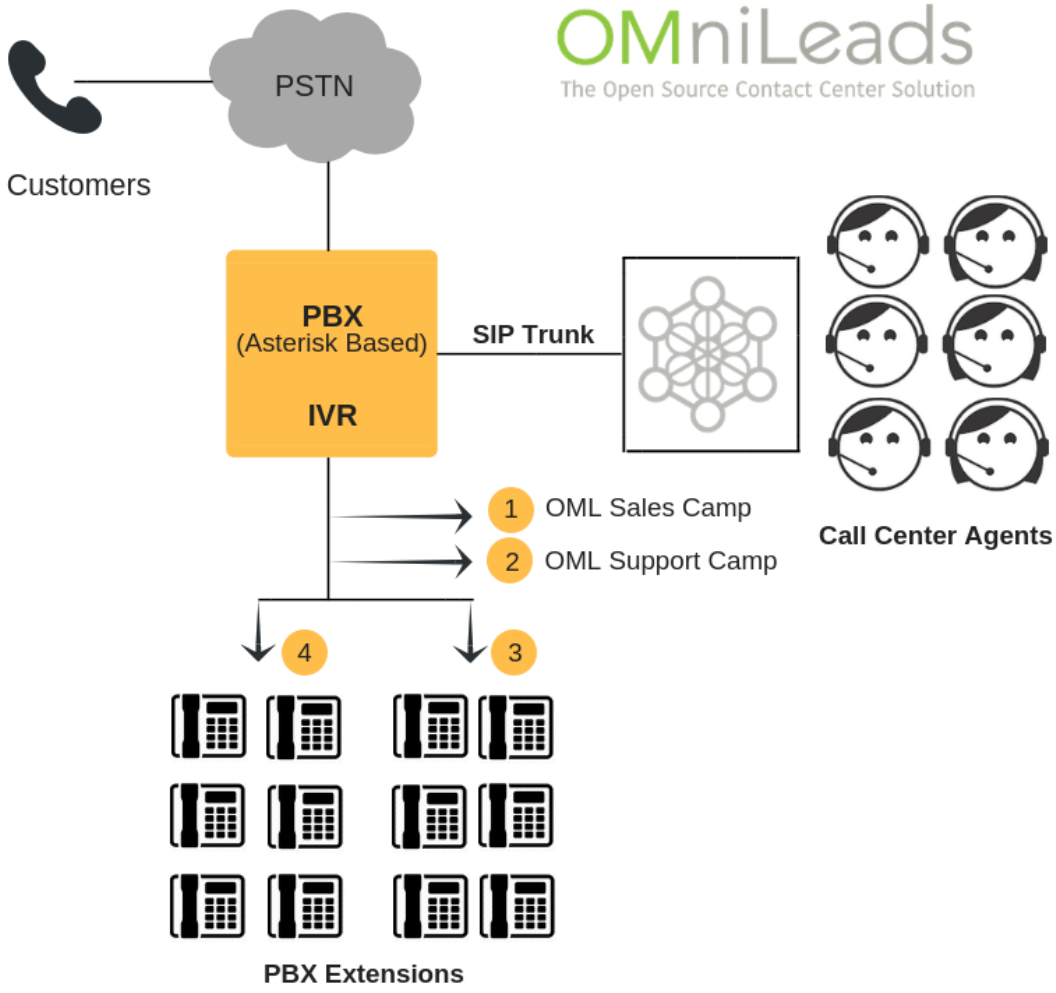
A picture is worth a thousand words:



13.1.3 How send calls from IP-PBX to OMniLeads

Now we exposed a way to connect the IP-PBX resources (inbound routes, IVRs, announcements, extensions, etc.) with OMniLeads. It means, that, for example from one option of the company main IVR can derive to an OMniLeads inbound campaing, or that one extension can contact or transfer a call to an OMniLeads inbound campaign or OMniLeads agent.

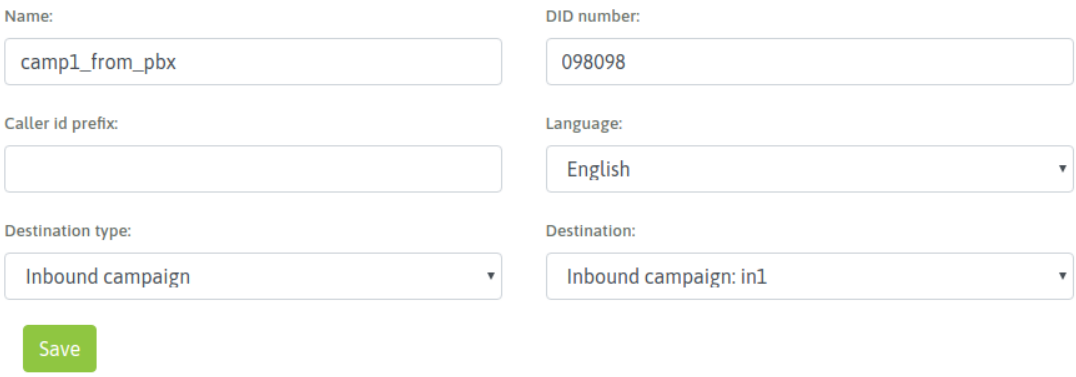
This is completely viable using the IP-PBX *custom extensions*, in ourexample case: Issabel-PBX



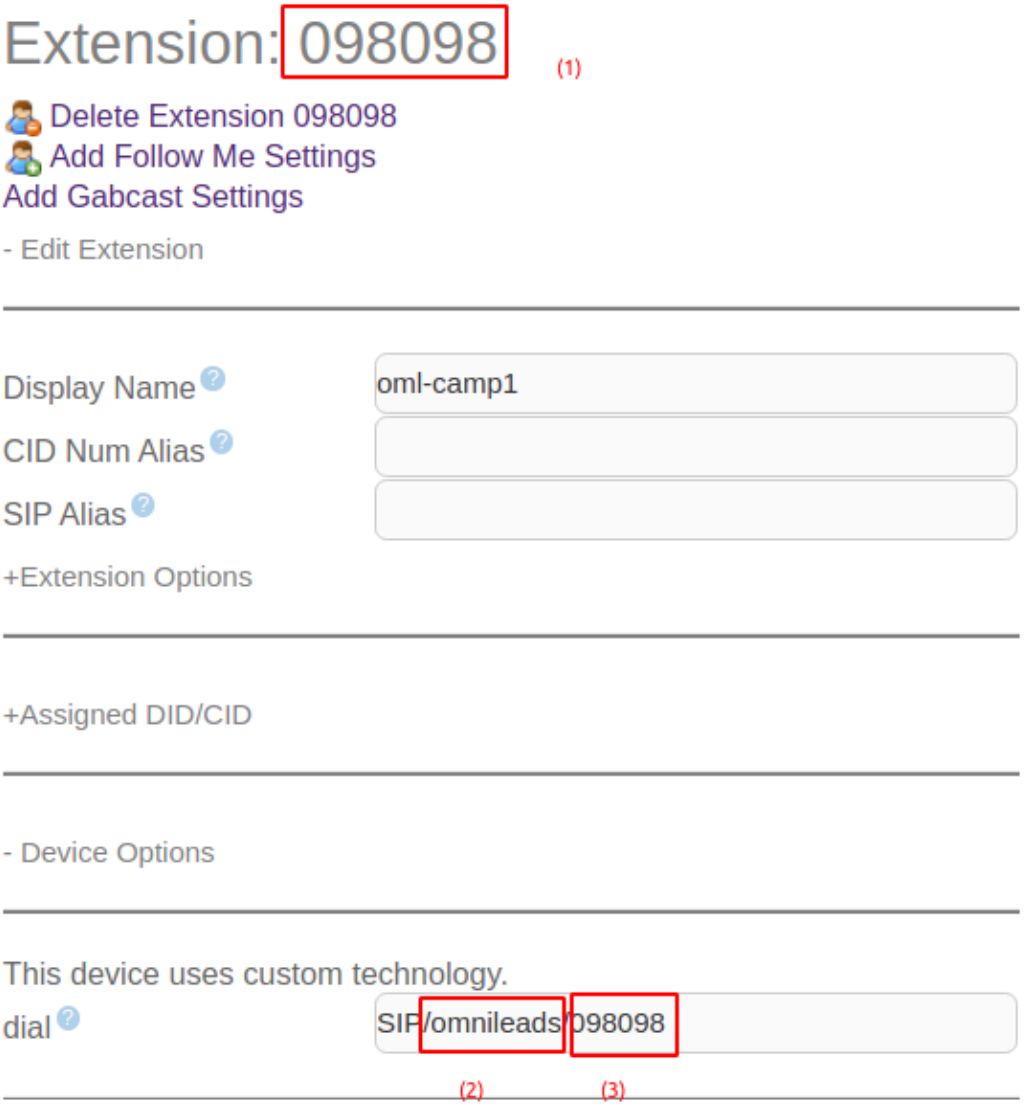
13.1.4 Calls to OMniLeads inbound routes

Now we present the example where the user want to create a *custom extension* in which, when call it from another extension or invoke it from some PBX object(IVR, inbound route, announcement, etc) create a channel against OMniLeads, particularly pointing to an *inbound route* which can at the same time, send the call to an inbound campaign.

For one side, we have an inbound route in OMniLeas, pointing for instance, to an inbound campaign:



Having in mind that the DID choosen was *098098*, in the IPPBX we mustgenerate an *extension* with type *custom*, where the *Dial* stringshould point to the OMniLeads SIP trunk, and the sent number should be *098098*.



In the figure we mark three elements:

(1) • The extension number, could be different to the number sent to OMniLeads(3) . It can be every numner, only making sure the Dial extension from the custom extension matches with the DID number from the OML inbound route (098098 for our example)

(2) • The *trunk* where the custom extension points. This values must matches with the field *Trunk Name* on the SIP trunk against OMniLeads generated on the IP-IPBX

(3) • The number to send by the trunk must match with the DID on the OMniLeads inbound route

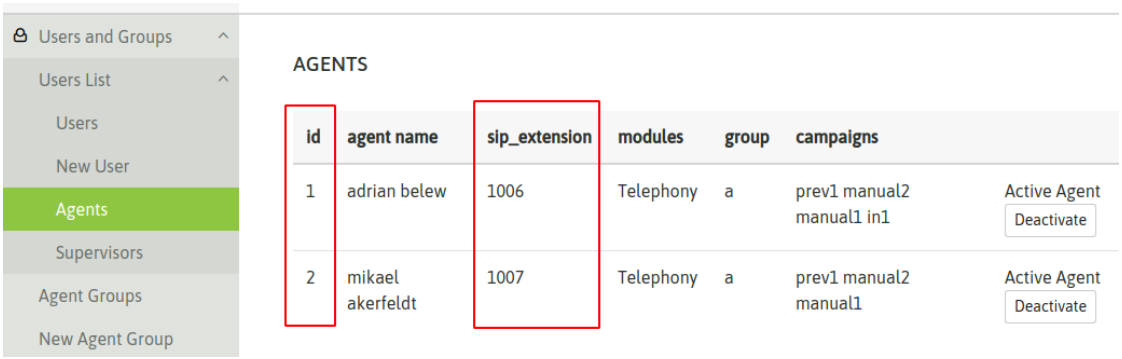
This way every IPPBX extension could call o transfer a call to this *custom extension* and will be sent to the related inbound route on OMniLeads for finally connect over an inbound campaign or the assigned element as destination of the OMniLeads inbound route

As a final mention, we want to make clear that we can have in the IPPBXmany custom extensions pointing to different OMniLeads inbound routesas we like !

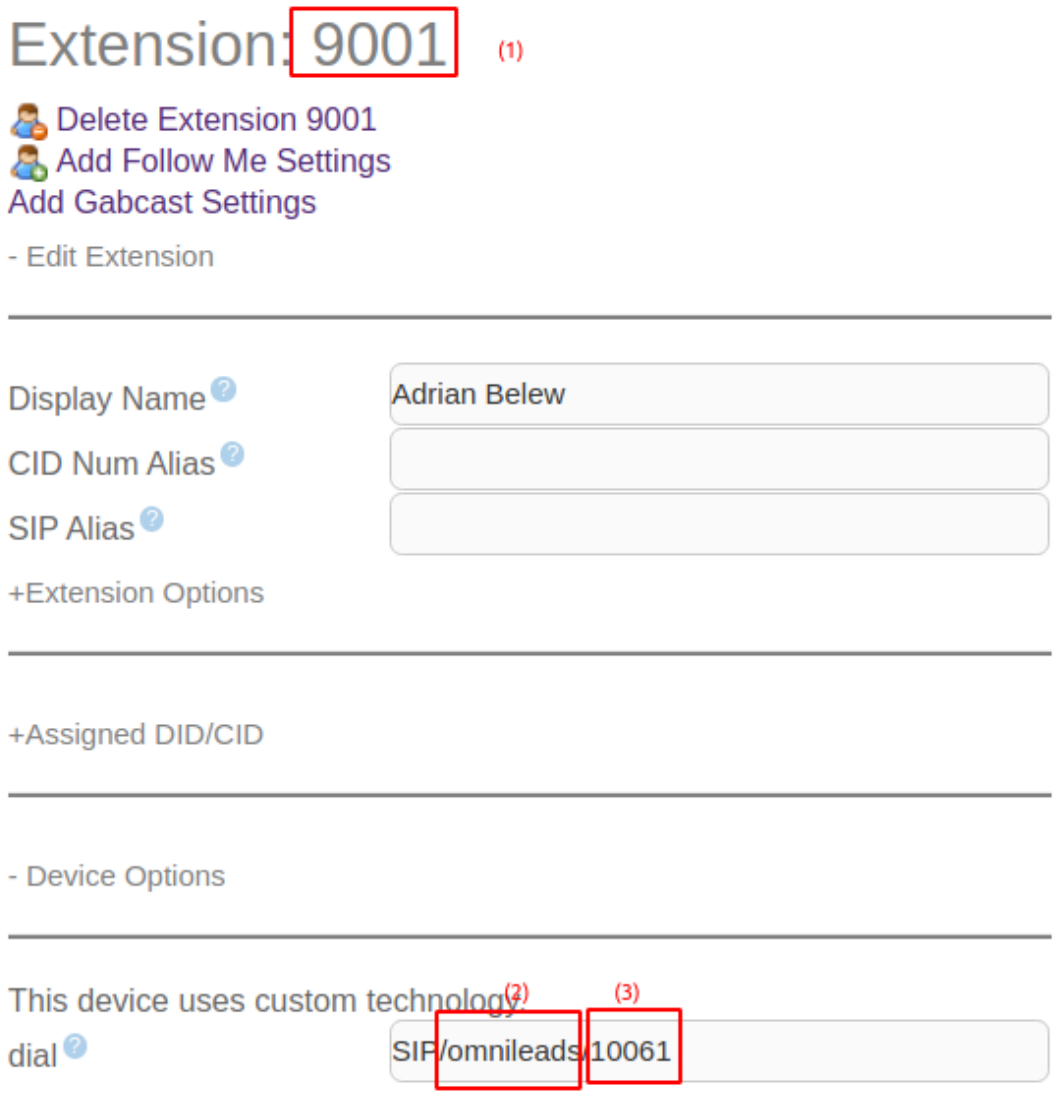
13.1.5 Calls to OMniLeads agents

For the case of link an OMniLeads agent on the IPPBX (it means, that from one extension we can call the *custom extension* and this extension ends linking ina call to an OMniLeads agent), the number to send in Dial string of custom extension will be formed not by a DID of OMniLeads inbound route, but will be a combination ofthe agent ID and its SIP number

Let's get to the point with this:



From the figure, let's take the agent *Adrian Belew*. Note that its ID is 1 and its SIP number is 1006. For that reason at the momment to make the number to send in the Dial string of the IPPBX custom extension we must concatenate the *SIP Number* with its *Agent ID*; in our example will be **10061** for agent *Adrian Belew* and *10072* for agent *Mikael Ackerfeldt*



In the figure we mark three elements:

(1) • The extension number, could be different to the number sent to OMniLeads(3) . It can be every numner, only making sure the Dial extension from the custom extension matches with the concatenation of the ID agent and its SIP number (10061 for our example)

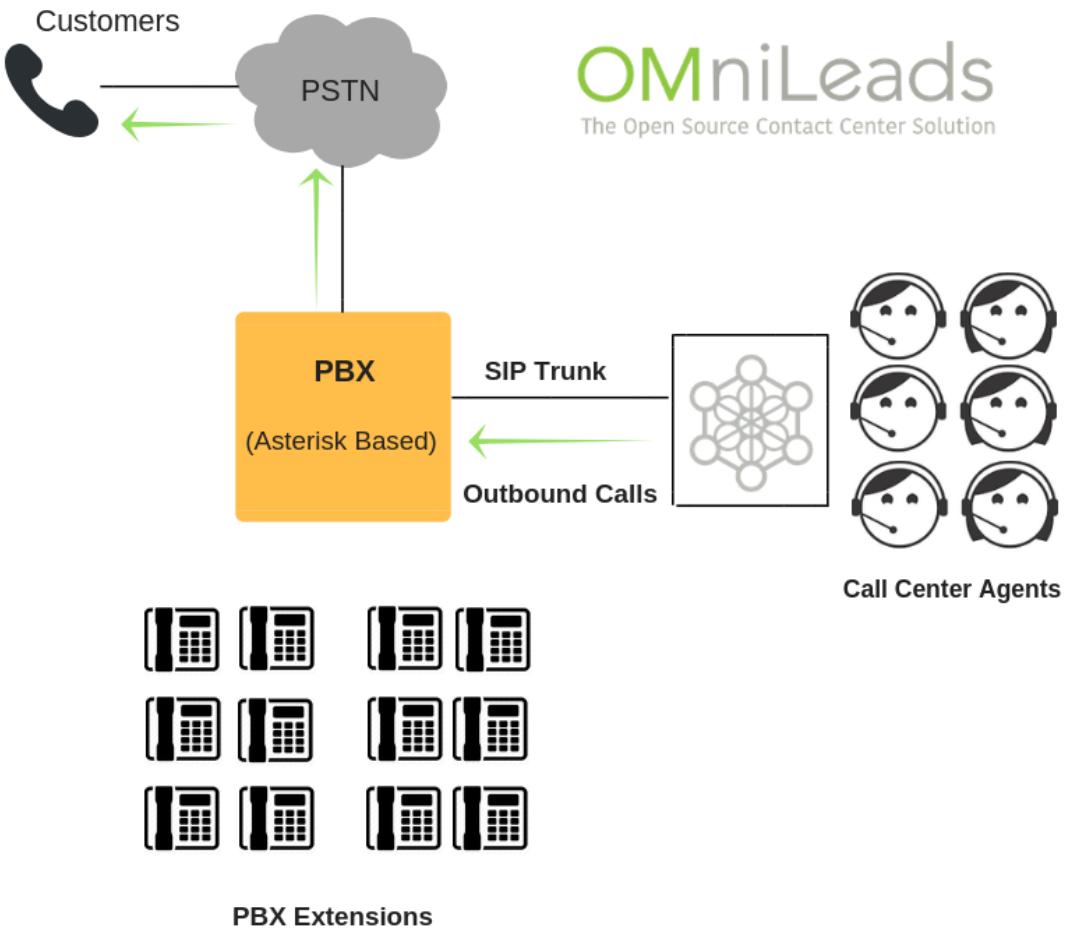
(2) • The *trunk* where the custom extension points. This values must matches with the field *Trunk Name* on the SIP trunk against OMniLeads generated on the IP-IPBX

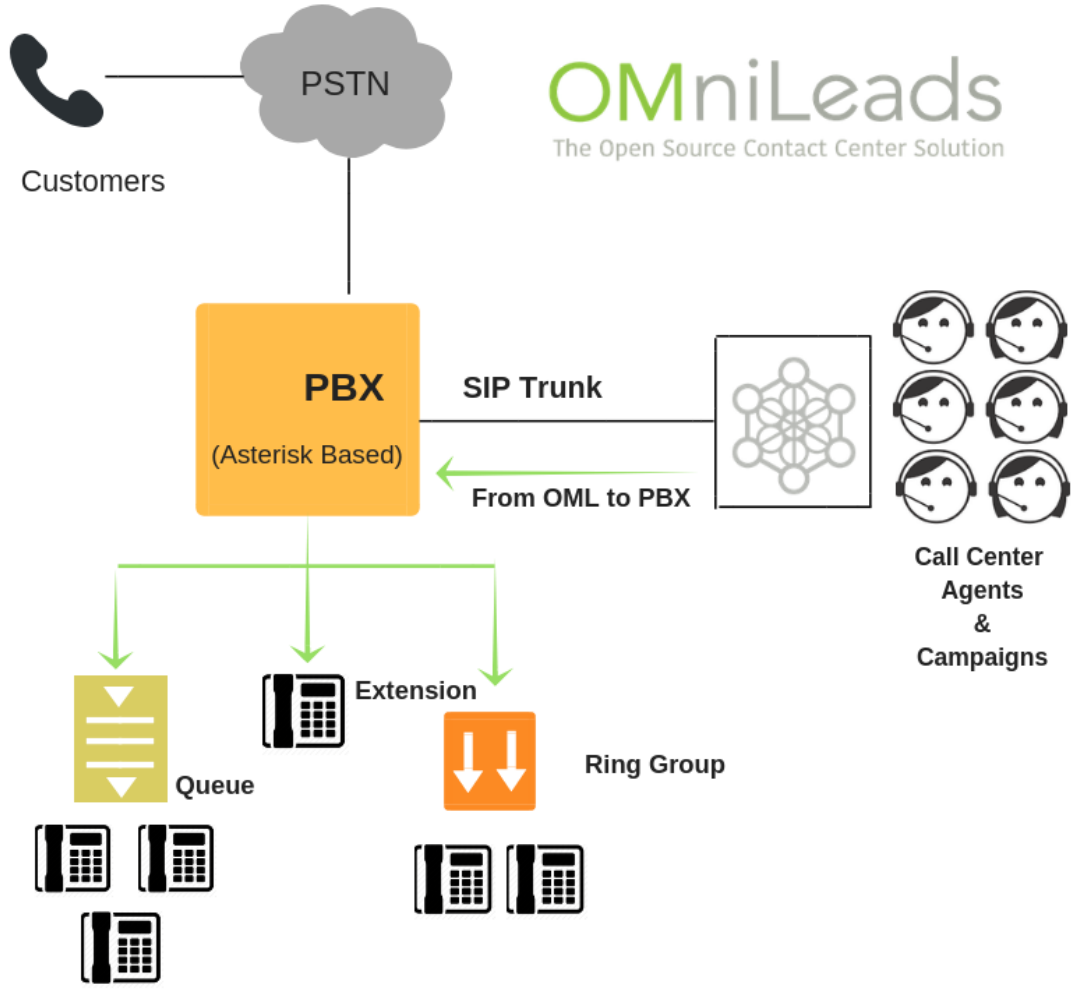
(3) • The number to send by the trunk must to match with the concatenation of the ID agent and its SIP number (10061 for our example)

We must to repeat the procedure for every agent for link inside the IPPBX

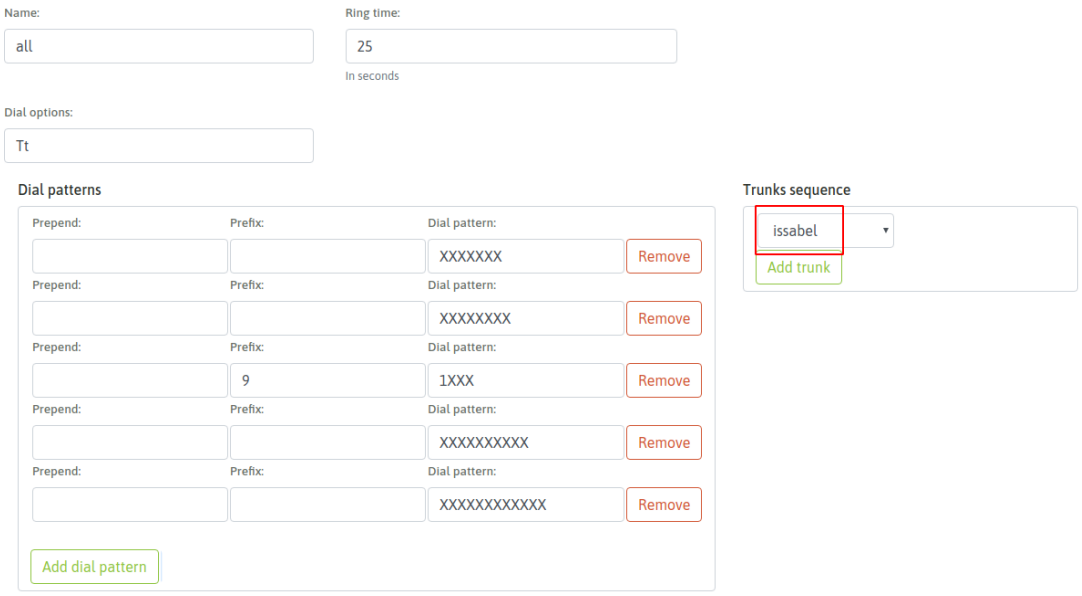
13.1.6 Calls from OMniLeads to the PSTB and the IPPBX resources

Finally we are going to generate the outboung routing inside OMniLeadsthat allow agents and diales raise calls to the PSTN, and , at the same timeallow agents to call or transfer calls to the IPPBX resources like extensions, ring, groups, queues calls, etc


Customers
PSTN
OMniLeads
The Open Source Contact Center Solution
PBX
(Asterisk Based)
SIP Trunk
Outbound Calls
Call Center Agents
PBX Extensions



Simply we must add a new outbound route that points to the trunk to the IPPBX



This way the integration is completely functional and both systems can make all types of calls and interactions

CHAPTER 14

IT ADMINISTRATOR MANAGEMENT

In this chapter is covered some tasks made from the IT administrator of OMniLeads. Refering to: configuration of the dialer plattform, upgrade of software management, backup & restore and network parameters change.

14.1 IT administrator managment

14.1.1 Environment variables

Through this section is going to be commented the procedures that needs the use of *inventory* file. As known the file is edited bofer installation and after that it becomes default. But the variables and their values stayes as environment variables.

To check this variables open the file */etc/profile.d/omnileads_envars.sh*.

```
cat /etc/profile.d/omnileads_envars.sh

AMI_USER=omnileadsami
AMI_PASSWORD=5_MeO_DMT
ASTERISK_IP=192.168.95.163
ASTERISK_HOSTNAME=localhost.localdomain
ASTERISK_LOCATION=/opt/omnileads/asterisk
CALIFICACION_REAGENDA=Agenda
DJANGO_PASS=098098ZZZ
DJANGO_SETTINGS_MODULE=ominicontacto.settings.production
EPHEMERAL_USER_TTL=28800
EXTERNAL_PORT=443
INSTALL_PREFIX=/opt/omnileads/
KAMAILIO_IP=192.168.95.163
KAMAILIO_HOSTNAME=localhost.localdomain
KAMAILIO_LOCATION=/opt/omnileads/kamailio
MONITORFORMAT=mp3
MYSQL_PWD=098098ZZZ
NGINX_HOSTNAME=localhost.localdomain
```

(continues on next page)

(continued from previous page)

```
OMNILEADS_IP=192.168.95.163
OMNILEADS_HOSTNAME=localhost.localdomain
PGHOST=localhost.localdomain
PGDATABASE=omnileads
PGUSER=omnileads
PGPASSWORD=my_very_strong_pass
PYTHONPATH=$INSTALL_PREFIX
REDIS_HOSTNAME=localhost
SESSION_COOKIE_AGE=3600
TZ=America/Argentina/Cordoba
WOMBAT_HOSTNAME=localhost.localdomain
WOMBAT_USER=demoadmin
WOMBAT_PASSWORD=demo

export AMI_USER AMI_PASSWORD ASTERISK_IP ASTERISK_HOSTNAME ASTERISK_LOCATION␣
↪CALIFICACION_REAGENDA DJANGO_SETTINGS_MODULE DJANGO_PASS EPHEMERAL_USER_
↪TTL EXTERNAL_PORT INSTALL_PREFIX KAMAILIO_IP KAMAILIO_HOSTNAME KAMAILIO_
↪LOCATION MONITORFORMAT MYSQL_PWD NGINX_HOSTNAME OMNILEADS_IP OMNILEADS_
↪HOSTNAME PGHOST PGDATABASE PGUSER PGPASSWORD PYTHONPATH REDIS_HOSTNAME␣
↪SESSION_COOKIE_AGE TZ WOMBAT_HOSTNAME WOMBAT_USER WOMBAT_PASSWORD
```

This way the administrator can see them when he/she wants

Important: Do not edit this file.

14.1.2 Configuration of the *Predictive Dialer* module

First of all, we notified if the OML instance deployed in the previous steps does not contemplate the use of campaigns with predictive outbound dialer, this step can be omitted. OMniLeads needs a third-party tool to implement the campaigns with predictive outbound dialer. This tool is based on comercial software licenses that must be managed with the manufacturer.

In any case the system can be used with a test channel that grants as a demo. Therefore we can configure the component and run concept tests before acquiring licenses for a real operation.

If running predictive campaigns is desired, we must generate the following basic Wobal Dialer settings. To generate this configuration we must follow some steps that begin with the access to the corresponding URL.

http://omnileads.yourdomain:8080/wombat or http://XXX.XXX.XXX.OML:8080/wombat

---

Note: In case of running OMniLeads Dockerized the URL is:http://XXX.XXX.XXX.OML:442/wombat

Where XXX.XXX.XXX.OML is the IP of docker engine host

When enter for the first time, we must proceed with the creation of a MariaDB database that uses Wombat Dialer. Click on the highlighted button in figure 2.

Figure 1: DB create

Luego es el momento de ingresar la clave del usuario root de MySQL y hacer click en botón remarcado en la figura 2.

Note: El password del usuario root de MySQL fue configurado en el archivo *inventory* al momento de la instalación y quedó disonible como variable de entorno que puede ser consultada según el procedimiento expuesto al comienzo de esta sección.

We proceed then with the creation of the MariaDB database that will use from now on the Wombat Dialer component.

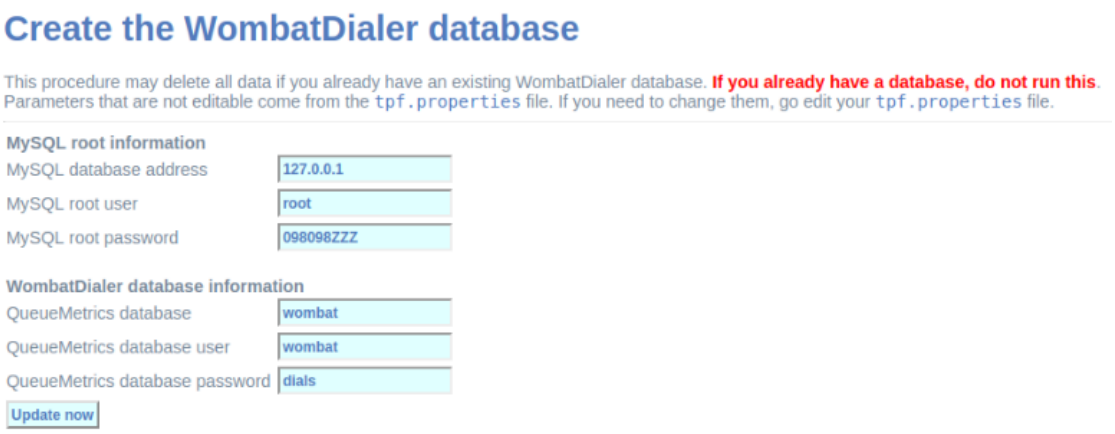


Figure 2: MariaDB root password

Once created the MariaDB database that Wombat Dialer uses, we proceed with the first login.

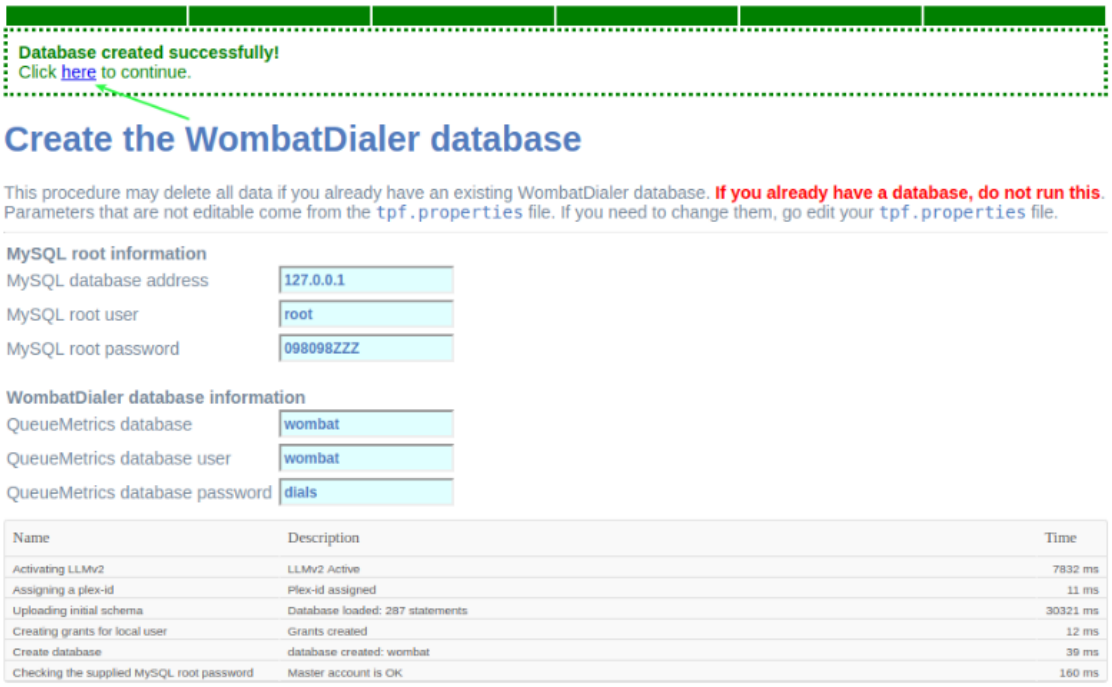


Figure 3: Login post db create

A login must be performed then in the Wombat Dialer administration interface to continue with the configuration of the necessary parameters for the OML interaction.

Upon entering a screen like the following is displayed, where we must access with the user and passwords generated within the installation. Remember that these variables are found in the copy of the inventory file (my_inventory).

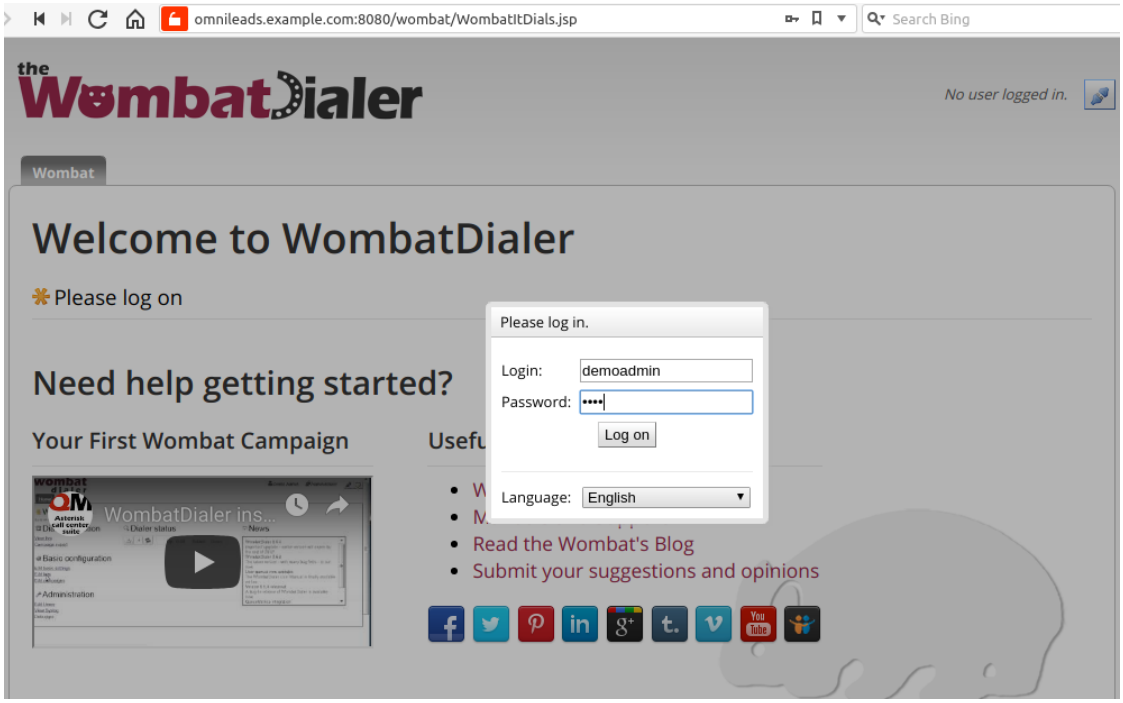


Figure 4: Access to WD

Once inside the system, we proceed with the configuration of the two basic parameters to make the integration with OMniLeads ready. To do this we must access the "Basic configuration" menu as indicated in figure 5.

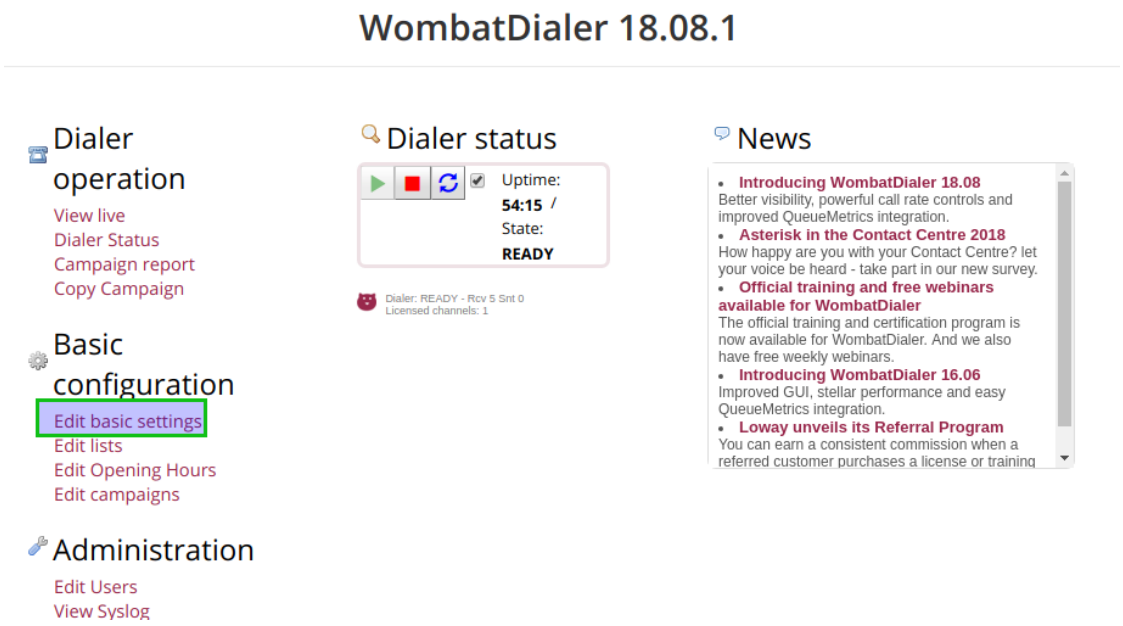


Figure 5: WD basic config

In this menú, first of all, a new conection instance must be generated inside the "Asterisk Servers" section as exposed in figure 6.

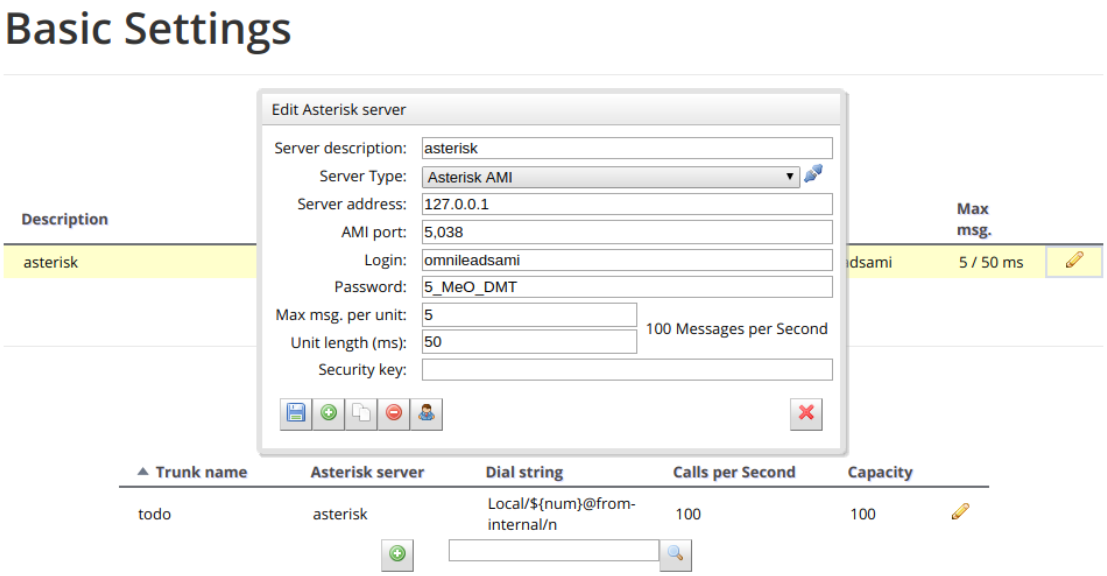


Figure 6: WD basic config - AMI Asterisk

In the next point, a Trunk is configured using an arbitrary "Trunk name", but with the calling chain marked in figure 7. Local/${num}@from-oml/n

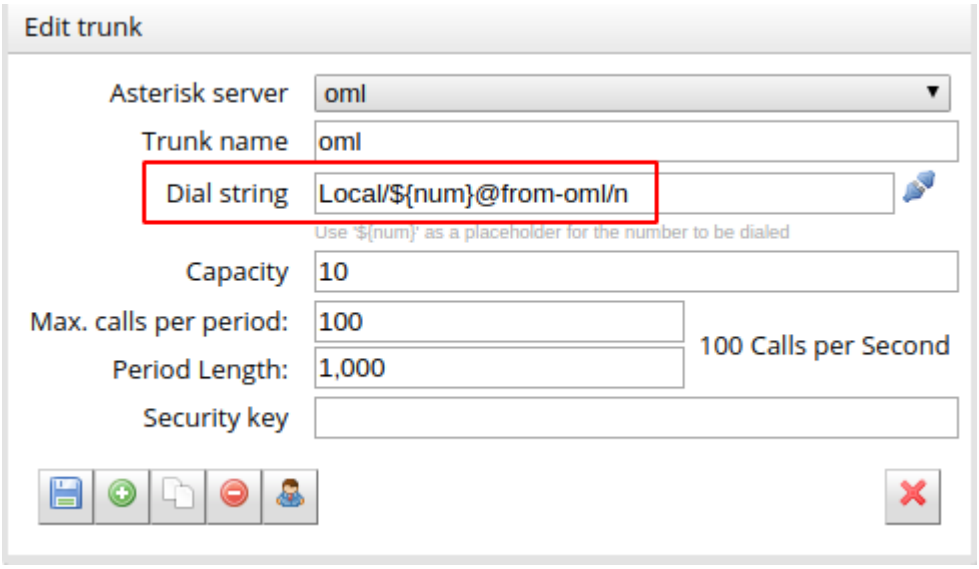


Figure 7: WD basic config - Asterisk Trunk

Por último, recuerde dar "play" al servicio de dialer, tal como lo indica la siguiente figura 8.

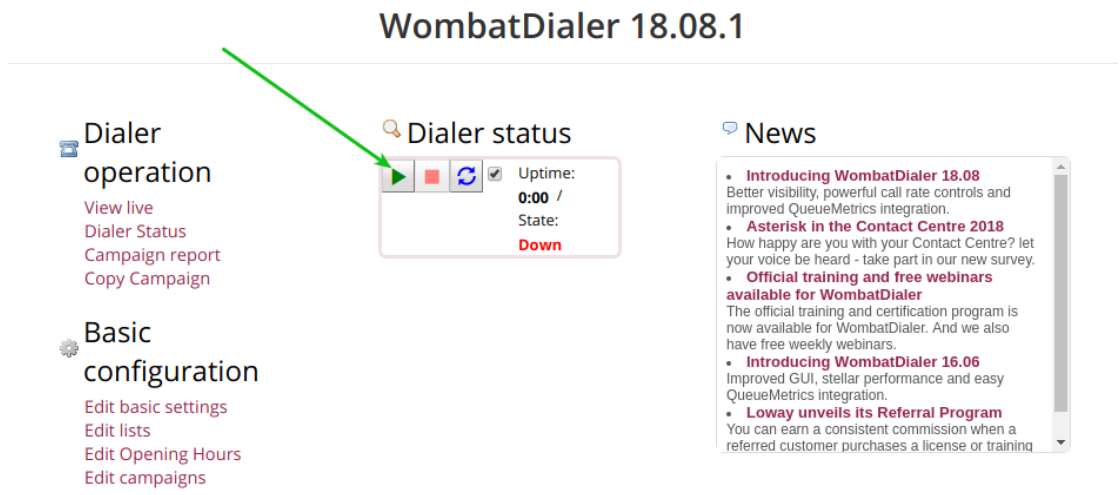


Figure 8: WD activate

Finally the platform is enabled to manage predictive calls. The default installation has a Wombat Dialer demo license for a channel.

14.1.3 Backup & Restore

OMniLeads has a script to perform the Backup/restore tasks.

Important: In case of perform the restore in a new machine, that machine must:

- Has OMniLeads installed in the same version that the productive machine
- Has the same IP and the same hostname of the productive machine

---

To perform a Backup:

We must access the host where OMniLeads is running through ssh. Once inside the host we execute the following commands.

```
su omnileads -
cd /opt/omnileads/bin
./backup-restore.sh -b
```

The execution of the script throws an output similar to the one on figure 9.

```
[omnileads@oml-111 bin]$ ./backup-restore.sh -b
Making backup of asterisk files
Making backup of kamailio files and certificates
Making dump of omnileads database
20190211-omnileads-backup/
20190211-omnileads-backup/asterisk.tgz
20190211-omnileads-backup/kamailio.tgz
20190211-omnileads-backup/postgres_database.tgz

 Backup made in this file: 20190211-omnileads-backup.tgz
Now you can restore doing: ./backup-restore.sh -r 20190211-omnileads-backup.tgz
```

Figure 9: backup

As we can see, it indicates us how to perform the restore of that Backup.

Inside the path **/opt/omnileads/backup**, the ".tgz" files that contain the executed backups are generated.

If the restore is performed in a new host, then the file generated in the Backup within the path **/opt/omnileads/backup** must be left available.

To perform a restore, we must execute:

```
 su omnileads
 cd /opt/omnileads/bin/
./backup-restore.sh -r nombre_del_archivo_de_backup
```

For example:

```
 su omnileads
 cd /opt/omnileads/bin/
./backup-restore.sh -r 20190211_database.tgz
```

It is no needed to add the full location path of the backup.

A succesful restore throws an output similar to figure 10.

```
[omnileads@oml111 bin]$ ./backup-restore.sh -r 20190213-omnileads-backup.tgz
Restoring asterisk files
Restoring kamailio files
Restoring omnileads database
```

Figure 10: restore

Once the restore is finished, execute the following command to regenerate the configuration files and AstDB values of the restored instance:

```
/opt/omnileads/bin/manage.sh regenerar_asterisk
```

14.1.4 Upgrades

Important: Upgrade under release-1.3.1 to release-1.3.1 (including it)

- Is ESSENTIAL to know the passwords for *postgresql*, *mysql* and *django admin* that were set during installation. You can see these passwords in *my_inventory* file and you will have to type them again in *inventory* file. If the same passwords are not used, the upgrade will set up the passwords you typed in inventory file
- If you don't use the same MySQL password you set during install, the upgrade will fail

OMniLeads is forged under a paradigm of continuous releases, which implies a flow of constant updates. That is why is very important to carry out the updates in a clear way.

Below are the steps to follow in order to perform a new platform update. This task is also performed with the script "deploy.sh".

The updates are announced through the oficial communication channels of the project. Depending of the installation method that has been selected.

Installation Self-Hosted

Access the host omnileads as root Position on the directory where the script "deploy.sh" resides

```
cd ominicontacto/deploy/ansible
```

Assuming that we are working on the stable releases (master) A "git pull origin master" must be executed to bring us the repository updates.

```
git pull origin master
```

Uncomment the line for self-hosted installation in the inventory file

```
#############################################################################
↪#############
# If you are installing a prodenv (PE) AIO y bare-metal, change the IP and␣
↪hostname here #
#############################################################################
↪#############
[prodenv-aio]
localhost ansible_connection=local ansible_user=root #(this line is for self-
↪hosted installation)
#10.10.10.100 ansible_ssh_port=22 ansible_user=root #(this line is for node-
↪host installation)
```

Then the script is executed with the -u (update) parameter. This execution will take some minutes and implies applying all the updates downloaded with the "git pull origin master" on our OMniLeads instance.

```
./deploy.sh -u --iface=**your_NIC_name**
```

If everything flows correctly, at the end of the task execution we will see a screen as shown in figure 11.

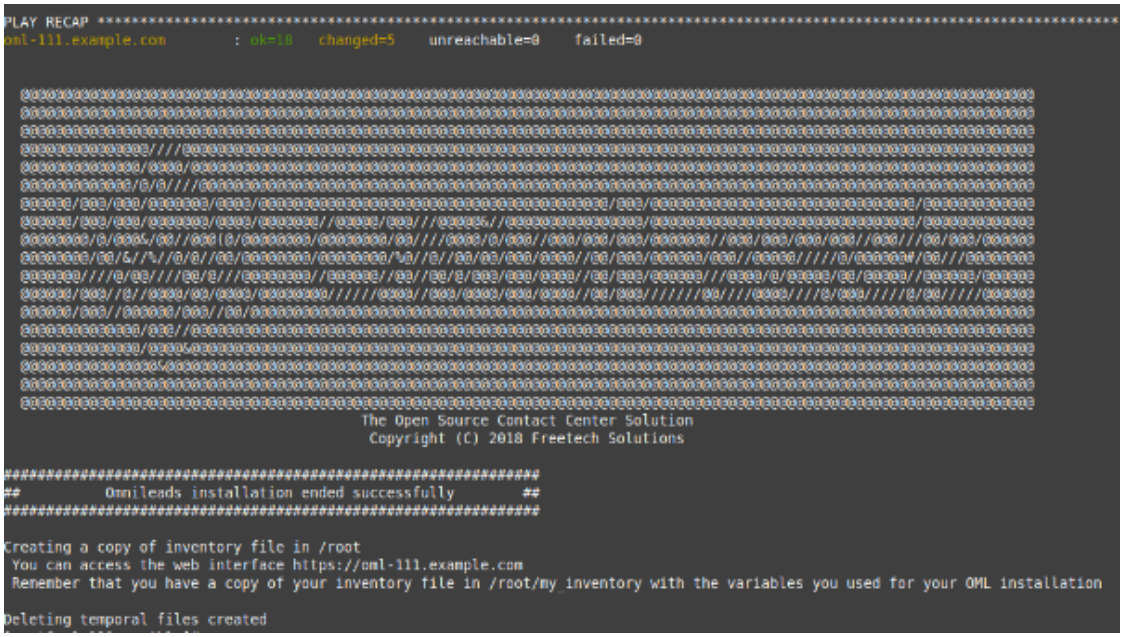


Figure 11: updates OK

Installation from remote Linux Workstation

The cloned repository must be accessed on our Workstation machine, to run the update on the Linux OMniLeads host.

```
cd PATH_repo_OML
git pull origin master
cd ominicontacto/deploy/ansible
```

Next and as in each execution of the script "deploy.sh", the inventory file must be reviewed, ensuring the match of the hostname parameter and IP address with respect to the host where OMniLeads runs and we will update.

```
##############################################################################
↪#############
# If you are installing a prodenv (PE) AIO y bare-metal, change the IP and␣
↪hostname here #
##############################################################################
↪#############
[prodenv-aio]
#localhost ansible_connection=local ansible_user=root #(this line is for␣
↪self-hosted installation)
10.10.10.100 ansible_ssh_port=22 ansible_user=root #(this line is for node-
↪host installation)
```

Note: It should be noted that for remote installation, the line with the parameter "ansible_ssh_port=22" (where 22 is the default port, but is normal for another port to be used) within the section [omnileads-aio]

Then the script is executed with the -u (update) parameter. This execution will take some minutes and implies applying all the updates downloaded with the "git pull origin master" on our OMniLeads instance.

```
./deploy.sh -u
```

Finally, the platform is updated to the last stable version "master"

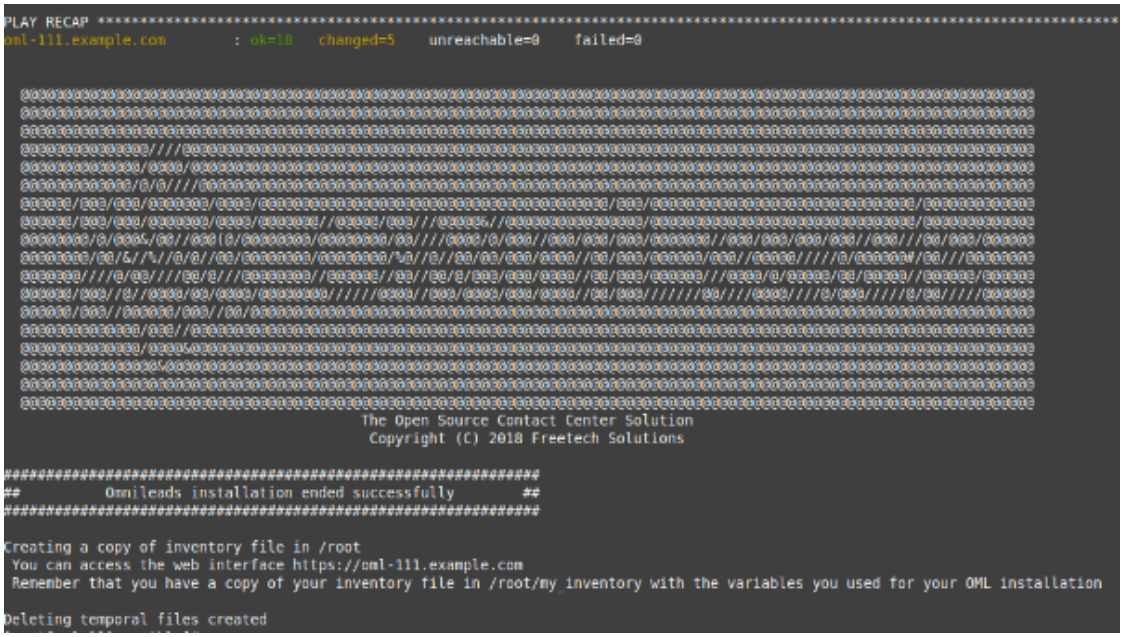


Figure 12: updates from ansible remote OK

Note: Las instalaciones AIO dejarán de ser soportadas en un futuro para Debian y Ubuntu, por lo que se recomienda usar CentOS

Instalation based on Docker containers

One of great advantages of containers technologies is the easy way to distribute, install and upgrade the applications. To OMniLeads follow these steps:

1. Create a copy of .env file in */home/omnileads/prodenv/* to keep your variables
2. Copy the content of *deploy/docker/prodenv/* folder in */home/omnileads/prodenv*
3. Modify the new file .env with your variables
4. Restart omnileads-pbx service

```
systemctl restart omnileads-pbx
```

En el proceso de reinicio cuando se invoca el *docker-compose* al percatarse del *tag* de versión modificado se procede con la descarga de las nuevas imagenes que implementan el release especificado.

Note:

1. The new releases use to bring new JavaScript code. The browser mantains the old code in the cache so it is **recommended** to install an addon in the browser to clear the cache. *Clear cache* to *Google Chrome*, for example
2. Una vez copiado el contenido de prodenv revisar el archivo .env, se pudo haber implementado una nueva variable de entorno

---

14.1.5 Changes of network parameters (Hostname and/or IP Address) and changes of passwords of services

OMniLeads is a complex system, with some services oriented to real-time communications running in the Linux Host. This implies that a change of *IP address* or *hostname* of the host involves certain complexity.

To perform this tasks, we must execute again the script "deploy.sh", same that was used to carry on the platform installation.

We must access the system with the root user, change the IP address in a **operative system** level and/or the hostname and make sure the host took the changes. A system *reboot* is recommended.

Then we continue with the changes on OML, for them we must stop over the directory where the IML repository was clones (if it was a self-hosted installation will be within the remote host, if it was an installtion from remote-ansible will be on the machine *deployer*.), to the access the directory *deply/ansible*, where we find the files *deploy.sh* and *inventory*.

There we must edit the file *inventory* again and review the hostname to make it match with the hostname of the host and there we also must configure the new IP address.

```
[prodenv-aio]
#localhost ansible_connection=local ansible_user=root #(this line is for␣
↪self-hosted installation)
10.10.10.100 ansible_ssh_port=22 ansible_user=root #(this line is for node-
↪host installation)
```

If you are going to use self-hosted type, uncomment the appropiate line in inventory file, if you are going to set host-node type, enter the new IP of the OMniLeads instance.

In the same way, if you want to change passwords of any service (postgresql, mysql, admin pass o AMI pass), uncomment the variables and enter the new password

```
#postgres_password=my_very_strong_pass
#admin_pass=my_very_strong_pass
ami_password=5_MeO_DMT
#mysql_root_password=my_very_strong_pass
```

Changes to the file are saved and finally the *deploy.sh* script is executed.

```
cd ominicontacto/deploy/ansible
./deploy.sh -u --iface=**your_NIC_name**
```

NOTE: if you are solving the hostname of OMniLeads with your file *host* on the work machine, do not forget to also change the parameters.

14.1.6 Users unblock

OMniLeads count with a block users system, when someone enter the wrong password three times. This is the security measure implemented to avoid brute force attacks in the platform's Login console. The administrator user has the possibility of unblocking any user who has been blocked by entering the wrong password unintentionally.

To unblock it you enter the following URL: https://omnileads-hostname/admin, this URL displays the **Django Administrator Console**.

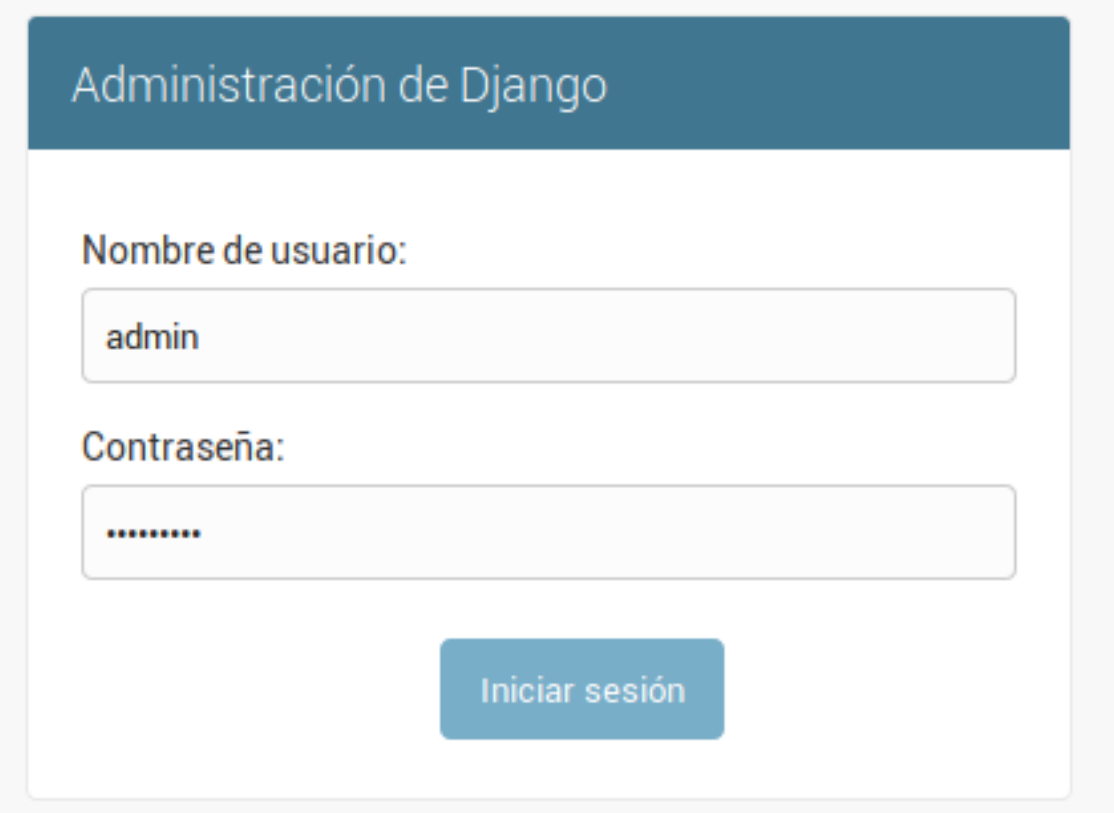


Figure 13: Django admin console

There, enter the admin user credentials. Then click on the button **Defend**

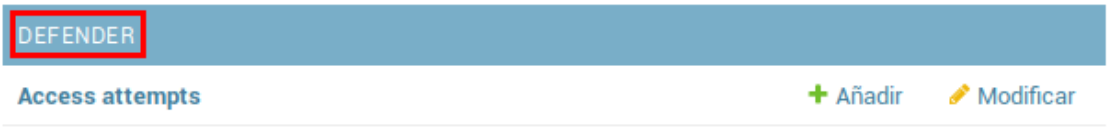


Figure 14: Defender in django admin

This opens the **Django Defender** administrator (https://github.com/kencochrane/django-defend) which is the Django plugin used to manage this. Click on **Blocked Users**

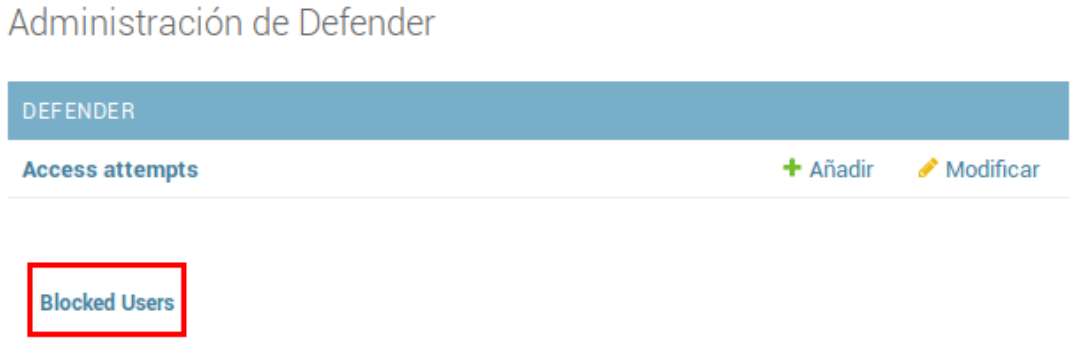


Figure 15: Blocked users view

You will see the blocked user. Just click on **Unblock** to unblock it.

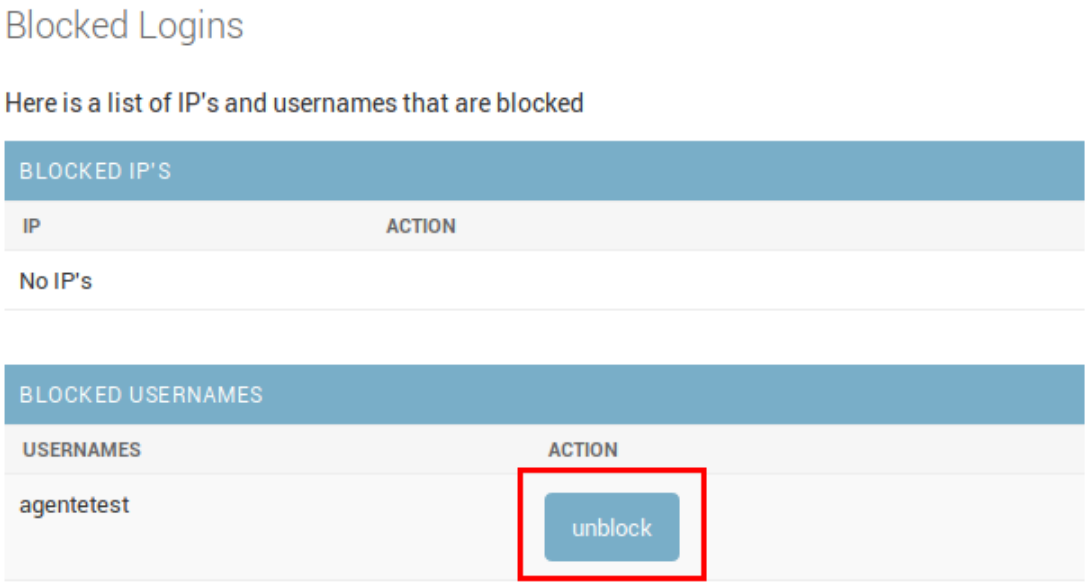


Figure 16: Unblock user view

Now the user can login without problem.

14.1.7 OMniLeads unistall

If by any reason you want to unistall OMniLeads from your machine or VM, there is a script for that. It is already incorportaed in the install process, just execute it:

```
oml-uninstall
```

This script:

- Unistall the essential services of omnileads: asterisk, kamailio, rtpengine, mariadb, postgreSQL, wombat dialer, redis, nginx and omniapp.
- Delete the file /opt/omnileads (including recordings)
- Remove the databases

Note: The script does not unistall the dependency packages used to install the services.

Important: Be careful when executing it, once executed there is no way to recover the system.

CHAPTER 15

CRM INTEGRATION

OMniLeads allows integration with Web CRM systems, allowing to configure the software to send notifications and requests from OMniLeads to the CRM system and vice vers through the system API.

15.1 CRM integration

As well mentioned in the official documentation, OMniLeads allows us a bi-directional interaction with a CRM system. That's why we go to divide the configurations in two parts.

From one hand:

15.1.1 OMniLeads to CRM Interaction

Each campaign can invoke a CRM or particular view

OMniLeads will execute program calls or notifications to the CRM when using campaigns with external URL set from the administration level. This can occur in incoming campaigns, preview or predictive dialer.

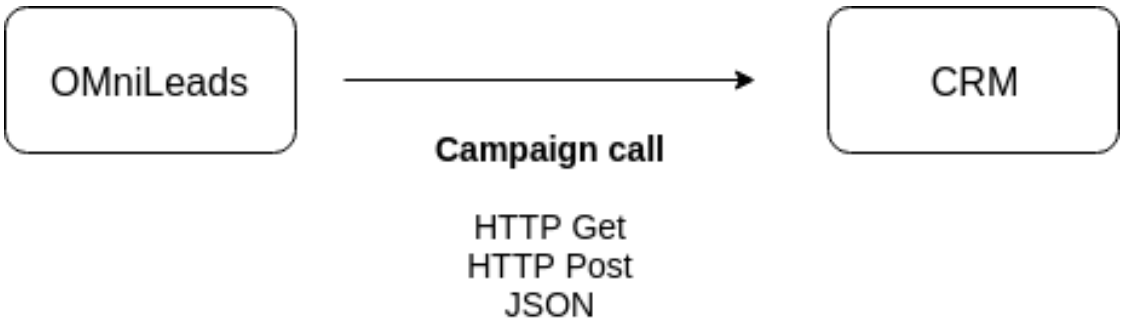


Figure 1: campaign calls and crm

The idea of this interaction is that the agent counts on with a view of the contact information in the CRM. OMniLeads allows a CRM URL invocation involving parameters of the call, contact and / or custom parameters within the context of the campaign running the program call to the CRM.

Depending on the settings applied in the campaign configuration, the invoked URL may be embedded within the agent console , or a new browser tab can be opened for each program call. Another action is to simply send an HTTP-Post JSON to the CRM.

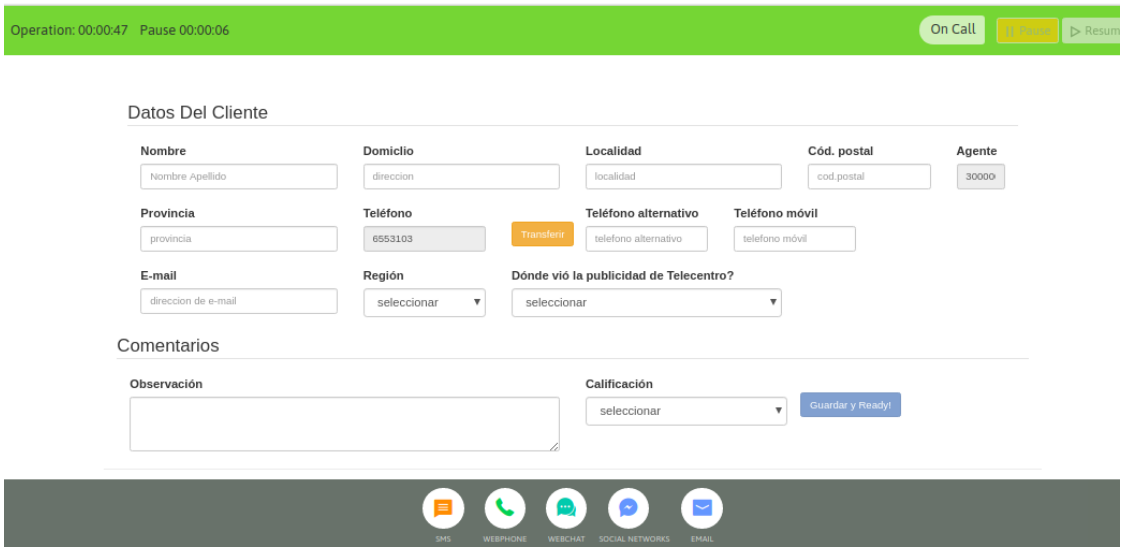


Figure 2: CRM and agent console

Finally we must consider that the execution of the CRM URL by OMniLeads can be done automatically (i.e. when call attempt is performed) or triggered by the agent after pressing a button in the agent console.

All configuration details associated with this scenario of CRM Integration are covered within this section.

Activate a new External CRM entity

The first step to follow is to register the entity ” External CRM” with the related web address (URL) and the interaction settings we expect. For that, access to the menu *Campaigns - External Sites - New Site*.

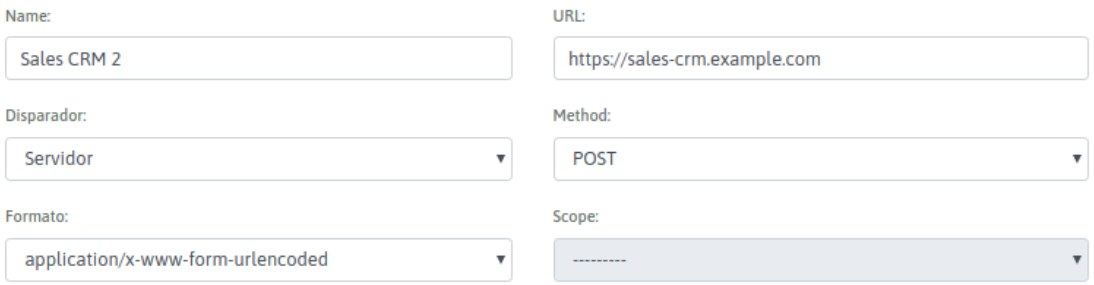


Figure 3: new crm

As stated in *figure 3*, in this step we need to simply complete the resource name, define the URL of the resource to invoke, and the type of interaction (GET, POST or JSON) with external CRM. Finally we set if the system will open a new tab per connected call, make the request and embed the result within the agent console or send a notification (JSON) to the CRM with the call parameters.

Let’s list each form field (*figure 3*):

- **Name**: reference name
- **URL**: this is the web address to invoke on each call. Here we only declare the web resource to invoke. As we will see later, the parameters are customized per campaign.
- **Trigger: here you can select the way you are going to invoke CRM URL web address.**
  - When "Agent" is selected, then when the call connects with an agent, this is the one who triggers the execution of the CRM URL through an AJAX request from the browser.
  - When "Automatic" is selected, the execution of CRM URL is performed at the time the call is delivered to the agent console, through an AJAX request from the browser.
  - When "Server" is selected, then a HTTP POST request to CRM is generated.
- **Method**: the execution of the CRM URL can be done through GET/POST request.
- **Format**: in case of using HTTP-POST, HTML format can be defined here.
- **Scope**: if the "trigger" is set as Automatic or Agent, then the result of the request made to the CRM URL can be displayed "embedded" in the agent console or by opening a new "tab" in the agent browser.

Once the CRM with all the configuration parameters is generated, we can affect it to different campaigns so that the CRM can be invoked in each call delivered to an agent.

Campaign configuration with CRM Interaction

All OMniLeads campaign types are able to activate this CRM interaction per agent call. In his point we will see some examples on how to perform this configuration using the campaign wizard (*figure 3*).

All campaigns have the ability to trigger a form or a CRM Interaction at the time a call is connected to an agent. In this configuration you may indicate CRM Interaction and then define the External CRM entity that will take place.

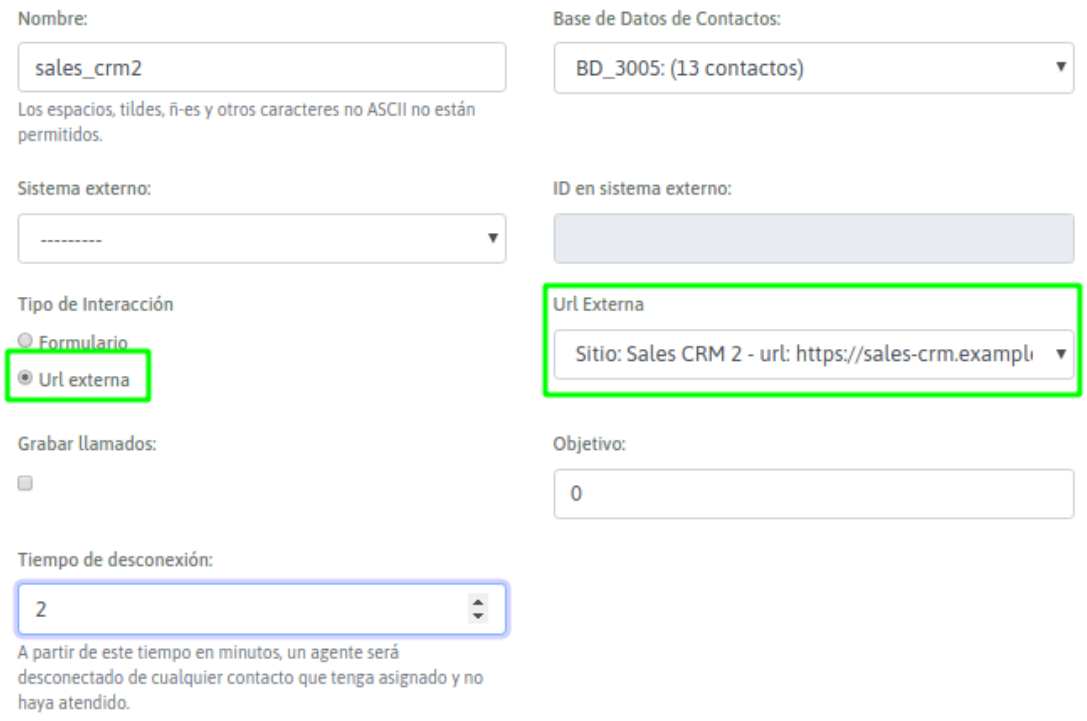


Figure 4: CRM campaign activate

Then it is necessary to assign the parameters to be sent to the CRM. This is also set in the campaign configuration. (*figure 5*).

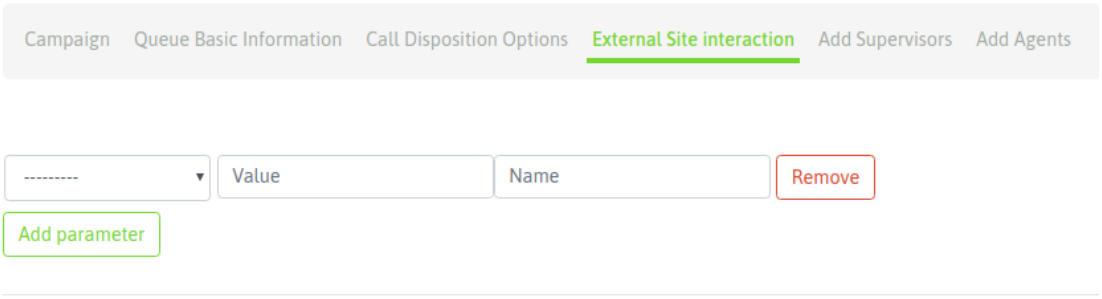


Figure 5: CRM campaign params

At this configuration phase, we can indicate the expected parameters available in OMniLeads System that must be sent to the CRM every time a call is connected to an agent. Those available parameters are grouped by four families:

- **Campaign data, conformed by the parameters:**
  - **id**: id of the campaign.
  - **name**: represents the campaign name.
  - **type**: represents the campaign type.
- **Call data, conformed by the parameters:**
  - **call_id**: is the transaction identifier within OMniLeads.
  - **agent_id**: this is the id of the agent that is processing the call in charged of triggering the request to the CRM.
  - **telephone**: is the contact telephone number.
  - **contact_id**: is the internal id of the contact in the campaign.
  - **rec_filename**: the name of the audio file that contains the recording of the call connected to the agent.
- **Dato de base de contactos**, serian parámetros disponibles a partir de las columnas de la base de contactos de campaña actual. Entonces quiere decir que podemos citar a cualquier columna de la base como parámetro a enviar en una llamada al CRM.
- **Fixed Parameter**, you can set custom parameters to be sent on each call.

With all parameters defined above, in the *figure 5* we need to note that three fields still need to be completed *per parameter* in order to be sent.

- **1st Field**: corresponds to the parameter type (campaign data, call data, database data or fixed data).
- **2nd Field**: corresponds to the specific name of the parameter to be sent (for example “name” if it is a campaign data).
- **3rd Field**: is the name of each parameter, expected from the CRM side.

Example 1: CRM interaction using GET

Let's assume the following URL needs to be executed: **https://my_crm.domain.com?idClient=321321321&idCamp=11&lang=115-20190604-2-4149014-1559667982.424.wav**

As you can note in our example of URL, each execution must provide:

- Contact ID
- Campaign ID that will invoke external CRM
- Parameter "lang=es"
- Recording of the current connected call

How would we implement this requirement from what we have covered in this chapter?

Generate the new CRM

Figure 6 shows the implementation of the proposed CRM seen in our example.

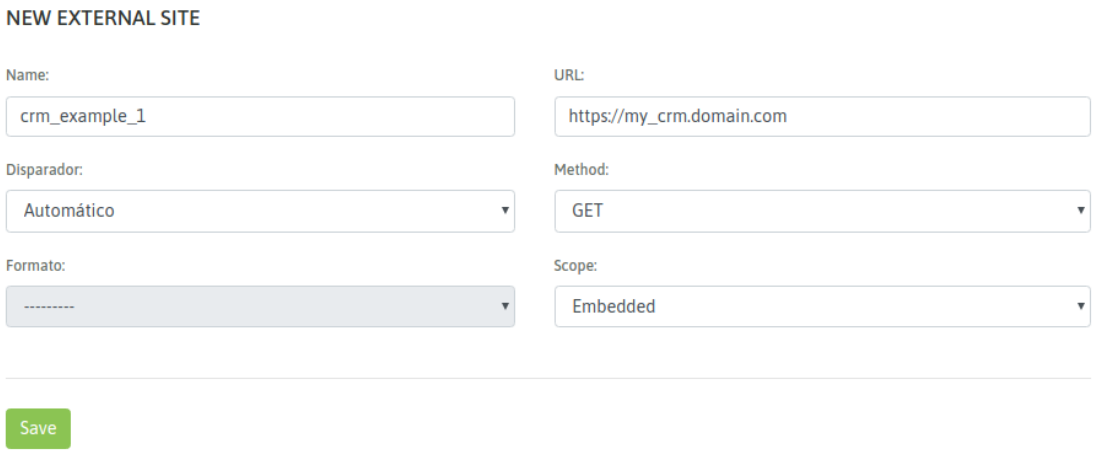


Figure 6: CRM definition

Therefore, campaign configuration now will proceed in order to invoke the CRM with the parameters specified above.

In *Figure 7*, we explain how to configure the campaign to work with the CRM of this example.

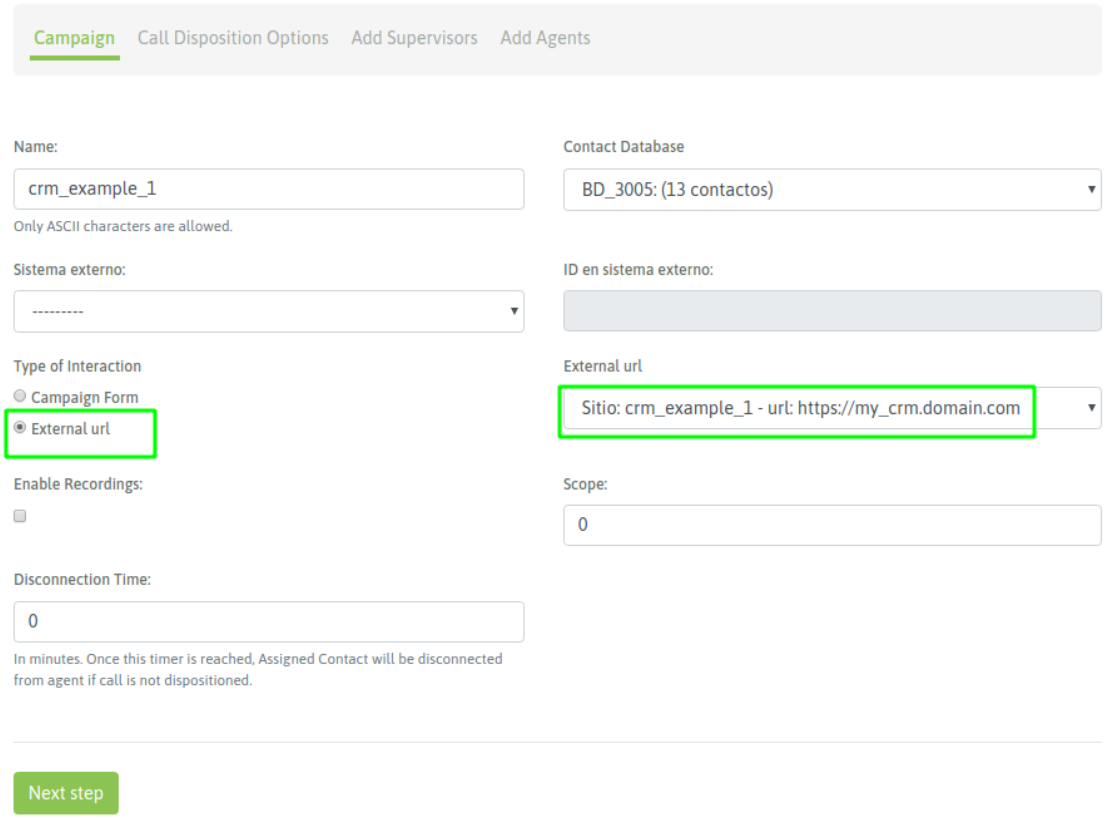


Figure 7: Campaign and CRM

The last step has to do with the assignment of the related parameters for CRM interaction. In *figure 8* we see an example of this step.

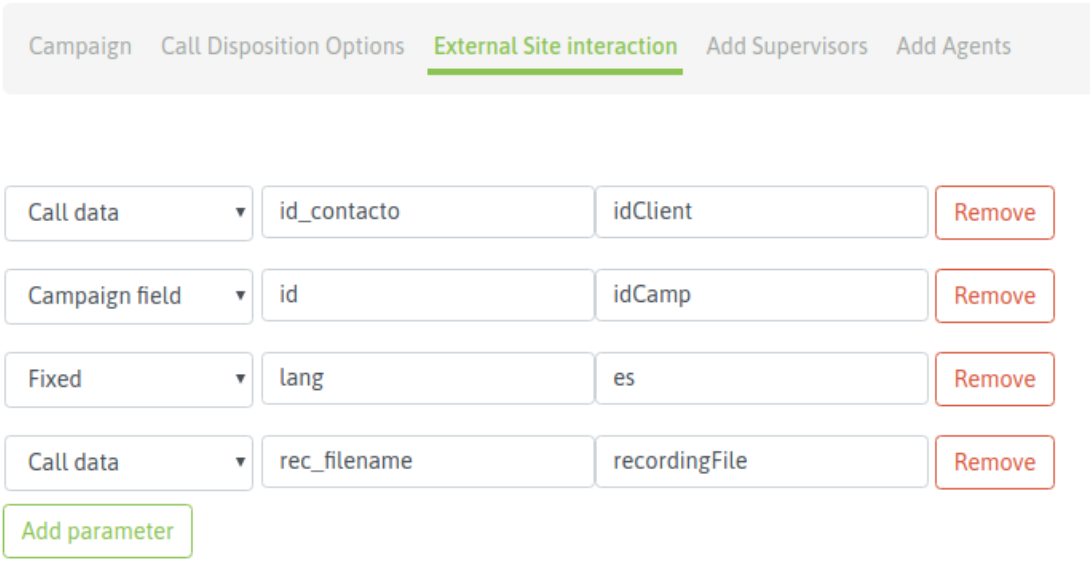


Figure 8: Campaign CRM parameters

Finally we highlight the relationship between columns 2 and 3 for each parameter, since they make the assignment between system parameters and the expected parameter names on the CRM side.

Example 2: CRM Interaction using GET and Clean URLs

Let's assume that you want to run a Clean URL : **https://my_crm.domain.com/idClient/idCamp/lang/recordingFile**

For instance: https://my_crm.domain.com/321321321/11/es/prev-115-20190604-2-4149014-1559667982.424.wav

As you can note in our example of URL, each execution must provide:

- Contact ID
- Campaign ID that will invoke external CRM
- Parameter "lang=es"
- Recording of the current connected call

How would we implement this requirement from what we have covered in this chapter?

Generate the new CRM

The implementation of the proposed CRM is shown as an example in *Figure 9*.

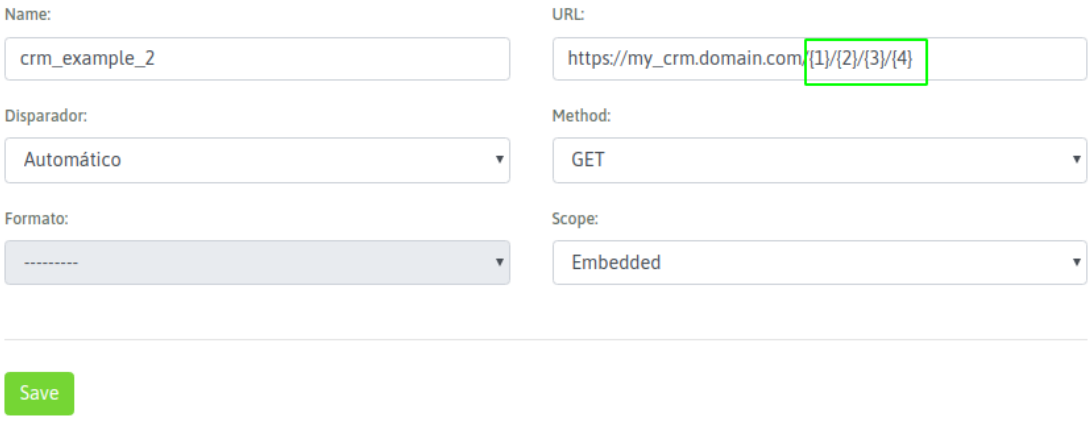


Figure 9: CRM definition with clean URL

The figure highlights the "holders" needed to work with Clean URLs. When generating the URL to be executed, parameters must be specified between brackets. Those Parameters will then be used when generating the campaign for that CRM interaction.

Therefore, campaign configuration now will proceed in order to invoke the CRM with the parameters specified above.

The main difference from the standard URLs (HTTP GET) that is exposed in the *example 1*, is that when assigning parameters in the campaign, "holders" need to be used instead of "Parameter Names", as shown in *figure 10*.

Figure 10: Campaign and CRM parameters

Finally we highlight the relationship between columns 2 and 3 of each parameter, within the scenario of "clean URLs ".

And for the other hand, we get:

15.1.2 CRM to OMniLeads Interaction

It is also possible from a CRM system to execute requests once accessing to the endpoints of the *API de sesión de Agente en Asterisk*. Throughout this section we will go to explain how to activate the actions available from an external CRM.

- **Click to call**: a CRM user who is available in the contact view can make a call to the contact number clicking on that number within the CRM. The click action triggers a program call to an OMniLeads API method in order for the system to route the call accordingly.
- **Call Disposition**: each call connected to an agent can execute a request to the CRM passing usable call parameters. When the user finishes the call within the CRM, she/he proceeds with the "Call Disposition". This action allows CRM to access an OMniLeads API method establishing a correlation between the OMniLeads campaign level and CRM contacts.

In order to implement the listed actions, CRM developer needs to implement these functionalities consuming the *API de sesión de Agente en Asterisk*. Once the functionalities from CRM side are available , the following configurations need to be set as to start operating with this integration level.

CRM and OMniLeads entities Relationships

As we well know, each call processed by OMniLeads implies a relationship between a Campaign, an Agent and in most cases a Contact. We could talk about the Trinity: "Agent - Contact - Campaign".

In the CRM universe we also know that we have the same relationship between the CRM User, the campaign she/he is operating and the contacts associated to the campaign. Therefore, the relationship stated here has to do with the fact of being able to relate each OMniLeads agent, campaign and contact with the expected CRM entity side. This concept will allow a perfect correlation between CRM and OMniLeads systems when executing a "click to call" or "disposition a contact".

OMniLeads has its own identifiers (agent, campaign and contacts of the campaign) self-generated. It is also common to find the same scenario in CRM , therefore it is desired to make a mapping process between OMniLeads and CRM in this aspect. We will explain below how to synchronize these identifiers between both systems.

OMniLeads Agents and CRM Users relationship

The first step is to register an External System entity and associate the OMniLeads agents with CRM users by entering the ID (identifier) of the user in the CRM, as indicated in Figure 1.

Figure 1: new external crm

That being said, each identified agent will be able to interact from the CRM.

Relationship between OMniLeads and CRM Contact Database

In order to create a Contact Database relationship between systems, the database upload process will offer an external ID selector field, as shown in figure 2.

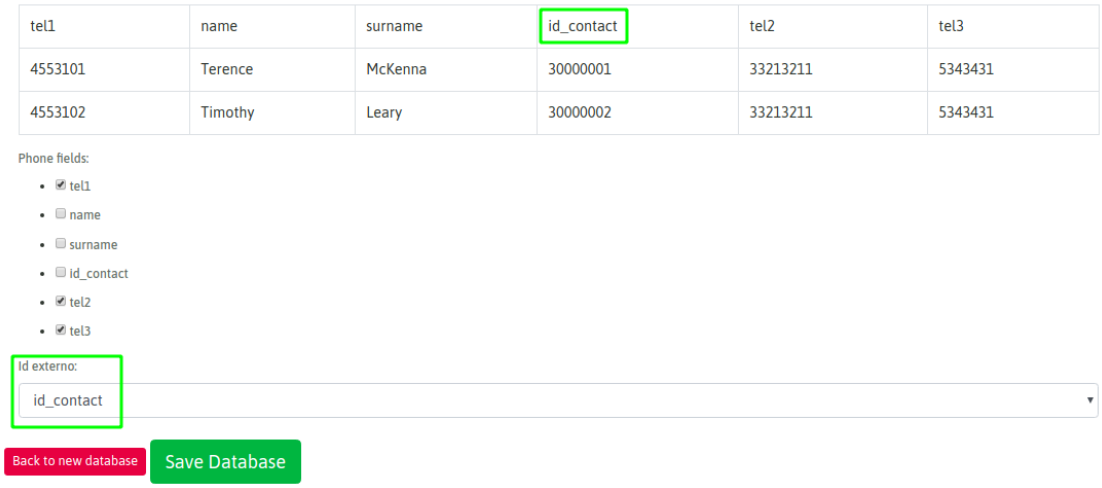


Figure 2: new crm contact database

This value must be unique for each contact in the database, so there should be no contacts with the same value. Each contact can only have one external identifier.

In Figure 2, the example assumes that column “id_contact” is the one reserved for external ID. Therefore, OMniLeads will consider this column/value when interacting with CRM system.

Relationship between OMniLeads and CRM Campaigns

When a Campaign Wiozard is in process, there are fields related to CRM interaction. Remember that in the section of OML to CRM Interaction we present the field Type of interaction “External URL”, to be able to launch a program call to the CRM for each call connected to the agent. In this section we present the Interactions from the CRM to OML, so let’s work with the fields “External system” and “external system ID” respectively (Figure 3).

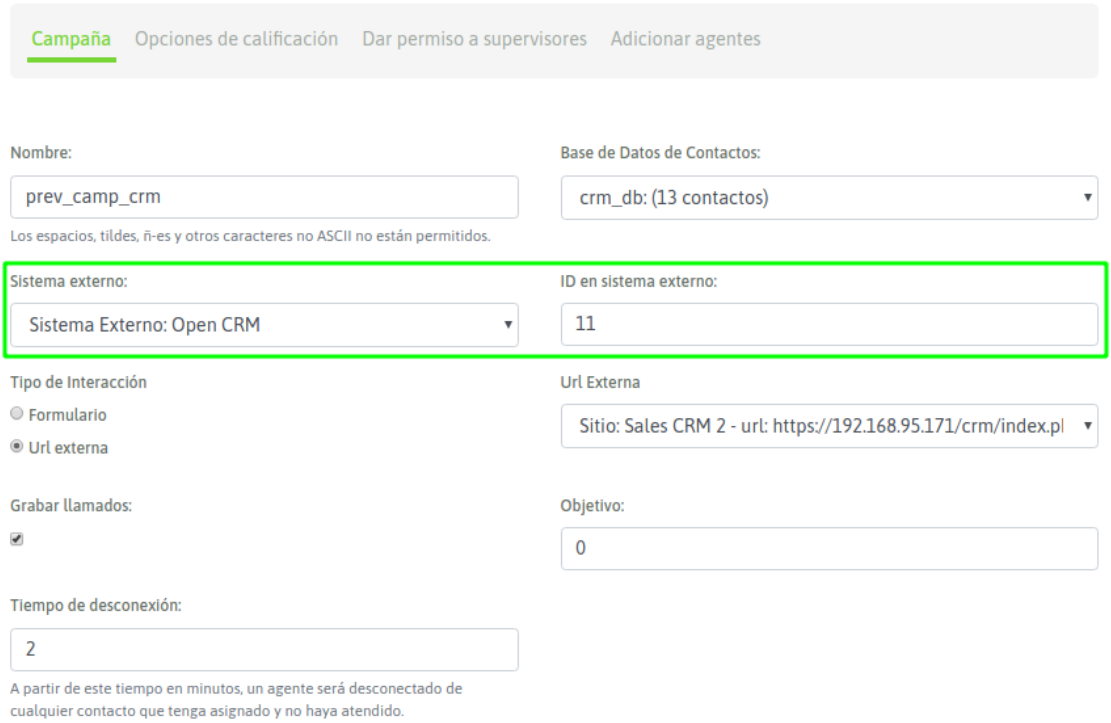


Figure 3: new camp crm to oml interaction

Therefore, the wizard process selects the external CRM that will execute the requests against OML (External System). And on the other hand, the ID of the corresponding campaign we wish to link to (external system ID). Each campaign can only be linked to one External CRM campaign.

Then we move forward with the creation of the campaign with all the steps that we already know.

Note: When the exposed linkage is processed, the following exceptions can exist.

- When assigning agents that do not have an external identifier in the selected External System.
- When assigning a Contact Database that is already assigned to a campaign with another External System configured.
- When assigning an External URL that is already assigned to a campaign with another External System.

  Notifications will also appear when editing an External System , if there are missing external identifiers in agents assigned to campaigns where External System is configured.

CHAPTER 16

OML API

In this section you will find all the information about the Rest API of the system

16.1 OMniLeads RESTful API

This section is destinated to developers that want to execute an integration between its CRM system and OMniLeads. For that reason, the terminology and information provided here has software developers as its public target.

OMnileads offers a RESTful API base on HTTPS / JSON. This API allows access to system resources and services by outside of the user web interface, allowing this way that external systems could communicate in a simple way with OMniLeads.

The authentication methods available for this API are: Session (the agent must be logged on the system via web interface) and Token (using its credentials to obtaing a token)" and then passing it on the headers of the request:

""Authorization: Bearer <token value>""

For example:

```
GET https://<omnileads-addr>/api/v1/disposition/
Authorization: Bearer 22bab1b107251a6771bdf3dbeb5184a7f4b5c182
Content-Type: application/json
```

Note: In new releases this section will increase, in order to add new endpoints.

Then when describe the availables endpoints

16.1.1 Login Endpoint

This endpoint allows to authenticate as a system user, in case of success, it allows access to others availables endpoints depending of the user profile

URL: POST https://<omnileads_addr>/api/v1/login

```
POST https://<omnileads-addr>/api/v1/login
Content-Type: application/json
{
        "username": "agente_username",
        "password": "agente_password"
}
```

figure 1: endpoint login request

filed name	type	description
username	string	username value generated from the OML users creation menu
password	string	password value generated from the OML users creation menu

Successful authentication

If the login is successful, the endpoint shows the following output:

```
HTTP 200 OK
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "token": "6c8a7b970996480adcbaf17acff053d145b039ff",
    "expires_in": "359.997618",
    "user": {
        "first_name": "Alessio",
        "last_name": "Tachinardo",
        "email": "a@a.it"
    }
}
```

figure 2: endpoint login request ok

As shown in the picture, a successful login, returnsfields like security “token”. This token must be used on next requests to the API from the authenticated user. Also, in the field “expires_in” indicates the token lifetime

In case that the system makes a request and the security token has expired, then a new authentication request must be done.

Note: The security token's lifetime can be configured modifying the "TOKEN_EXPIRED_AFTER_SECONDS" parameter located on "/opt/omnileads/ominicontacto/ominicontacto/settings/production.py"

Authentication failed

If the login failed, the endpoint returns the following output:

```
HTTP 404 Not Found
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "detail": "Invalid Credentials or activate account"
}
```

figure 3: endpoint login request fail

16.1.2 Endpoint to obtain Contact database structure

This endpoint makes possible to obtain information fields informationof contacts database related to a campaign. With this information is possiblethen to create a new contact. The credentials must belong to an Agent(*Create agents*) or a Supervisor (*Create supervisor user*)associated to a campaign

URL: POST https://<omnileads_addr>/api/v1/campaign/database_metadata/

filed name	type	description
idExternalSystem	integer	Optional parameter, if sent the system tries to locate the contact as 'external_id' on the Campaign contacts database. If not sent, the system will assume that the value of the paramater 'idContact' is the intern id on OML
idCampaign	string	Contact identifier to tag, its value depends on if 'idExternalSystem' parameter is sent

In case of no errors ocurred it will show an output like this, with data of the new disposition created

```
{u'external_id': u'idext',
 u'fields': [u'nombre',
             u'idext',
             u'tel',
             u'tel2'],
 u'main_phone': u'tel',
 u'status': u'OK'}
```

The 'fields' field indicate the list of all the fields of a contact database.The field 'main_phone' indicates which is the field correspondent to the main numberThe field 'external_id' indicates which field corre-

spond to the external id of contact.When the database doesn't have external id, the 'external_id' field will be None.

In case of errors ocurred, the endpoint returns a JSON with the field 'status':'ERROR' and the detailed information of the error on the field'errors'. On other case the 'status' field value will be 'OK'

16.1.3 Endpoit to create contact

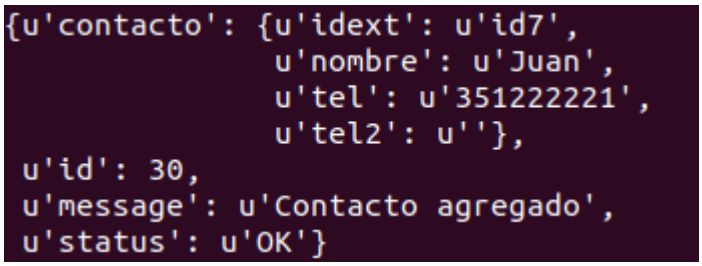
This endpoint allows to add a contact in a database referred to a campaign.The credentials must belong to an Agent(*Create agents*) or a Supervisor (*Create supervisor user*)associated to a campaign

URL: POST https://<omnileads_addr>/api/v1/new_contact/

filed name	type	description
idExternalSystem	integer	Optional parameter, if sent the system tries to locate the contact as 'external_id' on the Campaign contacts database. If not sent, the system will assume that the value of the paramater 'idContact' is the intern id on OML
idCampaign	string	Contact identifier to tag, its value depends on if 'idExternalSystem' parameter is sent

Also, it must send the values of the fields correspondent to the contactdatabase, and their names can be obtained with using the endpoint: Obtain Contact databse structure (*Endpoint to obtain Contact database structure*). Is mandatory to send the value to the field 'main_phone', and in case the database has external id, the field's value 'external_id' mustn't exist previously in other contact of database.

In case of no errors ocurred it will show an output like this, with data of the new disposition created



In case of errors ocurred, the endpoint returns a JSON with the field 'status':'ERROR' and the detailed information of the error on the field'errors'. On other case the 'status' field value will be 'OK'

16.1.4 Call generator endpoint

Allows to generate calls (click to call) from an External CRM System.The credentials must belong to an Agent

URL: POST https://<omnileads_addr>/api/v1/makeCall

```
{
'idExternalSystem': <idExternalSystem> :String,
'idAgent': <idAgent> :String,
'idContact': <idContact> :String,
'idCampaign': <idCampaign> :String,
'phone': <phoneContact> :String
}
```

figure 4: endpoint new call request

filed name	type	description
idExternalSystem	string	Optional parameter, it must be sent if needed to link campaign with the external CRM system
idCampaign	string	Required parameter, must match with an OML campaign identifier. If the parameter 'idExternalSystem' is sent, it must match with the field "external identifier" of a campaign associated to the External System specified
idAgent	string	Required parameter, must match to a system Agent identifier. If the parameter 'idExternalSystem' is sent must match to the field "external identifier"of an Agent associated to the external CRM System
idContact	string	Optional parameter, if is not sent the system assumes that is a new contact. If sent must match with an identifier of the campaing databasecontact. If the 'idExternalSystem' is sent, it must match with the contacts database field marked as an external identifier

In case of errors ocurred, the endpoint returns a JSON with the field 'status':'ERROR' and the detailed information of the error on the field'errors'. On other case the 'status' field value will be 'OK'

16.1.5 Disposition options list endpoint

URL GET https://<omnileads_addr>/api/v1/campaign/<idc:integer>/dispositionOptions/ (1)

URL GET https://<omnileads_addr>/api/v1/campaign/<idc:string>/dispositionOptions/<ids:integer>/ (2)

This endpoint allows to get a disposition options list avalaible for tag a contact on a campaign. The credentials must belong to an Agent (*Create agents*)

The parameters for this endpoint must be specified on the url. It has two modes to use, if it uses the (1) mode, with a single parameter, the 'idc' parameter value must be an integer specifying the OML intern campaignidentifier

The mode (2) is for using the endpoint from an external CRM system to OML and in this case the parameter 'ids' must indicates an external CRM system id and the 'idc' parameter must indicates the identifier of one campaign in this external system

In case of the execution without errors the endpoint will return a disposition options list like the following:

```
HTTP 200 OK
Allow: GET
Content-Type: application/json
Vary: Accept

[
    {
        "id": 44,
        "name": "Cuelga enojado"
    },
    {
        "id": 45,
        "name": "Ya es cliente"
    },
    {
        "id": 43,
        "name": "Venta"
    }
]
```

In case that the id does not match with an id of a campaign or CRM system the endpoint will return an output like:

```
HTTP 404 Not Found
Allow: GET
Content-Type: application/json
Vary: Accept

{
    "detail": "No encontrado."
}
```

16.1.6 Dispositions list endpoint

This endpoint allows get a dispositions list made by the agent who make the request. (*Create agents*)

URL: GET https://<omnileads_addr>/api/v1/disposition/

In case of no errors ocurred, it returns the dispositions list made it by the agent

```
HTTP 200 OK
Allow: GET, POST
Content-Type: application/json
Vary: Accept

[
    {
        "id": 35,
        "idContact": 48,
        "callid": "96855041",
        "idDispositionOption": 131,
        "comments": "Create peace."
    },
    {
        "id": 36,
        "idContact": 49,
        "callid": "93178525",
        "idDispositionOption": 132,
        "comments": "Me indeed open."
    }
]
```

16.1.7 Create new disposition endpoint

This endpoint allows to "tag" the result of a management about a contact. When a CRM user ends a management, it is normal that management closes with a disposition made, and using this endpoint an External CRM System can integrate this action to OML. The credentials used must belong to an Agent (*Create agents*).

URL: POST https://<omnileads_addr>/api/v1/disposition/

filed name	type	description
idExternalSystem	integer	Optional parameter, if sent the system tries to locate the contact as 'external_id' on the Campaign contacts database. If not sent, the system will assume that the value of the paramater 'idContact' is the intern id on OML
idContact	string	Contact identifier to tag, its value depends on if 'idExternalSystem' parameter is sent
idDispositionOption	integer	The disposition option campaign id that will be used to tag the contact, each campaign defines its disposition options. See the endpoint that allows to obtain that values
callid	string	Optional parameter, call identifier
comments	string	The agent observations in the disposition

In case of no errors ocurred it will show an output like this, with data of the new disposition created

```
HTTP 201 Created
Allow: GET, POST
Content-Type: application/json
Vary: Accept

{
    "id": 37,
    "idContact": 25,
    "callid": "93178525",
    "idDispositionOption": 60,
    "comments": "Me indeed open."
}
```

If an attempt to create a new disposition instance is made, to a contact already tagged on the campaign, the endpoint will return the following error:

```
HTTP 400 Bad Request
Allow: GET, POST
Content-Type: application/json
Vary: Accept

{
    "status": "ERROR",
    "msg": "There is another disposition for this contact on this campaign"
}
```

If the contact id on the campaign database is not found the endpoint will return the following error:

```
HTTP 400 Bad Request
Allow: GET, POST
Content-Type: application/json
Vary: Accept

{
    "contact": "Contact id not found"
}
```

If the disposition option id is not found the endpoint will return the following error:

```
HTTP 400 Bad Request
Allow: GET, POST
Content-Type: application/json
Vary: Accept

{
    "idDispositionOption": "Disposition option id not found"
}
```

16.1.8 Create new contact and assign it a new disposition endpoint

This endpoint allows to 'tag' a management and, at a same time, to create a contact, it means that it creates the contact and the disposition is linked to this new contact. The credentials used must belong to an Agent (*Create agents*).

URL: POST https://<omnileads_addr>/api/v1/new_contact/disposition/

filed name	type	description
phone	string	The contact phone number
idExternalContact	string	Optional parameter, the contact id on an external CRM system
idDispositionOption	integer	The disposition option campaign id that will be used to tag the contact, each campaign defines its disposition options. See the endpoint that allows to obtain that values
comments	string	The agent observations in the disposition
callid	string	Optional parameter, call identifier
<optional_bd_field>	string	Optional parameters, they can define values to fill the custom data" of the contact that will be created, the field names must match with the fields of campaign database

In case of no errors ocurred it will show an output like this, with data of the new disposition created

```
HTTP 201 Created
Allow: GET, POST
Content-Type: application/json
Vary: Accept

{
    "id": 37,
    "idContact": 25,
    "callid": "93178525",
    "idDispositionOption": 60,
    "comments": "Me indeed open."
}
```

If the disposition option id is not found the endpoint will return the following error:

```
HTTP 400 Bad Request
Allow: GET, POST
Content-Type: application/json
Vary: Accept

{
    "idDispositionOption": "Disposition option id not found"
}
```

16.1.9 Disposition update endpoint

This endpoint allows to update an existent disposition in OMniLeads

The credentials must belong to an Agent (*Create agents*)

URL: PUT https://<omnileads_addr>/api/v1/disposition/<idDisposition>

filed name	type	description
idExternalSystem	integer	Optional parameter, if sent the system tries to locate the contact as 'external_id' on the Campaign contacts database. If not sent, the system will assume that the value of the paramater 'idContact' is the intern id on OML
idContact	string	Contact identifier to tag, its value depends on if 'idExternalSystem' parameter is sent
idDispositionOption	integer	The disposition option campaign id that will be used to tag the contact, each campaign defines its disposition options. See the endpoint that allows to obtain that values
callid	string	Optional parameter, call identifier
comments	string	The agent observations in the disposition

If doesn't exist, the endpoint returns the following output:

```
HTTP 200 OK
Allow: GET, PUT
Content-Type: application/json
Vary: Accept

{
    "id": 35,
    "idContact": 48,
    "callid": "96855041",
    "idDispositionOption": 131,
    "comments": "Create peace. asdasd"
}
```

If in the url a non-existent disposition id is specified, the endpoint will return the following output error:

```
HTTP 404 Not Found
Allow: GET, PUT
Content-Type: application/json
Vary: Accept

{
    "detail": "No encontrado."
}
```

If the disposition instance is tried to be modified, changing the parameters 'idContact' and 'idDispositionOption' the system detects that this would make to disposition for the oen contact on the same campaing the endpoint will return the following error output:

```
HTTP 400 Bad Request
Allow: GET, POST
Content-Type: application/json
Vary: Accept

{
    "status": "ERROR",
    "msg": "There is another disposition for this contact on this campaign"
}
```

If the contact id on the campaign database is not found the endpoint will return the following error:

```
HTTP 400 Bad Request
Allow: GET, POST
Content-Type: application/json
Vary: Accept

{
    "contact": "Contact id not found"
}
```

If the disposition option id is not found the endpoint will return the following error:

```
HTTP 400 Bad Request
Allow: GET, POST
Content-Type: application/json
Vary: Accept

{
    "idDispositionOption": "Disposition option id not found"
}
```

16.1.10 API de Sesión de Agente en Asterisk

Endpoints de la API utilizados por el WebPhone mediante los cuales puede controlar las sesiones del agente en Asterisk.

API de sesión de Agente en Asterisk

Endpoints para controlar la sesion del agente en Asterisk para el uso del Webphone

Inicio de sesión de agente en Asterisk

Establece el estado de sesión del Agente (*Create agents*) en Asterisk como iniciada. Las credenciales deberán pertenecer al Agente, y no hace falta enviar ningún parámetro extra.

URL: POST https://<omnileads_addr>/api/v1/asterisk_login/

Cierre de sesión de agente en Asterisk

Establece el estado de sesión del Agente (*Create agents*) en Asterisk como finalizada. Las credenciales deberán pertenecer al Agente, y no hace falta enviar ningún parámetro extra.

URL: POST https://<omnileads_addr>/api/v1/asterisk_logout/

Ingreso en pausa de agente

Establece el estado de sesión del Agente (*Create agents*) en Asterisk como en Pausa (*Pauses*). Las credenciales deberán pertenecer al Agente.

URL: POST https://<omnileads_addr>/api/v1/asterisk_pause/

field name	type	description
pause_id	string	Id de la pausa en la que entra el Agente.

Salida de pausa de agente

Establece el estado de sesión del Agente (*Create agents*) en Asterisk como "Disponible" e indica ela finalizacion de una Pausa (*Pauses*). Las credenciales deberán pertenecer al Agente.

URL: POST https://<omnileads_addr>/api/v1/asterisk_unpause/

field name	type	description
pause_id	string	Id de la pausa de la que sale el Agente.

CHAPTER 17

KNOWN ISSUES

17.1 Known Issues

- disposition form not showing up when receiving a transfer call
- Pause times are not separated by date
- Supervision calls capture is broken
- Recycle over recycled campaign duplicates records to call
- The system does not reproduce some temporary voice messages produced by some carriers on the webphone
- The mysql password cannot be changed using –change-passwords
- The restore of the system doesn't include the Wombat Dialer database so the dialer campaigns are useless after restore

CHAPTER 18

RELEASE NOTES

18.1 Release Notes

Febrero 7, 2020

18.1.1 Detalles de Release 1.4.0

New features

- OML can be deployed and integrated now with Issabel PBX & FreePBX using docker
- The system changed its default trunk technology from SIP to PJSIP
- Asterisk agents endpoints were migrated from chan_sip to PJSIP
- The internal interaction with Asterisk from agent console component was refactored touse AMI protocol
- The installation on Docker for production was refined
- PostgreSQL component was updated to version 11
- The CRM interaction was refined with the addition of new endpoints on the public API
- kamailio.cfg file was refactored and optimized
- Database logger trigger script was ported from plpython to plperl

Bug fixes

- Solucionado bug que contaba como ABANDONWEL los eventos de llamadas entrantes que excedían el límite de llamadas en cola configurado that exceeds the queue calls limit, afectando a los reportes
- Fixed bug on telephony list views (on paginators)
- Fixed bug that counted as EXPIRED inbound calls with failover configured to point to a custom destination, affecting reports

- Fixed infinite redirection bug on removed agents
- Fixed AstDB inconsistency bug when remove agents
- Fixed bug that generated an incorrect callid for manual click2call generating errors on campaign reports
- Fixed bug that inhibits to edit boost factor in dialer campaings
- Fixed error generated on inbound calls supervision view when there was enqueued dialer calls
- Fixed bug that remove constance OML data on docker installation after Kamailio restarts
- Fixed bug that sent an incorrect id after ends an ACW pause
- Fixed error on agents view pagination on supervision