DESARROLLO TERRITORIAL
SECRETARÍA DE DESARROLLO AGRARIO, TERRITORIAL Y URBANO

CONAVI
COMISIÓN NACIONAL DE VIVIENDA

MANUAL TÉCNICO: SISTEMA NACIONAL DE INDIADORES DE VIVIENDA (SNIIV)

CLAVE DE IDENTIFICACIÓN

AGOSTO 2021

Contenido

I. CONSIDERACIONES.

I.1. Objetivo.

Conforme a lo establecido en los artículos 43, 44 y 45 de la Ley de Vivienda, el Sistema Nacional de Información e Indicadores de Vivienda (SNIIV), tiene el objetivo de integrar, generar y difundir la información que se requiera para la adecuada planeación, instrumentación y seguimiento de la Política Nacional de Vivienda, así como para el fortalecimiento de la oferta articulada de vivienda en el país. El presente documento tiene como finalidad la descripción de la construcción técnica del sistema.

De igual forma, de acuerdo con lo establecido en el Convenio Modificatorio al Convenio de Coordinación para la entrega del Sistema Nacional de Información de Indicadores de Vivienda, el 30 de septiembre de 2021, la Comisión Nacional de Vivienda hará entrega del Sistema a la Dirección General de Desarrollo Urbano, Suelo y Vivienda (SEDATU). No obstante, la CONAVI debe continuar con la publicación la información de los programas que opera, la información que investiga, así como los documentos de análisis y la información estadística que genera, dando cumplimiento a los principios de transparencia y rendición de cuentas.

I.2. Marco Jurídico.

- Constitución Política de los Estados Unidos Mexicanos.
- Ley de Vivienda.
- Ley Federal de las Entidades Paraestatales.
- Ley Federal de Presupuesto y Responsabilidad Hacendaria.
- Ley Federal de Transparencia y Acceso a la Información Pública.
- Plan Nacional de Desarrollo (2019-2024).
- Programa Sectorial de Desarrollo Agrario, Territorial y Urbano (2020-2024).
- Programa Nacional de Vivienda (2019-2024).
- Estatuto Orgánico de la Comisión Nacional de Vivienda.
- Reglas de Operación del Programa de Vivienda Social.
- Reglas de Operación del Programa Nacional de Reconstrucción
- Manual de Organización de la Comisión Nacional de Vivienda.
- Manual de Procedimientos para la Operación del Programa de Vivienda Social en el esquema de cofinanciamiento.
- Guía para la elaboración del manual de procedimientos de la Comisión Nacional de Vivienda

De acuerdo con el capítulo III de la Ley de Vivienda, donde se establecen las modificaciones de las funciones de la Comisión Nacional de Vivienda, fracción VII del artículo 19 donde se establece que la Comisión Nacional de Vivienda debe: "Desarrollar, ejecutar y promover esquemas, mecanismos y programas de financiamiento, subsidio y ahorro previo para la vivienda, en sus diferentes tipos y modalidades, priorizando la atención a la población en situación de pobreza, coordinando su ejecución con las instancias correspondientes;"

No obstante, es importante hacer referencia al Convenio Modificatorio al Convenio de Colaboración celebrado entre la Secretaria de Desarrollo Agrario, Territorial y Urbano (SEDATU) a través de la Dirección General de Desarrollo Urbano, Suelo y Vivienda y la Comisión Nacional de Vivienda (CONAVI) representada la Dirección General, el cual tiene como objetivo establecer las acciones tendrá a bien operar y funcionar el SNIIV hasta su entrega programada el 30 de septiembre de 2021.

Asimismo, conforme a lo establecido en los artículos 43, 44 y 45 de la Ley de Vivienda, el Sistema Nacional de Información e Indicadores de Vivienda (SNIIV), tiene el objetivo de integrar, generar y difundir la información que se requiera para la adecuada planeación, instrumentación y seguimiento de la Política Nacional de Vivienda, así como para el fortalecimiento de la oferta articulada de vivienda en el país.

El SNIIV es de acceso público, sin costo en ambiente web y que actualmente se encuentra disponible en la página de la Comisión Nacional de Vivienda (CONAVI), con el fin de mostrar y ofrecer información sobre la evolución de los principales indicadores del sector de la vivienda.

El SNIIV se encuentra dirigido a productores, constructores, entidades financieras, ONAVIS, OREVIS, inversionistas, instituciones académicas y de investigación, notarios, actores que participan en alguna etapa del proceso habitacional, así como público en general nacional y extranjero.

Dentro de la gran variedad de oportunidades que en materia de datos tiene el SNIIV, la principal es concentrar en un mismo lugar la información e indicadores relacionados con el sector de la vivienda. Si bien, mucha de esta información ya existía, no estaba a disposición del público de manera periódica, ni a través de herramientas de reporte amigables y con la posibilidad de extracción de datos.

I.3. Referencias.

Sistema Nacional de Información e Indicadores de Vivienda (SNIIV), disponible en: https://sniiv.conavi.gob.mx/

I.4. Alcance.

El SNIIV está dirigido a personal de la CONAVI, entidades financieras, ONAVIS, instituciones académicas y de investigación, notarios, actores que participan en alguna etapa del proceso habitacional, así como público en general nacional y extranjero.

Este documento normativo está dirigido a personal de la CONAVI con el fin de registrar la información técnica dentro del Sistema. Este documento puede ayudar a establecer una base para registrar la programación del SNIIV.

I.5. Vigencia.

Sin vigencia

I.6. Glosario

Diagrama de componentes: Representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes.

Esquema: describe la estructura de una base de datos, en un lenguaje formal soportado por un sistema de gestión de base de datos.

Framework: Un entorno de trabajo, o marco de trabajo es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

Host: Computadoras u otros dispositivos (tabletas, móviles, portátiles) conectados a una red que proveen y utilizan servicios de ella.

Información normalizada: La normalización de bases de datos es un proceso que consiste en designar y aplicar una serie de reglas a las relaciones obtenidas tras el paso del modelo entidad-relación al modelo relacional con objeto de minimizar la redundancia de datos.

Interface: Medio común para que los objetos no relacionados se comuniquen entre sí. Estas son definiciones de métodos y valores sobre los cuales los objetos están de acuerdo para cooperar.

Interoperabilidad: Es la capacidad de dos o más sistemas o componentes para intercambiar información y usar la información que se ha intercambiado.

Modelo multidimensional: Se compone de cubos y dimensiones que se pueden anotar y ampliar para admitir construcciones de consultas complejas.

Modelo relacional: En el modelo relacional se utiliza un grupo de tablas para representar los datos y las relaciones entre ellos.

Open source: Es el software cuyo código fuente y otros derechos que normalmente son exclusivos para quienes poseen los derechos de autor, son publicados bajo una licencia de código abierto o forman parte del dominio público.

PATH: Variable de entorno de los sistemas operativos.

Procedimiento almacenado: es un conjunto de instrucciones de T-SQL que SQL Server compila, en un único plan de ejecución, los llamados "store procedures" se encuentran almacenados en la base de datos, los cuales pueden ser ejecutados en cualquier momento.

Puerto: Los puertos son asignados por el sistema operativo de tu dispositivo cada vez que un proceso va a hacer un pedido por el internet.

Representación cartográfica: Establece una relación ordenada entre los puntos de la superficie curva de la Tierra y los de una superficie plana.

Script: Secuencia de comandos o guion es un término informal que se usa para designar a un programa relativamente simple.

Servidor virtual: Es la simulación de un servidor físico en un entorno virtual. Esto permite ejecutar varios servidores virtuales en una máquina física.

Shapefile: Es un formato para bases de datos geoespaciales y vectoriales en sistemas de información geográfica (en inglés, GIS – Geographic Information System).

Sistema Operativo: Es un conjunto de programas que permite manejar la memoria, disco, medios de almacenamiento de información y los diferentes periféricos o recursos de nuestra computadora.

Términos de licencia: Es un contrato entre el licenciante (autor/titular de los derechos de explotación/distribución) y el licenciatario (usuario consumidor, profesional o empresa) del programa informático, para utilizarlo cumpliendo una serie de términos y condiciones establecidas dentro de sus cláusulas.

Webservices: Es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.

Listado de acrónimos

AJAX. Asynchronous JavaScript And XML.
APP. Application.
ASP. Active Server Pages.
CDDL. Common Development and Distribution License.
CONAVI. Comision Nacional de Vivienda.
CRUD. Create, Read, Update and Delete.
CSS. Cascading Style Sheets.
CSV. Comma-Separated Values.
DAO. Data Access Object.
DLL. Dynamic Link Library.
ETL. Extract, Transform and Load.
FONHAPO. Fideicomiso Fondo Nacional de Habitaciones Populares.
FOVISSSTE. Fondo de la Vivienda del Instituto de Seguridad y Servicios Sociales de los Trabajadores del Estado.
GNU GPL. GNU General Public License.
HTML. HyperText Markup Language.
HTTP. Hypertext Transfer Protocol.
HTTPS. Hypertext Transfer Protocol Secure.
IDE. Integrated Development Environment.
IIS. Internet Information Services.
IMSS. *Instituto Mexicano del Seguro Social.*
INFONAVIT. Instituto del Fondo Nacional de la Vivienda para los Trabajadores.
IP. *Internet Protocol.*
IPYNB. IPython Notebook.
JSON. JavaScript Object Notation.
OGC. Open Geospatial Consortium.

PCU. Perímetros de Contención Urbana.
PMU. Programa de Mejoramiento Urbano.
RUV. Registro Único de Vivienda.
SEDATU. Secretaría de Desarrollo Agrario, Territorial y Urbano.
SHF. Sociedad Hipotecaria Federal.
SHP. Shapefile.
SISEVIVE. Sistema de Evaluación de la Vivienda Verde.
SNIIV. Sistema Nacional de Información e Indicadores de Vivienda.
SOA. Service-Oriented Architectures.
SSH. Secure Shell.
TFS. *Team Foundation Server*.
TXT. Textfile.
VO. Value Object.
WCF. Windows Communication Foundation.
WGS 84. World Geodetic System 1984.
WMS. *Web Map Service*.
XHTML. Extensible Hypertext Markup Language.
XLS. Microsoft Excel Spreadsheet.
XLSX. Microsoft Excel Open XML Spreadsheet.
ZIP. Compressed file archive.

II. Introducción

En el capítulo III de la Ley de Vivienda se modifica las funciones de la Comisión Nacional de Vivienda. De acuerdo con la fracción VII del artículo 19 donde se establece que la Comisión Nacional de Vivienda debe: "Desarrollar, ejecutar y promover esquemas, mecanismos y programas de financiamiento, subsidio y ahorro previo para la vivienda, en sus diferentes tipos y modalidades, priorizando la atención a la población en situación de pobreza, coordinando su ejecución con las instancias correspondientes;"

De acuerdo con lo establecido en los artículos 43,44 y 45 de la Ley de Vivienda, el Sistema Nacional de Información e Indicadores de Vivienda (SNIIV), tiene el objetivo de integrar, generar y difundir la información que se requiera para la adecuada planeación, instrumentación y seguimiento de la Política Nacional de Vivienda, así como para el fortalecimiento de la oferta articulada de vivienda en el país.

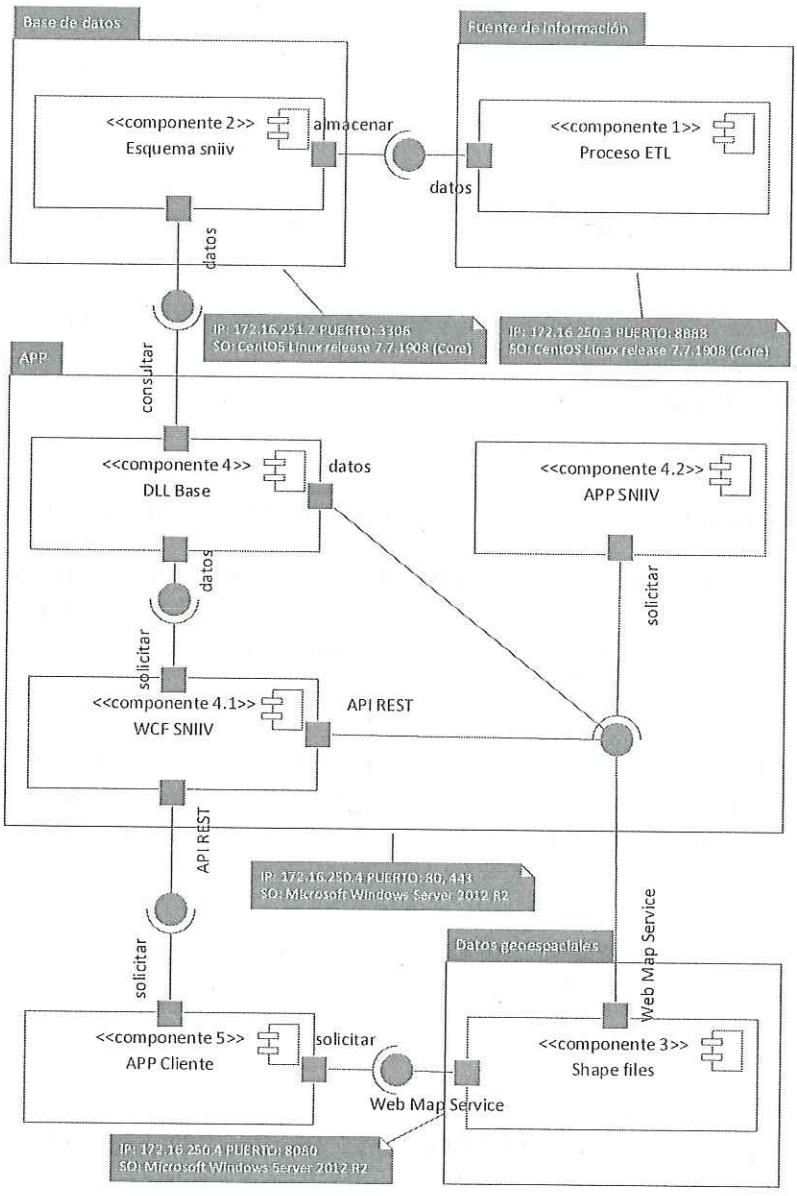
En referencia a la reforma a la Ley de Vivienda, en su artículo 16 Fracción XIV que menciona "Coordinar la operación y funcionamiento del Sistema de Información" y de acuerdo con el convenio de coordinación celebrado entre la Secretaria de Desarrollo Agrario, Territorial y Urbano (SEDATU) a través de la Dirección General de Desarrollo Urbano, Suelo y Vivienda y la Comisión Nacional de Vivienda (CONAVI) representada la Dirección General, el cual tiene como objetico establecer las acciones tendrá a bien operar y funcionar hasta su entrega programada el 31 de diciembre de 2020.

III. Componentes

Diagrama de componentes

El SNIIV consta principalmente de siete componentes que se relacionan entre sí en una arquitectura de software.

Diagrama 1 Diagrama de componentes del SNIIV



Fuente: CONAVI, 2021

1. Componente 1

La carga de información se realiza mediante Anaconda, una suite de código abierto que abarca una serie de aplicaciones, una de las aplicaciones es el IDE Jupyter Notebook, un entorno de trabajo interactivo que permite desarrollar código en Python de manera dinámica para el análisis, tratamiento y normalización de la información de cada uno de los organismos que conforman el SNIIV.

2. Componente 2

Posteriormente al análisis, tratamiento y normalización de la información, se almacena en el servidor de base de datos MySQL del SNIIV.

3. Componente 3

Para la publicación de datos espaciales WMS (Web Map Service) del SNIIV se requiere del servidor GeoServer, para su correcto funcionamiento es necesario instalar el servidor Glassfish (puerto 4848).

Las capas geográficas del SNIIV se almacenan en formato Shapefile mediante WMS, un servicio de representación cartográfica que se utiliza para publicar un grupo de capas de mapa a fin de integrarlas en mapas web interactivos del SNIIV.

4. Componente 4

Para la publicación de la aplicación del SNIIV se requiere de IIS (Internet Information Server).

BASE contiene todas las clases del proyecto SNIIV, con el fin de utilizarlas como librerías DLL (Dynamic Link Library) previamente compiladas que constan de código ejecutable. Contiene principalmente las clases VO (Value Object) y DAO (Data Access Object) para consultar la información de Base de Datos.

5. Componente 4.1

WCF (Windows Communication Foundation) es un modelo de programación para el desarrollo de aplicaciones con arquitectura orientada a servicios (SOA) que nos permitirá solicitar datos de DDL BASE y mostrar información en formato JSON.

6. Componente 4.2

Para el funcionamiento y la publicación del sitio web del SNIIV requiere solicitar información de DDL BASE, los servicios WCF en formato JSON y los servicios WMS.

7. Componente 5

La App Cliente es cualquier aplicación que requiera conectarse para consultar información del SNIIV de manera dinámica, oportuna y actualizada de los programas CONAVI, los Financiamientos en Vivienda que otorgan las entidades financieras, la Oferta de Vivienda del Registro Único de Vivienda (RUV) y la información geográfica de la Oferta de Vivienda, el Sistema de Evaluación de la Vivienda Verde (SISEVIVE) y los Perímetros de Contención Urbana (PCU). Por cuestiones de interoperabilidad

este componente puede ser consumido mediante webservices por otras instituciones, ejemplo, INFONAVIT y SEDATU.

IV. Servidores

Actualmente la SEDATU (Secretaría de Desarrollo Agrario, Territorial y Urbano) posee la infraestructura que permiten procesar, gestionar y publicar la información del sector vivienda en el SNIIV.

Para facilitar y reducir costos de poseer y administrar el SNIIV se implementan servidores virtuales que difiere de las infraestructuras tradicionales. La virtualización hace que las infraestructuras sean más simples y eficientes, permitiendo que las aplicaciones se implementen más rápido y que el rendimiento y la disponibilidad aumenten.

La infraestructura del SNIIV cuenta con tres servidores:

Servidor Data Warehouse

El servidor cuenta con un Sistema Operativo CentOS Linux release 7.7.1908 (Core), con las siguientes especificaciones técnicas;

Tabla 2.1 Especificaciones técnicas del Servidor Data Warehouse

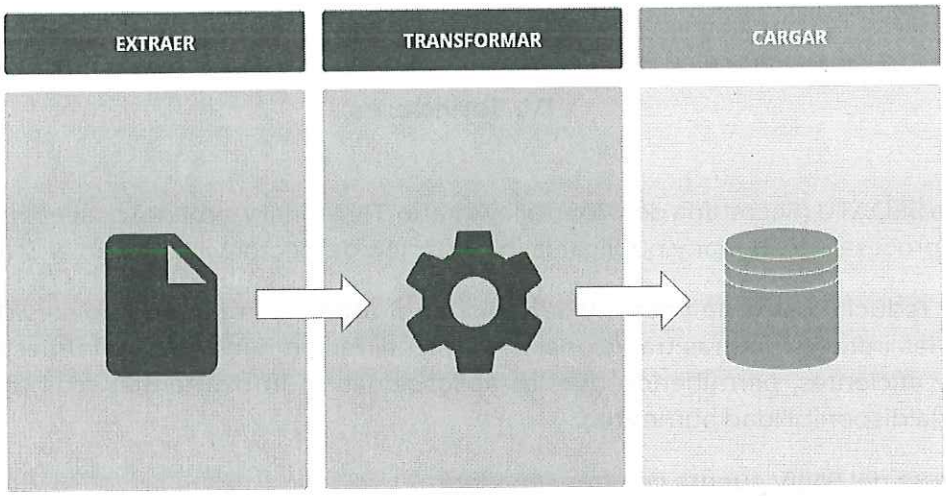
Almacenamiento	Total: 536GB Partición de sistema: 260 MB Partición de almacenamiento: 520 GB
Memoria RAM	32GB
Procesador	CPU(s): 8
Programas instalados	Conda versión: 4.8.2 Conda-build versión: 3.17.6 Python versión: 3.7.6.final.0
IP	172.16.250.3
Puertos	8888 (Jupyter Notebook)

El Data Warehouse del SNIIV integra información para luego ser tratada mediante procesos ETL de diferentes fuentes de los organismos en un almacén de datos único y centralizado; CONAVI, INFONAVIT, FOVISSSTE, SHF, FONHAPO, CNBV, RUV, etc.

Proceso ETL

Para el proceso de la información, el Data Waterhouse del SNIIV cuenta con herramientas que permitirán extraer, transformar y cargar la información. Durante este proceso, los datos se toman (extraen) de un sistema de origen, se convierten (transforman) en un formato que se puede almacenar y se almacenan (cargan) en el servidor Data Warehouse, servidor Base de Datos y reportes específicos que requiere la CONAVI para su análisis y toma de decisiones.

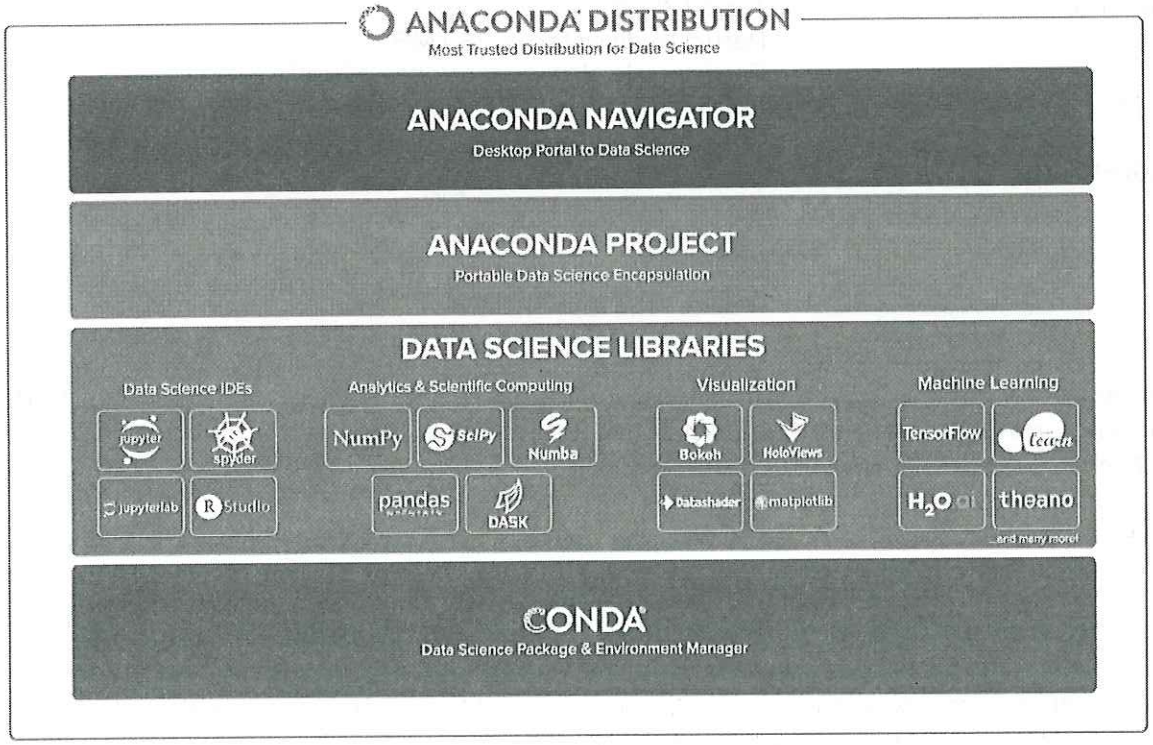
Imagen 2.1 Proceso ETL



Herramientas

Anaconda es una Suite de código abierto que abarca una serie de aplicaciones, librerías y conceptos diseñados para el desarrollo de la Ciencia de datos con Python.

Imagen 2.2 Anaconda Distribution



La carga de información se realiza mediante el IDE Jupyter Notebook que es un entorno de trabajo interactivo que permite desarrollar código en Python de manera dinámica, a la vez que integrar en un mismo documento tanto bloques de código como texto, gráficas o imágenes. En este entorno se implementan librerías como; Pandas, NumPy, Matplotlib, etc. Para el análisis y proceso de la información de cada uno de los organismos que conforman el SNIIV.

Instalación y configuración Anaconda Python

1. Descargar el script de instalación de Anaconda.

   Navegar en el directorio **/tmp** y descargar el script de instalación de Anaconda usando el enlace que copiado de la página de descargas:

```
cd /tmp
curl -O https://repo.anaconda.com/archive/Anaconda3-5.3.1-Linux-x86_64.sh
```

2. Verificar la integridad de los datos del script.

   Utilizar **sha256sum** comando para verificar la suma de comprobación del script:

```
sha256sum Anaconda3-5.3.1-Linux-x86_64.sh
```

   Se deberá ver una salida como la siguiente:

```
d4c4256a8f46173b675dd6a62d12f566ed3487f932bab6bb7058f06c124bcc27  Anaconda3-5.3.1-Linux-x86_64.sh
```

   Asegurar que el **hash** impreso del comando anterior coincida con el disponible en Anaconda con Python 3 en la página de Linux de 64 bits para su versión apropiada de Anaconda.

```
https://docs.anaconda.com/anaconda/install/hashes/Anaconda3-5.3.1-Linux-x86_64.sh-hash.html
```

3. Ejecutar el script de instalación de Anaconda

```
bash Anaconda3-5.3.1-Linux-x86_64.sh
```

   Se deberá ver una salida como la siguiente:

```
Welcome to Anaconda3 5.3.1


In order to continue the installation process, please review the license agreement.

Please, press ENTER to continue
```

Presionar ENTER para continuar, posteriormente presione nuevamente ENTER para desplazarse por la licencia. Una vez que terminado la revisión de la licencia, se solicitará que se aprueben los términos de licencia:

```
Do you accept the license terms? [yes|no]
```

Escribir **yes** para aceptar la licencia, después se solicitará elegir alguna ubicación de instalación.

```
Anaconda3 will now be installed into this location:

/home/python/anaconda3


- Press ENTER to confirm the location

- Press CTRL-C to abort the installation

- Or specify a different location below
```

La ubicación predeterminada está bien para la mayoría de los usuarios, presionar ENTER para confirmar la ubicación y el proceso de instalación continuará.

En caso de obtener el error **bunzip2: command not found**, instalar el **bzip2paquete** con el siguiente comando:

```
sudo yum install bzip2.
```

La instalación llevará algún tiempo, una vez completada, se mostrará el siguiente resultado:

```
Installation finished.


Do you wish the installer to initialize Anaconda3

in your /home/python/.bashrc ? [yes|no]
```

Si se desea utilizar el conda tipo de comando, seleccione **yes** y presione ENTER, se le presentará el siguiente resultado:

```
Appending source /home/python/anaconda3/bin/activate to /home/python/.bashrc

A backup will be made to: /home/python/.bashrc-anaconda3.bak


For this change to become active, you have to open a new terminal.
```

```
Thank you for installing Anaconda3!
```

El instalador también solicitará si desea descargar e instalar Visual Studio Code.

```
Anaconda is partnered with Microsoft! Microsoft VSCode is a streamlined
code editor with support for development operations like debugging, task
running and version control.

To install Visual Studio Code, you will need:
- Administrator Privileges
- Internet connectivity

Visual Studio Code License: https://code.visualstudio.com/license

Do you wish to proceed with the installation of Microsoft VSCode? [yes|no]
```

Para activar la instalación de Anaconda, cargar la nueva **PATH** variable de entorno que fue agregada por el instalador de Anaconda en la sesión de shell actual con el siguiente comando:

```
source ~/.bashrc
```

4. Verificar la instalación

```
conda info
active environment : base
active env location : /home/linuxize/anaconda3
shell level : 1
user config file : /home/linuxize/.condarc
populated config files :
conda version : 4.5.11
conda-build version : 3.15.1
python version : 3.7.0.final.0
base environment : /home/linuxize/anaconda3  (writable)
channel URLs : https://repo.anaconda.com/pkgs/main/linux-64
https://repo.anaconda.com/pkgs/main/noarch
```

```
https://repo.anaconda.com/pkgs/free/linux-64
https://repo.anaconda.com/pkgs/free/noarch
https://repo.anaconda.com/pkgs/r/linux-64
https://repo.anaconda.com/pkgs/r/noarch
https://repo.anaconda.com/pkgs/pro/linux-64
https://repo.anaconda.com/pkgs/pro/noarch
package cache : /home/linuxize/anaconda3/pkgs
/home/linuxize/.conda/pkgs
envs directories : /home/linuxize/anaconda3/envs
/home/linuxize/.conda/envs
platform : linux-64
user-agent : conda/4.5.11 requests/2.19.1 CPython/3.7.0 Linux/3.10.0-957.1.3.el7.x86_64 centos/7
glibc/2.17
UID:GID : 0:0
netrc file : None
offline mode : False
```

5. Configuraciones acceso web; generar **config file**:

```
jupyter notebook --generate-config
```

Editar archivo **config file**:

```
vi /home/python/.jupyter/jupyter_notebook_config.py

c.NotebookApp.allow_origin = '*'
c.NotebookApp.allow_root = True
c.NotebookApp.ip = '0.0.0.0'
c.NotebookApp.notebook_dir = '/home/python/'
c.NotebookApp.open_browser = False
c.NotebookApp.port = 8888
```

Abrir puerto de acceso en firewall:

```
firewall-cmd --zone=public --add-port=8888/tcp –permanent
```

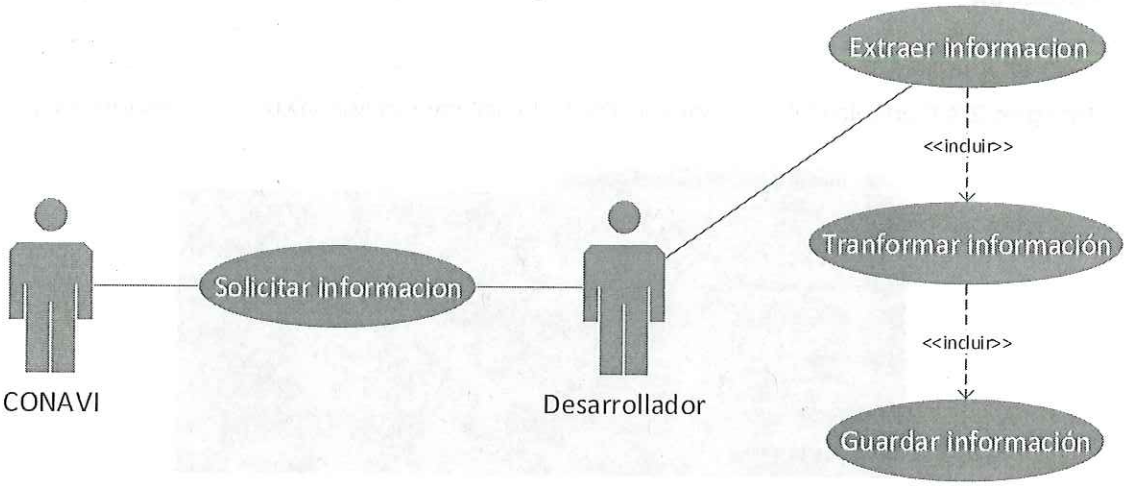
Instalación de librerías necesarias en los procesos de carga:

```
conda install pymysql

conda install rtree

conda install geopandas
```

Implementación Jupyter Notebook

1. CONAVI solicita la información estructurada o no estructurada de cada uno de los organismos que conforma el SNIIV en un almacén de datos único y centralizado; CONAVI, INFONAVIT, FOVISSSTE, SHF, FONHAPO, CNBV, RUV, IMSS etc.

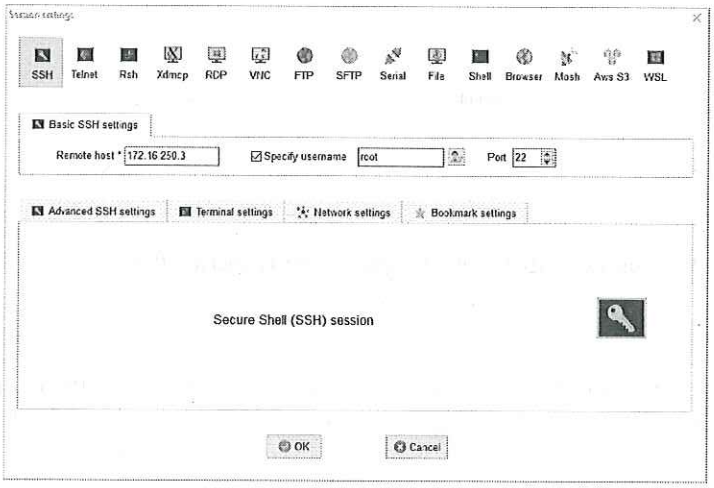
Diagrama 2 Casos de uso para el tratamiento de información



Nota: *La periodicidad, el formato de la fuente, la forma de integración de la información (acumulada, por mes, por semana, etc.) dependerá de cada organismo, esta información es detallada en el documento "Manual de cargas".*

2. Para almacenar los datos, en necesario conectarse vía remota (SSH) y acceder al directorio **data** donde se alojan los archivos fuente estructurados o no estructurados (txt, csv, xls, xlsx, zip, etc.) de cada organismo organizados por periodos.

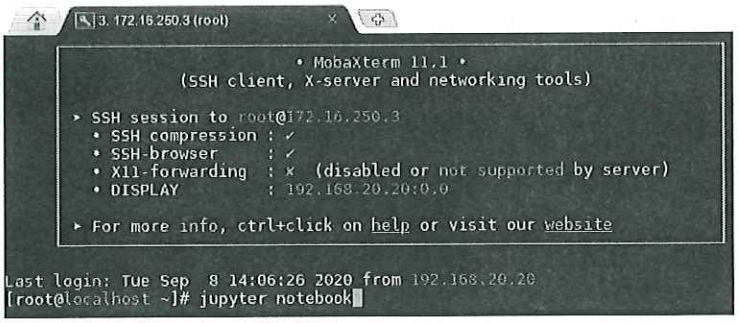
Imagen 2.3 Conexión remota vía aplicación MobaXterm



Nota: *Ver contraseñas del Servidor Data Warehouse del archivo credenciales.*

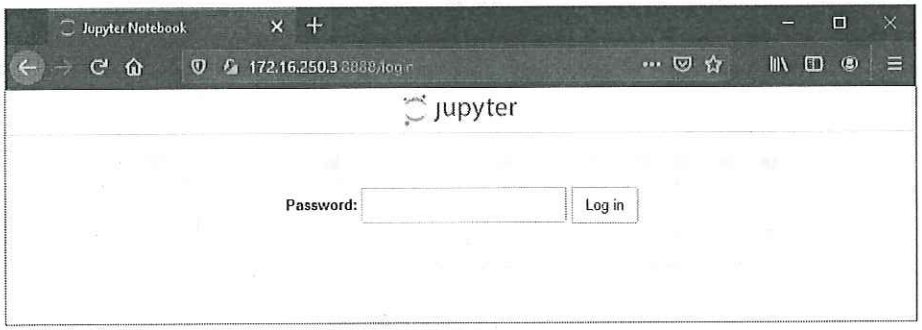
3. Posteriormente al almacenamiento de datos, el desarrollador realizar el proceso ETL ejecutando la herramienta Jupyter Notebook, para correr dicha herramienta es necesario entrar a la consola del servidor Data Warehouse y ejecutar el siguiente comando; **jupyter notebook**

Imagen 2.4 Consola del servidor mediante la aplicación MobaXterm conexión remota



4. El siguiente paso es acceder desde cualquier navegador capturando la siguiente ruta; **http://172.16.250.3:8888/login**

Imagen 2.5 Jupyter Notebook desde el navegador

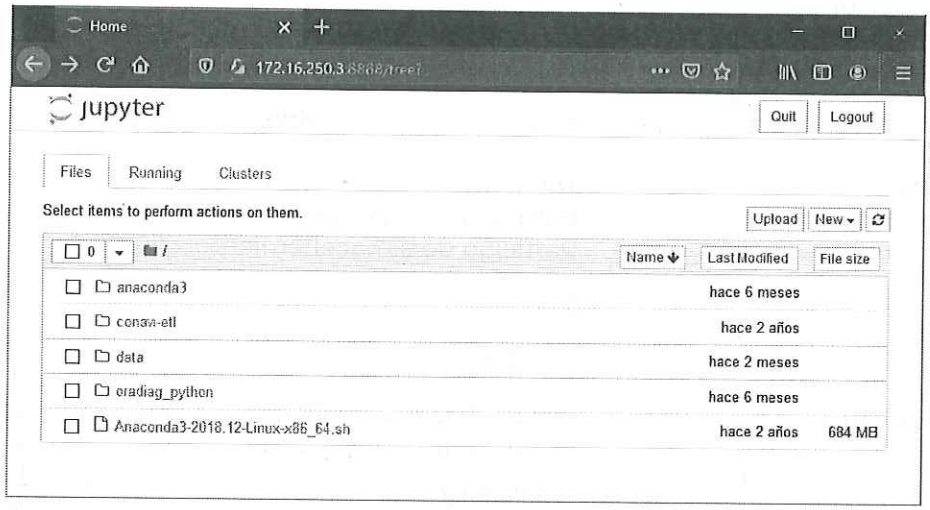


Nota: *Ver contraseña de jupyter notebook del archivo credenciales.*

Al acceder se mostrarán varias carpetas; **conavi-etl** carpeta que contiene dos directorios; **financiamientos-carga** (contiene el código para tratar la información de cada organismo que

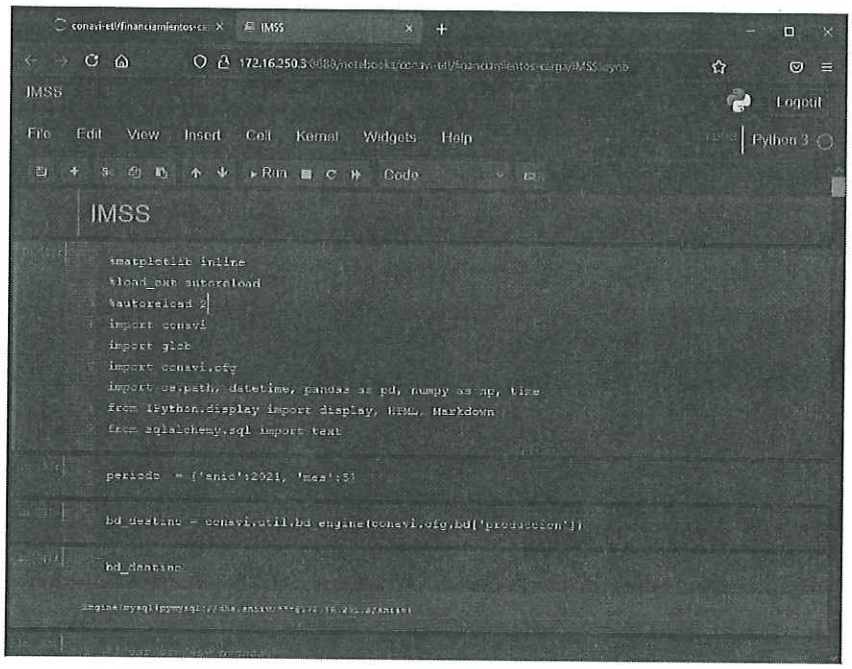
conforma el SNIIV) y **ruv-carga** (contiene el código para tratar la información del Registro Único de Vivienda), cada directorio contiene archivos IPYNB de cada proceso de información y documentación.

Imagen 2.6 Directorio en Jupyter Notebook desde el navegador



5. El último paso es ejecutar cada bloque de cada archivo IPYNB para extraer, transformar y cargar la información, los archivos IPYNB constan de documentación y código en leguaje Python para su ejecución.

Imagen 2.7 Archivo IMSS.ipynb



Servidor Base de Datos

El servidor cuenta con un Sistema Operativo CentOS Linux release 7.7.1908 (Core), con las siguientes especificaciones técnicas;

Tabla 2.2 Especificaciones técnicas del Servidor Base de Datos

Almacenamiento	Total: 751GB Partición de sistema: 500 MB Partición de almacenamiento: 660 GB
Memoria RAM	7.6G
Procesador	CPU(s): 4
Programas instalados	MySQL Ver 8.0.19 for Linux on x86_64 (MySQL Community Server - GPL)
IP	172.16.251.2
Puertos	3306 (MySQL)

Implementación MySQL

1. En casos excepcionales, con el paso del tiempo los organismos categorizan la información añadiendo nuevos datos en sus catálogos, el desarrollador tiene la tarea de agregar esos datos a los catálogos de la base de datos del SNIIV. El proceso ETL detecta cuando existe una nueva categoría.

Diagrama 3 Casos de uso para el almacenamiento de información normalizada en Base de Datos

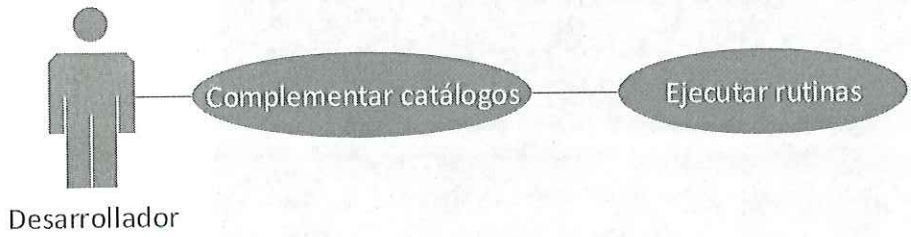
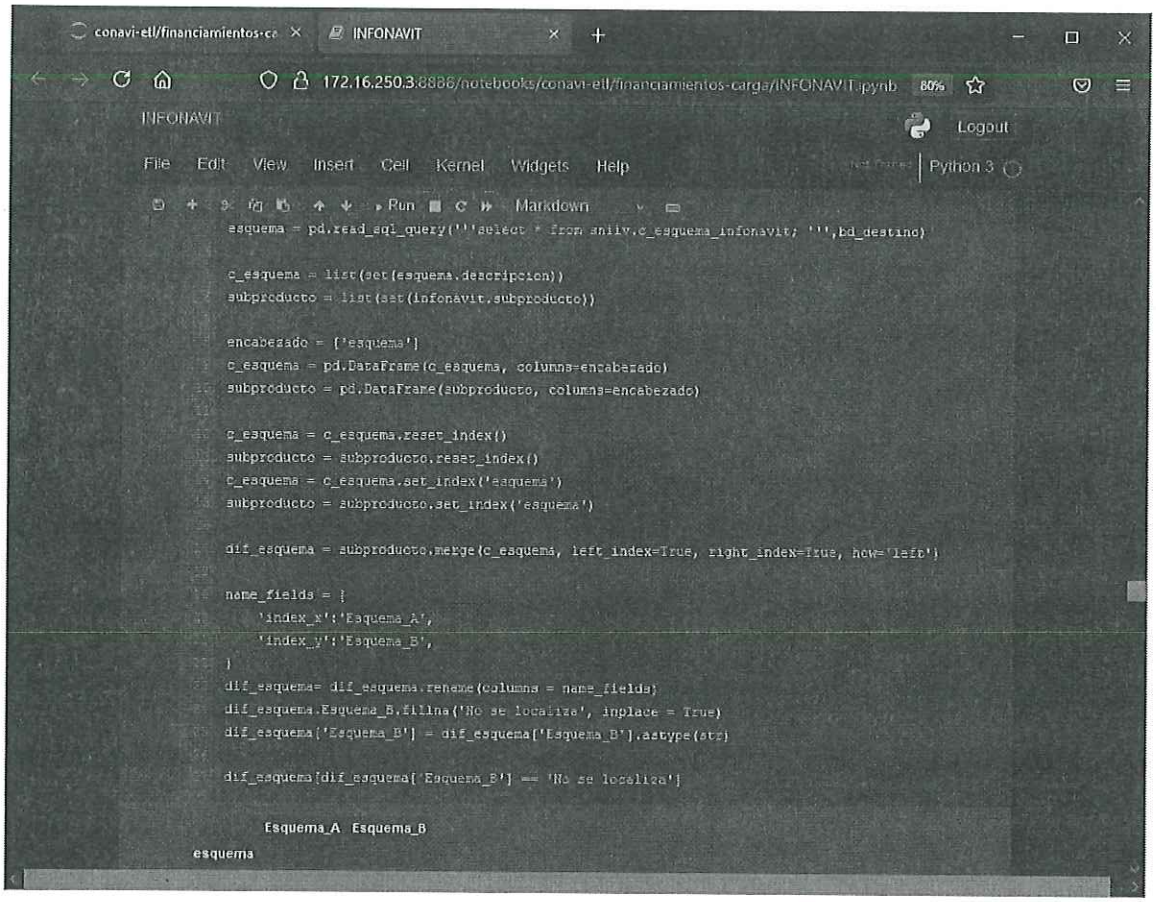


Imagen 2.8 Algoritmo para mostrar nuevos registros del catálogo c_esquema_infonavit de la carga INFONAVIT

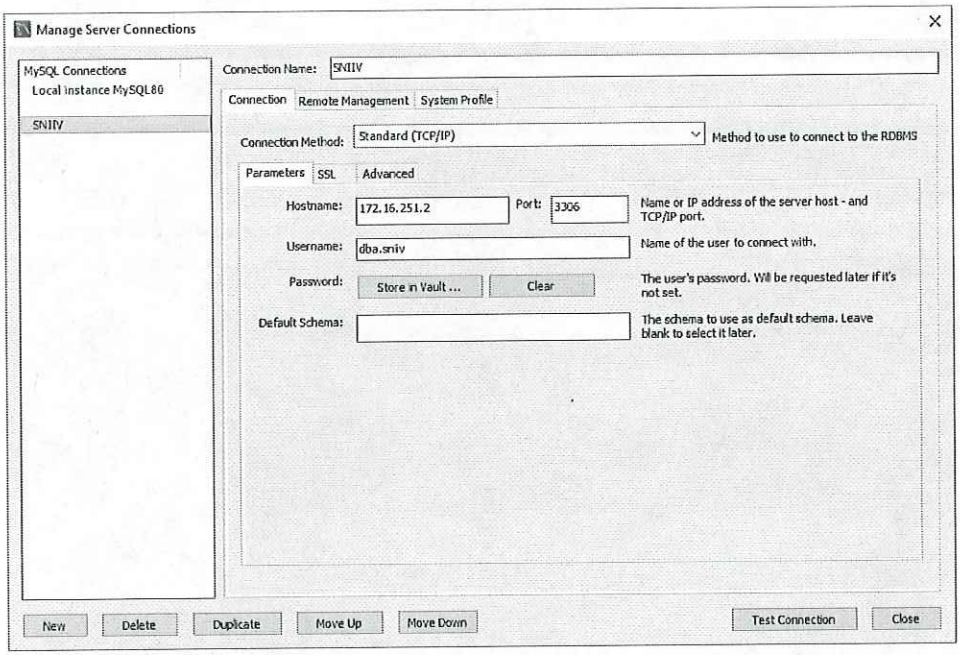


2. Posteriormente a la carga de información del proceso ETL, se deberán de ejecutar rutinas para la inserción de datos en tablas de consulta del SNIIV, la ejecución de rutinas se hará vía remota con las siguientes credenciales;

Usuario **root**: Tiene todos los privilegios.

Usuario **dba.sniiv**: Tiene privilegios sobre el esquema SNIIV

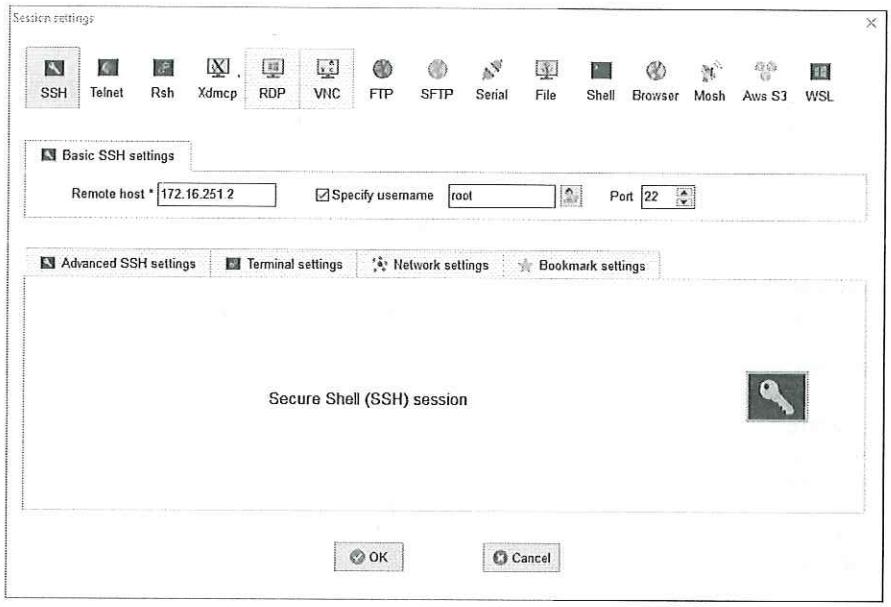
Imagen 2.9 Conexión remota vía aplicación MySQL Workbench



Notas: *Ver contraseñas de MySQL del archivo credenciales. Para saber el proceso de ejecución de rutinas a detalle, véase el documento "Manual de cargas".*

3. En caso de que se requiera acceder al directorio del servidor de base de datos deberá ser vía remota SSH.

Imagen 2.10 Conexión remota vía aplicación MobaXterm

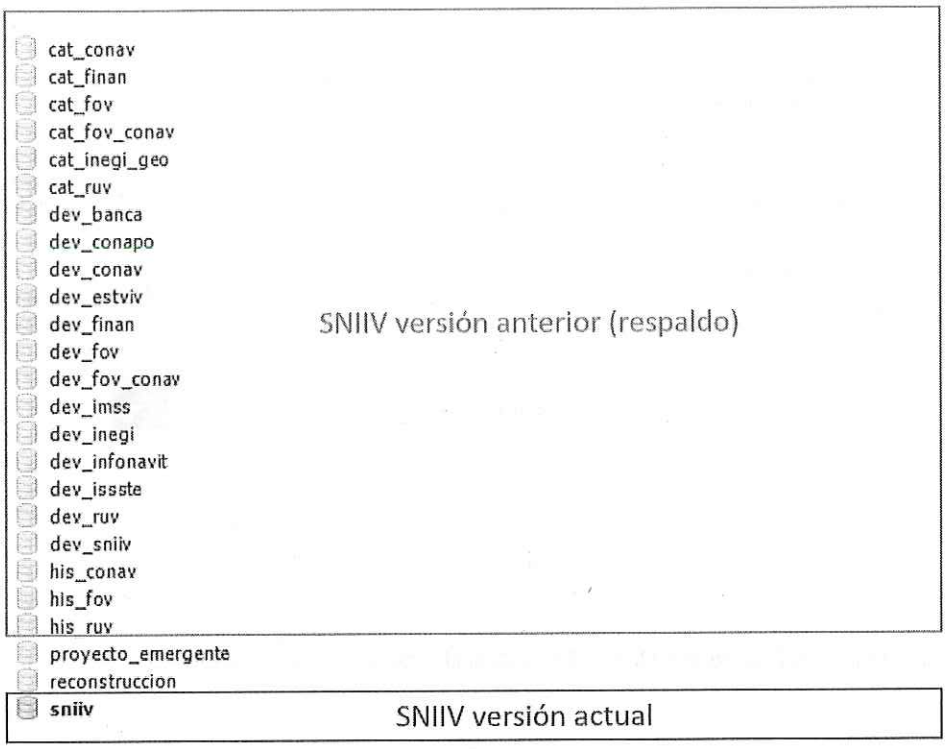


Notas: *Ver contraseña del Servidor Base de Datos del archivo credenciales.*

Esquema SNIIV

Este esquema contiene las tablas, procedimientos almacenados y funciones que conforman actualmente el SNIIV.

Imagen 2.11 Esquemas del servidor de Base de Datos



Tablas

Almacenan la información organizada y normalizada de los organismos de conforman el SNIIV. Algunas tablas utilizan nomenclatura; anteponer **c** para catálogos y **cubo** para datos multidimensionales, ejemplo:

Imagen 2.12 Tabla c_genero (catálogo)

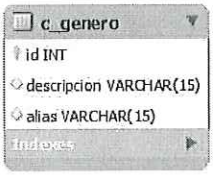


Imagen 2.13 Tabla conavi_semanal (modelo relacional)

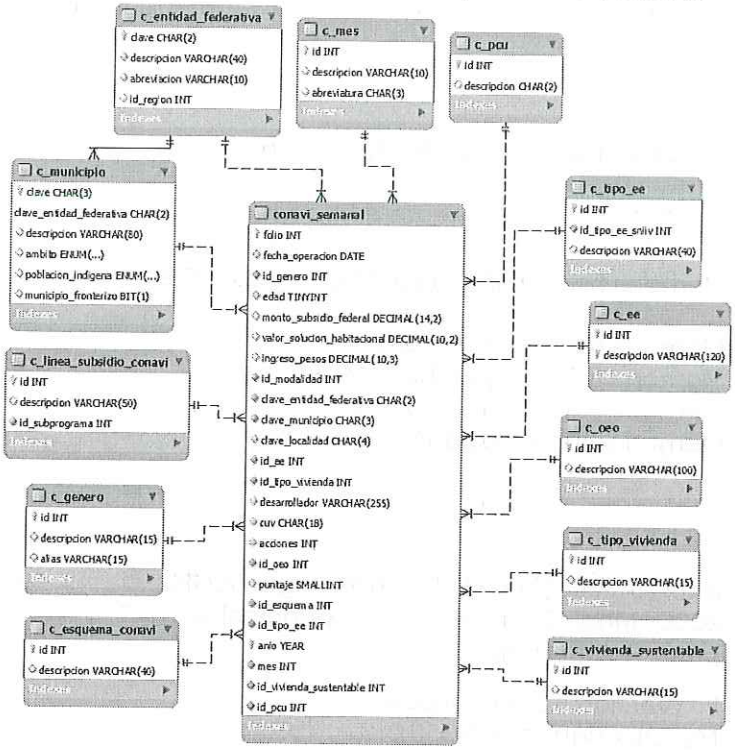


Imagen 2.14 Tabla cubo_conavi_bak (modelo multidimensional)

Una de las principales características del SNIIV es proveer de información de manera dinámica para que el usuario realice reportes conforme a sus necesidades, para realizarlo se implementa el modelo multidimensional, estos modelos ayudan a mejorar la eficiencia, velocidad y simplicidad de las consultas.

Un modelo multidimensional no puede sustituir un modelo relacional, detrás de cada modelo multidimensional deberá existir un único repositorio con la información normalizada; en este caso la tabla conavi_semanal (modelo relacional) almacena la información normalizada en base de datos y cubo_conavi_bak (modelo multidimensional) almacena información agrupada en base de datos con motor de almacenamiento MyISAM, que en cuanto a consultas se refiere ofrece una mayor velocidad sobre grandes cantidades de información.

Procedimientos almacenados y funciones

Para automatizar tareas en la base de datos del SNIIV se implementan procedimientos almacenados y funciones, la nomenclatura es; **sp** para procedimientos almacenados y **fn** para funciones.

Servidor Aplicativo y Datos espaciales

El servidor cuenta con un Sistema Operativo Microsoft Windows Server 2012 R2 Standard, con las siguientes especificaciones técnicas;

Tabla 2.3 Especificaciones técnicas del Servidor Aplicativo

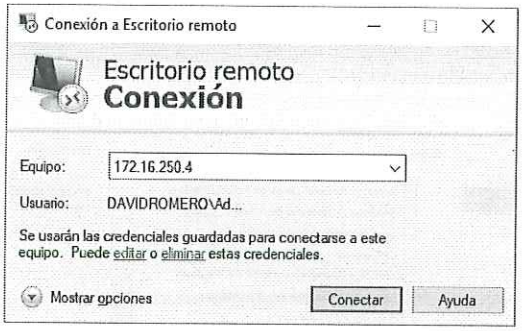
Almacenamiento	499 GB
Memoria RAM	Memoria física instalada (RAM) 32,0 GB Memoria física total 32,0 GB Memoria física disponible 27,3 GB Memoria virtual total 36,7 GB Memoria virtual disponible 31,4 GB
Procesador	Intel(R) Xeon(R) CPU E5-2620 0 @ 2.00GHz, 2000 Mhz, 4 procesadores principales, 4 procesadores lógicos
Programas instalados	IIS (Version 8.5.9500.16384) MySQL Connector 8.0.18.0 GlassFish Server Open Source Edition 4.1.1 GeoServer Version2.9.1 GeoTools Version15.1 GeoWebCache Version1.9.1
IP	172.16.250.4
Puertos	80 (HTTP), 443 (HTTPS), 8080 (GeoServer), 4848 (GlassFish)

IIS

IIS (Internet Information Server) es un servidor web extensible que provee un conjunto de servicios para sistemas operativos Windows. Una característica de este servicio permite publicar el sitio web del SNIIV. Para la gestión del sitio web, se deberá acceder de manera remota al servidor con las siguientes credenciales;

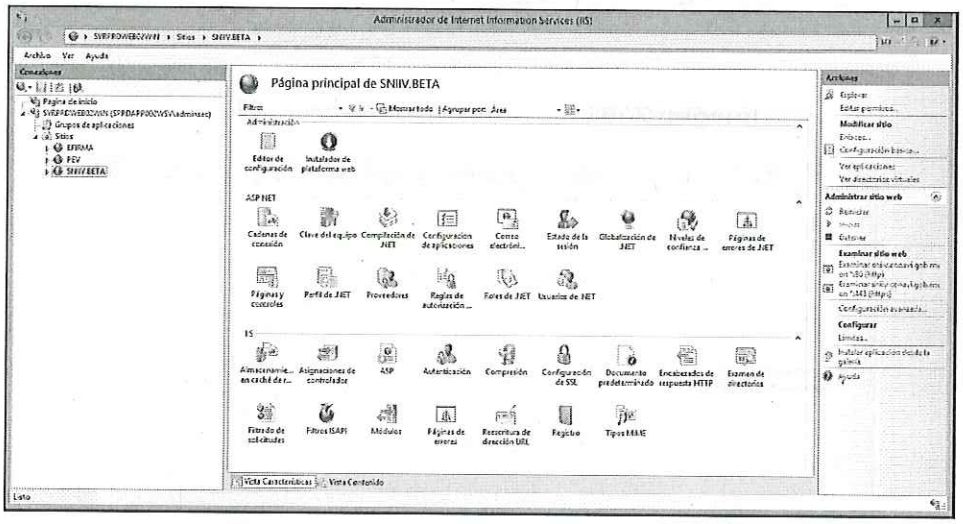
Nota: Ver contraseña del IIS del archivo credenciales.

Imagen 2.15 Conexión a escritorio remoto



IIS permite la gestión de sitios abriendo el Administrador de Internet Information Services para la configuración de Aplicaciones y Grupos de aplicaciones.

Imagen 2.16 Administrador de Internet Information Services



En el directorio C:\inetpub\wwwroot\sniiv se encuentra la estructura del proyecto web, es decir, todos los archivos que componen la Aplicación del SNIIV.

Configuración IIS

1. Servidor IIS y Servicios WCF

Para el correcto funcionamiento de la APP SNIIV, se requiere tener instalado el rol de servidor web IIS (Internet Information Services) y las características de Servicios WCF (Windows Communication Foundation) para la implementación de servicios web.

Imagen 2.17 Rol de servidor web IIS

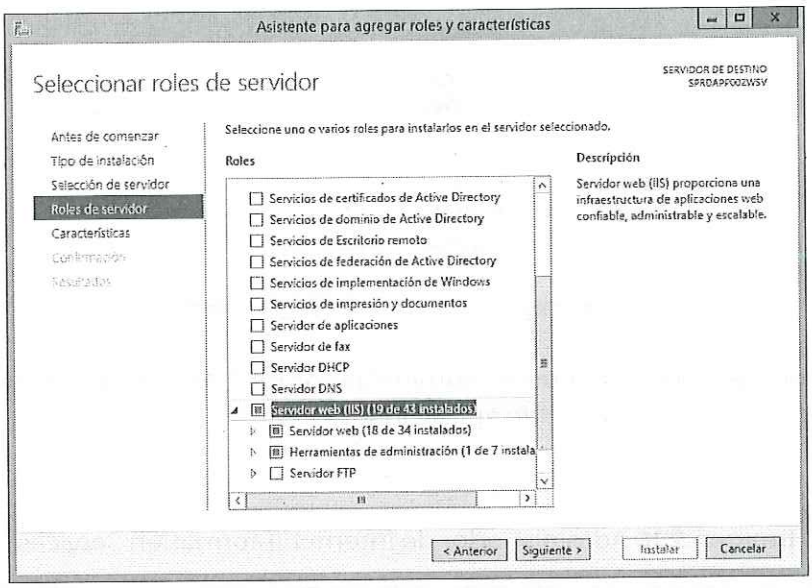
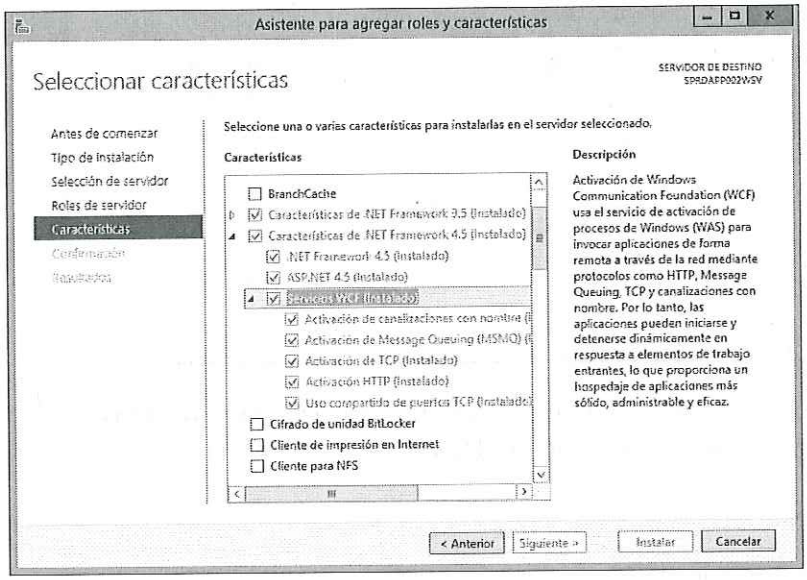


Imagen 2.18 Característica de Servicio WCF



2. Grupo de aplicaciones

Para organizar las aplicaciones web en el servidor, se recomienda crear grupos de aplicaciones para mantener una estructura y una organización. El Administrador de Internet Information Services permite gestionar las aplicaciones web de una manera gráfica.

Imagen 2.19 Administrador de Internet Information Services

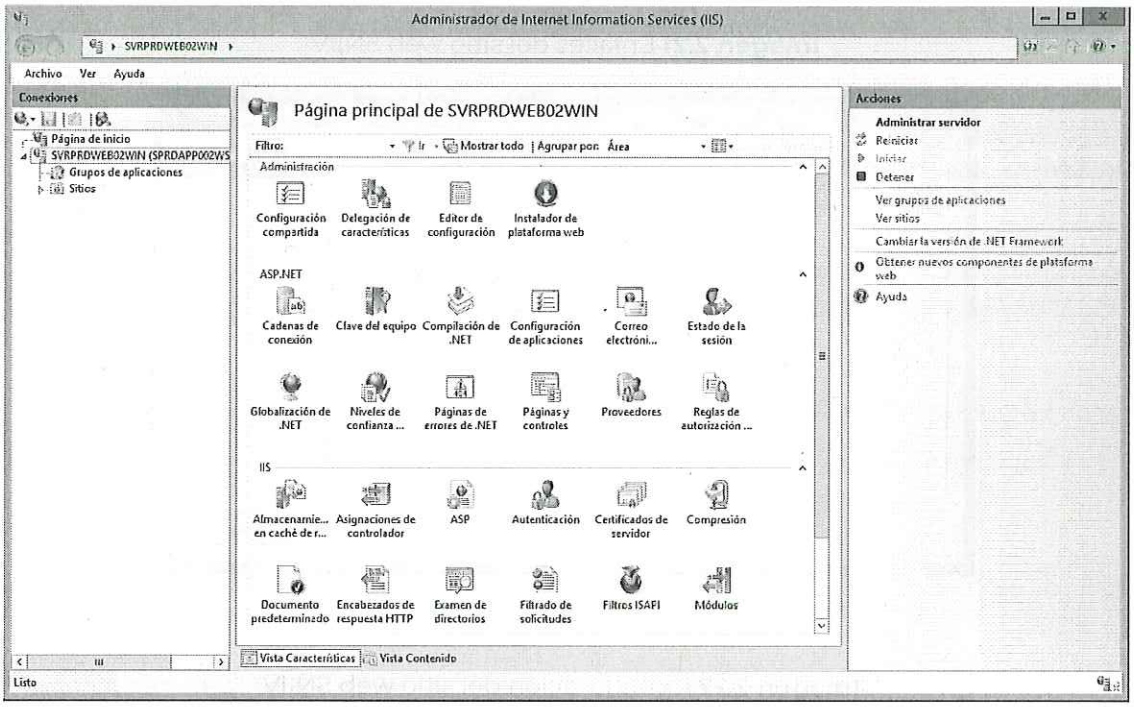


Imagen 2.20 Grupo de aplicaciones

3. Sitio web

Posteriormente a la creación de grupo de aplicaciones, se agregará un sitio web con los siguientes datos; nombre de sitio, grupo de aplicaciones, ruta de acceso física, tipo de enlace (http, https, etc.), dirección IP, puerto y nombre del host.

Imagen 2.21 Enlaces del sitio web SNIIV

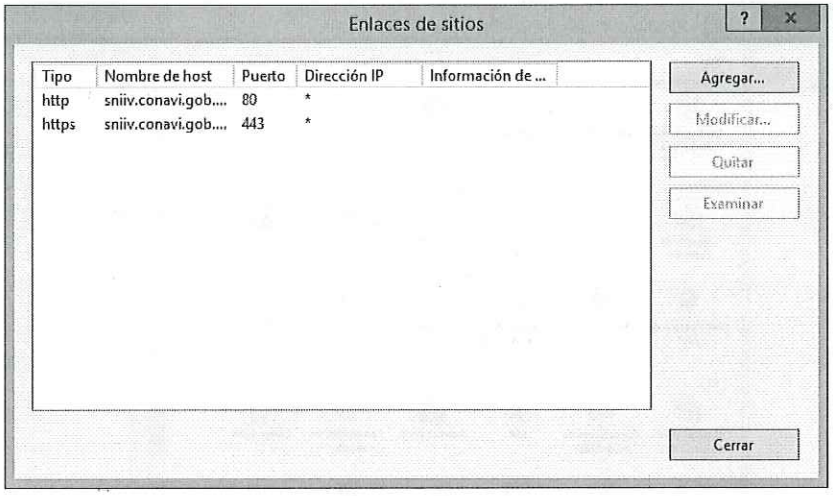
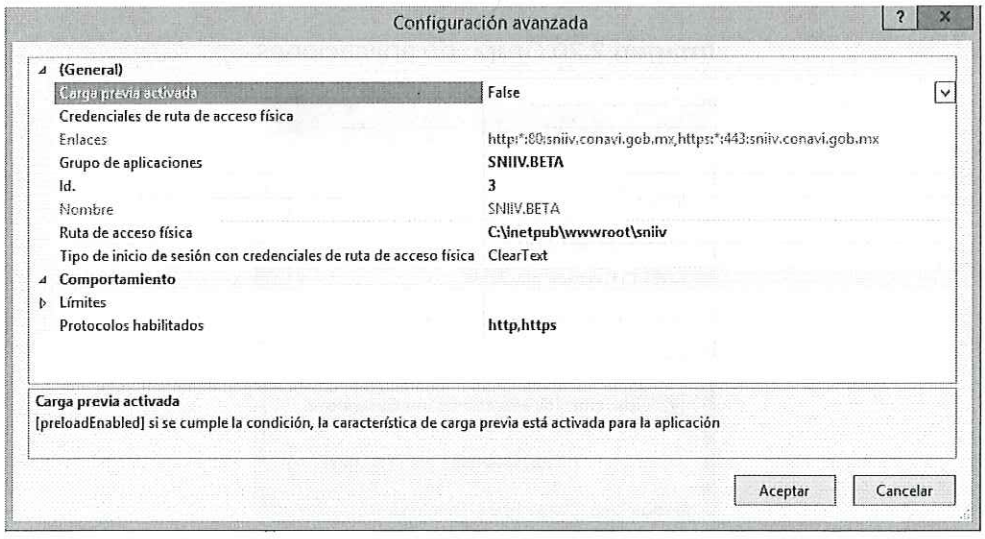


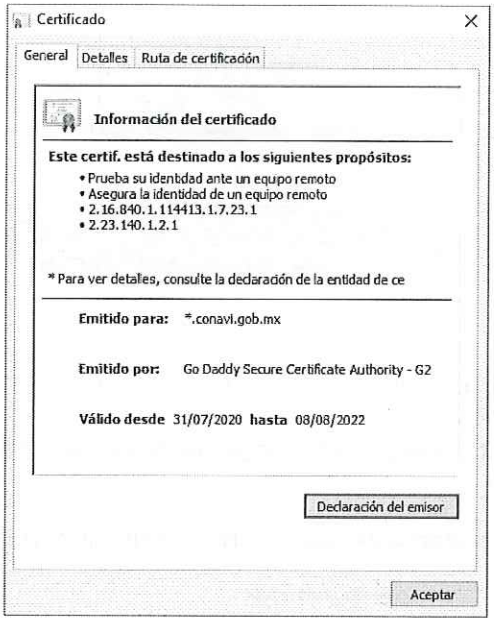
Imagen 2.22 Configuración del sitio web SNIIV



4. Certificado del servidor

Para crear un canal digital seguro en el sitio web del SNIIV, es indispensable agregar un certificado al servidor web en Administrador de Internet Information Services.

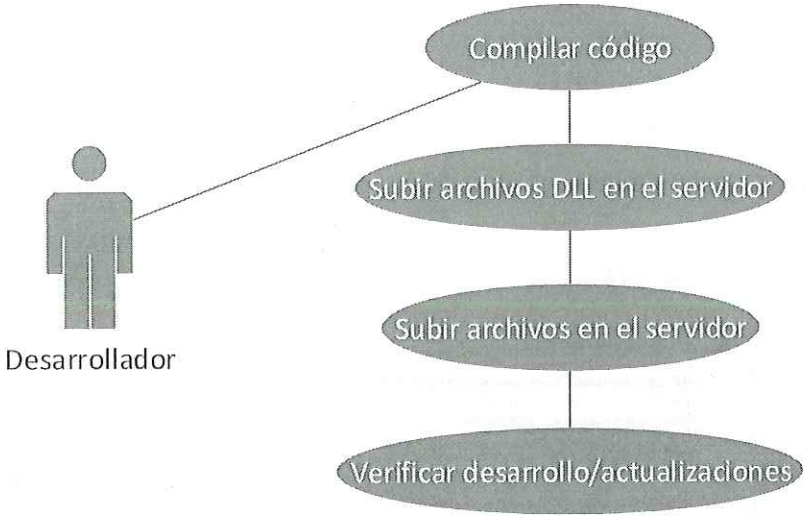
Imagen 2.23 Certificado de seguridad del sitio web SNIIV



Implementación IIS

1. Con la ejecución de rutinas en el servidor de Base de Datos, bastará para que la APP del SNIIV muestre de manera actualizada la información.

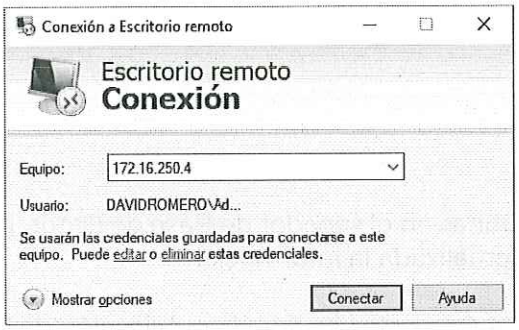
Diagrama 4 Casos de uso para la publicación de la APP del SNIIV en IIS



2. En caso de que se realicen actualizaciones y/o un nuevo desarrollo, se deberá tomar en cuenta las siguientes indicaciones;

2.1. Para la gestión del sitio web, el rol de administrador se deberá acceder de manera remota al servidor.

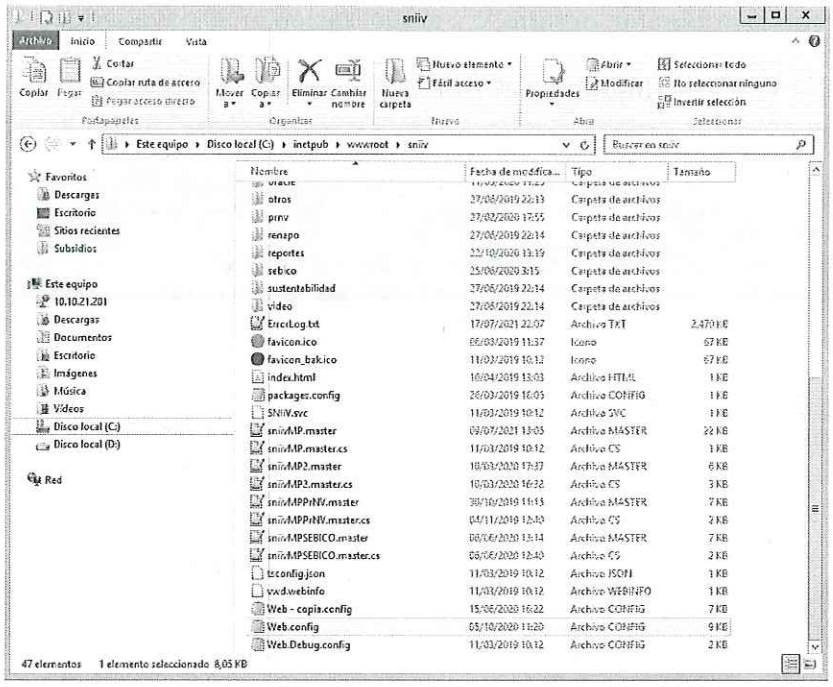
Imagen 2.24 Conexión a escritorio remoto



Nota: *Ver contraseña del IIS del archivo credenciales.*

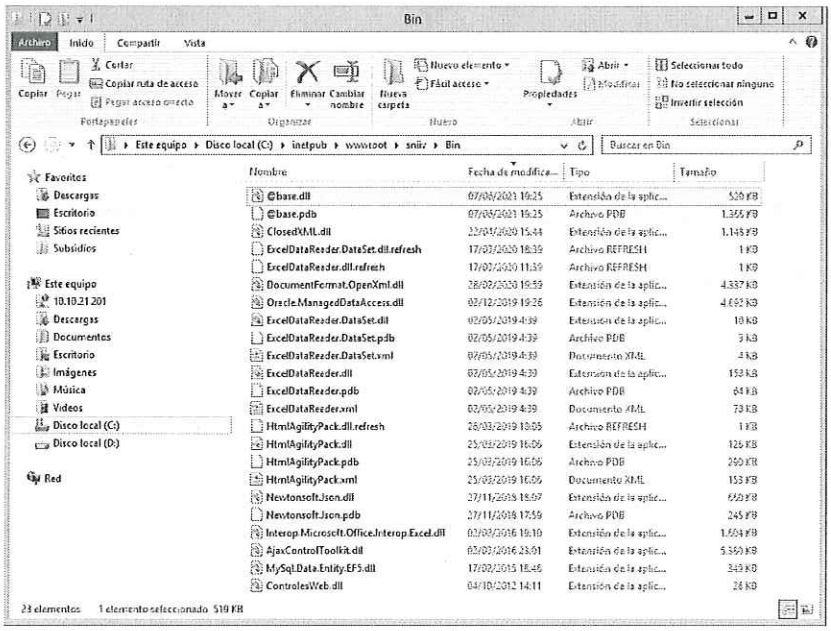
2.2. En el directorio **C:\inetpub\wwwroot\sniiv** se encuentra la estructura del proyecto web, es decir, todos los archivos que componen la Aplicación del SNIIV. Para publicar los cambios es indispensable reemplazar los archivos nuevos o actualizados.

Imagen 2.25 Directorio de la APP SNIIV



2.3. En caso de que se haya actualizado alguna clase, será necesario reemplazar la librería **base.dll** ubicado en la carpeta **Bin** del sitio web SNIIV.

Imagen 2.26 Directorio Bin de la APP SNIIV



2.4. Finalmente corroborar el desarrollo o actualización en el navegador.

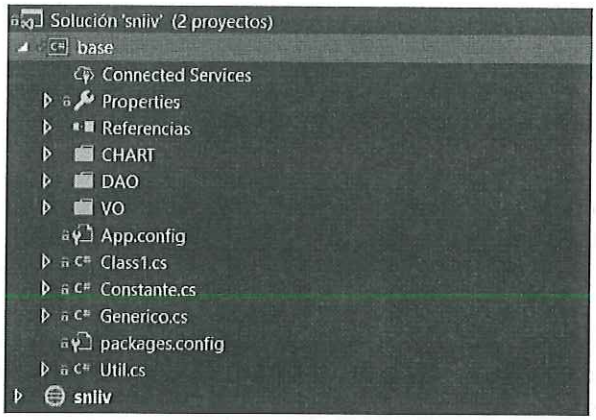
Imagen 2.27 APP SNIIV desde el navegador

Estructura del proyecto web

El proyecto SNIIV está separado en dos partes; La Biblioteca de clases **base** y el Sitio web **SNIIV**.

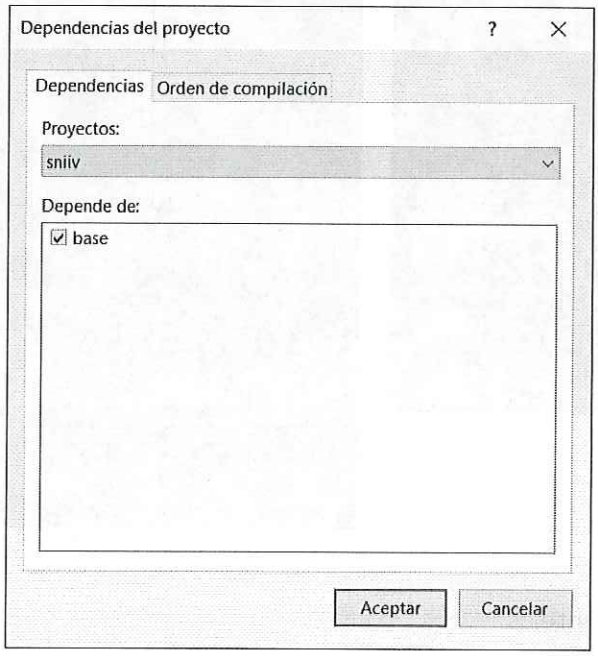
Biblioteca de clases (base)

Imagen 2.28 Directorio base



La biblioteca de clases **base** contiene todas las clases del proyecto SNIIV, con el fin de utilizarlas como librerías (Dynamic Link Library) previamente compiladas que constan de código ejecutable. Al compilar el proyecto SNIIV se sobrescribirá un archivo DLL en el sitio web sniiv (Bin/@base.dll) ya que depende directamente de base.

Imagen 2.29 Dependencias del sitio web SNIIV



Nota: *@base.dll al ser un archivo de código ejecutable, no será necesario cargar en el servidor todo el directorio base cada vez que se realice una actualización, bastará con reemplazar el archivo @base.dll en la carpeta \Bin del directorio del servidor IIS.*

A continuación, se dará explicación de los archivos y directorios en **base;**

- Referencias

Hace referencias a otras librerías que el proyecto requiere, por ejemplo; MySql.Data.dll y MySql.Web.dll, conector que permite trabajar con base de datos relacionales desde nuestro lenguaje de programación C#.

- VO y DAO

  Para un código organizado, legible y mantenible, además de permitir reutilizar código y aumentar la escalabilidad del proyecto SNIIV se emplea el patrón de diseño VO que va de la mano con el patrón de diseño DAO.

  VO contiene las clases Value Object, clases que únicamente contienen sus atributos; constructor, getters y setters, estas clases no tiene comportamiento. Por otra parte, DAO contiene las clases Data Access Object, clase conocida como clase de negocio, se encarga de obtener datos desde la base de datos y llenar la clase Value Object para enviarla al cliente, o a su vez recibir la clase desde el cliente y enviar datos al servidor, por lo general tiene los comandos SQL que se ejecutaran, ejemplo;

Imagen 2.30 Clase ConaviVO

```
public class ConaviVO : CuboVO
{
    [DataMember(EmitDefaultValue = false)]
    public string entidad_ejecutora { get; set; }
    [DataMember(EmitDefaultValue = false)]
    public string tipo_entidad_ejecutora { get; set; }
    [DataMember(EmitDefaultValue = false)]
    public string organismo_ejecutor_obra { get; set; }
    [DataMember(EmitDefaultValue = false)]
    public string tipo_vivienda { get; set; }
    [DataMember(EmitDefaultValue = false)]
    public string desarrollo_certificado { get; set; }
    [DataMember(EmitDefaultValue = false)]
    public string vivienda_sustentable { get; set; }
    [DataMember(EmitDefaultValue = false)]
    public string pcu { get; set; }

    public ConaviVO()...
}
```

Imagen 2.31 Clase ConaviDAO

```
public class ConaviDAO
{
    private static ConaviDAO _instancia = null;

    public static ConaviDAO instancia()
    {
        return _instancia == null ? new ConaviDAO() : _instancia;
    }

    public ConaviDAO()...

    #DASHBOARD

    #CUBO

    #CATALOGO

    #region METAS

    public DataTable seleccionarMeta(int anio)
    {
        string str = "call sp_get_meta_conavi(" + anio + ")";
        DataTable dt = new DataTable();

        try { dt = Generico.instancia().seleccionar(str, Constante.BD_SNIIV); }
        catch (Exception ex) { Util.instancia().setLogError(ex); }
        return dt;
    }

    public List<CatalogoVO> seleccionarAnioMeta()...

    #endregion
}
```

- Generico.cs, Util.cs y Constante.cs

  Son clases de uso general, Generico.cs cuenta con los métodos CRUD (create, read, update y delete) de forma genérica, es decir, es posible que cualquier clase instancie cualquiera de sus métodos para ejecutar algún comando SQL, sin la necesidad de capturar repetidamente el proceso de conexión.

  Util.cs contiene métodos que generalmente se reutilizan en el proyecto.

Finalmente, la clase Constante.cs contiene los valores permanentes que se ocuparan para el proyecto, por ejemplo; el nombre de la cadena de conexión.

Sitio web SNIIV

Imagen 2.32 Directorio SNIIV

El sitio web sniiv contiene todas páginas de ASP.NET, conocidas oficialmente como "web forms" (formularios web), son el principal medio de construcción para el desarrollo de aplicaciones web del SNIIV, estos archivos típicamente contienen etiquetas HTML o XHTML estático, y también etiquetas definiendo Controles Web que se procesan del lado del servidor y Controles de Usuario donde los desarrolladores colocan todo el código estático y dinámico requerido por la página web. También contiene todos los archivos que se ocuparan en el lado del cliente; CSS (Cascading Style Sheets), código ASP (Active Server Pages), librerías JavaScript, documentos, web services, imágenes, etc.

A continuación, se dará una breve explicación de los archivos y directorios en **SNIIV;**

- App_Code

  Es un directorio para códigos. El servidor ASP.NET automáticamente compilará los archivos (y subdirectorios) en esta carpeta en un ensamblado que es accesible desde cualquier página del sitio. App_Code es típicamente usada para código de acceso a datos, código de modelo o código de negocios. También cualquier manejador http específico para el sitio e implementación de módulos y

servicios web van este directorio. Este directorio cuenta con la clase SNIIV.cs que hereda de la interface ISNIIV.cs, archivos que nos permite implementar una arquitectura orientada a servicios.

- Bin

  Contiene código compilado (archivos .dll) para controles, componentes, y otro código que pueda ser referenciado por el sitio web sniiv. Cualquier clase representada por código en la carpeta Bin es automáticamente referenciada en la aplicación. Son archivos o librerías que tienen como principal acción ejecutar una función cuando estas son llamadas o se invocan. En este directorio se encuentra la librería @base.dll (biblioteca de clases ya mencionada anteriormente), además de librerías que complementan el sitio web sniiv; AJAX Control Toolkit, Json.NET, Html Agility Pack, ExcelDataReader, etc.

- js

  Todos los archivos que contienen código JavaScript que se ejecutan en el lado del cliente se alojan en este directorio, el SNIIV además de implementar código dinámico en lenguaje JavaScript, también implementa una serie de librerías open source que permiten visualizar mapas, gráficos, dashboars, tablas, controles, etc.

  Algunas librerías que implementa el SNIIV son; jQuery, linq.js (LINQ for JavaScript), Apache ECharts, D3.js (Data-Driven Documents), dc.js (Dimensional Charting Javascript Library), DataTables, PivotTable.js, OpenLayers, jQuery Geocoding, etc.

- css

  Aloja todos los archivos Hojas de Estilo en Cascada; estilos exclusivos del sitio web sniiv, estilos implementados por algunas librerías JavaScript y estilos del framework Bootstrap.

- fonts, img, doc

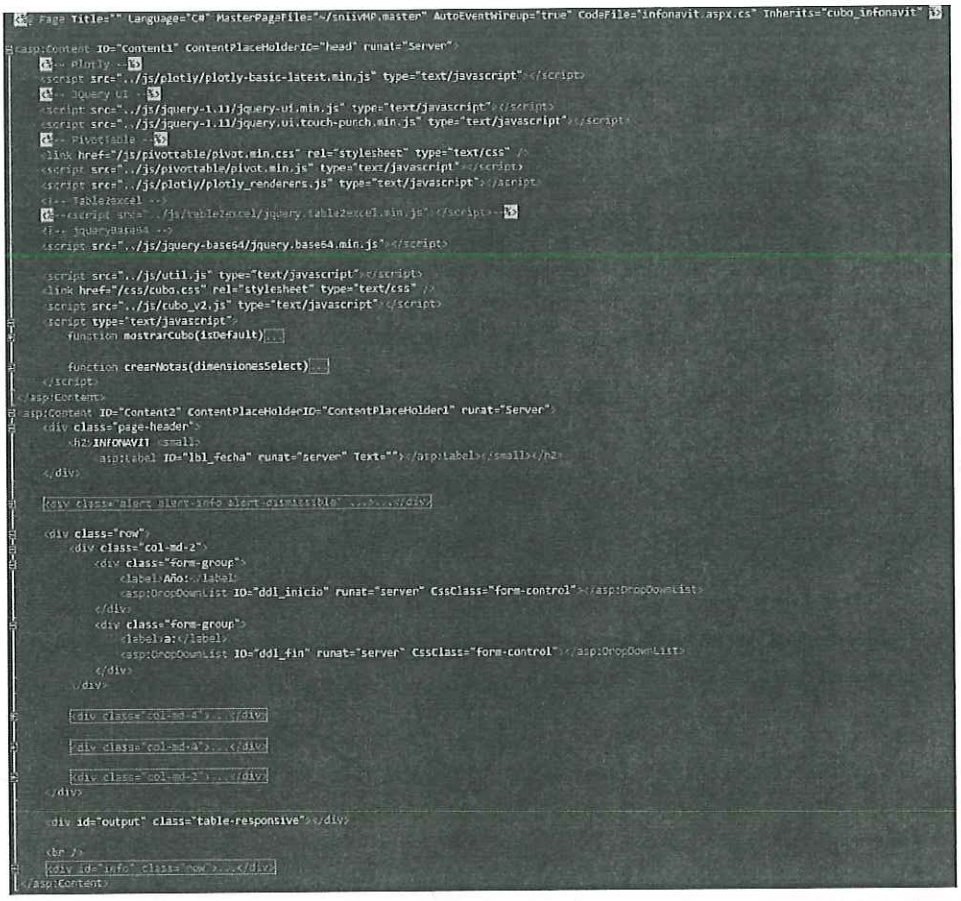
  Contiene tipos de fuente, imágenes en varios formatos y archivos que publica el SNIIV; reportes PDF, reportes CSV, capas geográficas, etc.

- oferta, demanda y demás carpetas

  En estos directorios están todos los formularios Web Forms que componen los módulos del SNIIV. Con los formularios Web Forms ASP.NET es posible crear sitios web dinámicos con cientos de controles y componentes que permitirán crear rápidamente sitios potentes y sofisticados.

  Para mantener independientes la interfaz de usuario y la lógica asociada a la aplicación, la implementación de las páginas ASP.NET se divide en dos archivos. En un archivo con extensión **.aspx** se especifica el aspecto de la interfaz gráfica, utilizando tanto etiquetas HTML estándar como etiquetas específicas para hacer referencia a los controles ASP.NET que se deseen incluir en la página web. En un segundo archivo, que será un archivo de código con extensión **.cs** se utiliza lenguaje C#, en este archivo se implementa la lógica de la aplicación, ejemplo;

Imagen 2.33 Archivo cubo/Infonavit.aspx



```
<%@ Page Title="" Language="C#" MasterPageFile="~/sniivMP.master" AutoEventWireup="true" CodeFile="infonavit.aspx.cs" Inherits="cubo_infonavit" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="Server">
    <!-- Plotly -->
    <script src="../js/plotly/plotly-basic-latest.min.js" type="text/javascript"></script>
    <!-- JQuery UI -->
    <script src="../js/jquery-1.11/jquery-ui.min.js" type="text/javascript"></script>
    <script src="../js/jquery-1.11/jquery.ui.touch-punch.min.js" type="text/javascript"></script>
    <!-- Pivotable -->
    <link href="/js/pivottable/pivot.min.css" rel="stylesheet" type="text/css" />
    <script src="../js/pivottable/pivot.min.js" type="text/javascript"></script>
    <script src="../js/plotly/plotly_renderers.js" type="text/javascript"></script>
    <!-- Table2excel -->
    <!--<script src="../js/table2excel/jquery.table2excel.min.js"></script>-->
    <!-- jqueryBase64 -->
    <script src="../js/jquery-base64/jquery.base64.min.js"></script>

    <script src="../js/util.js" type="text/javascript"></script>
    <link href="/css/cubo.css" rel="stylesheet" type="text/css" />
    <script src="../js/cubo_v2.js" type="text/javascript"></script>
    <script type="text/javascript">
        function mostrarCubo(isDefault)...

        function crearNotas(dimensionesSelect)...
    </script>
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="Server">
    <div class="page-header">
        <h2>INFONAVIT <small>
            <asp:Label ID="lbl_fecha" runat="server" Text=""></asp:Label></small></h2>
    </div>

    <div class="alert alert-info alert-dismissible" ...>...</div>

    <div class="row">
        <div class="col-md-2">
            <div class="form-group">
                <label>Año:</label>
                <asp:DropDownList ID="ddl_inicio" runat="server" CssClass="form-control"></asp:DropDownList>
            </div>
            <div class="form-group">
                <label>a:</label>
                <asp:DropDownList ID="ddl_fin" runat="server" CssClass="form-control"></asp:DropDownList>
            </div>
        </div>

        <div class="col-md-4">...</div>

        <div class="col-md-4">...</div>

        <div class="col-md-2">...</div>
    </div>

    <div id="output" class="table-responsive"></div>

    <br />
    <div id="info" class="row">...</div>
</asp:Content>
```

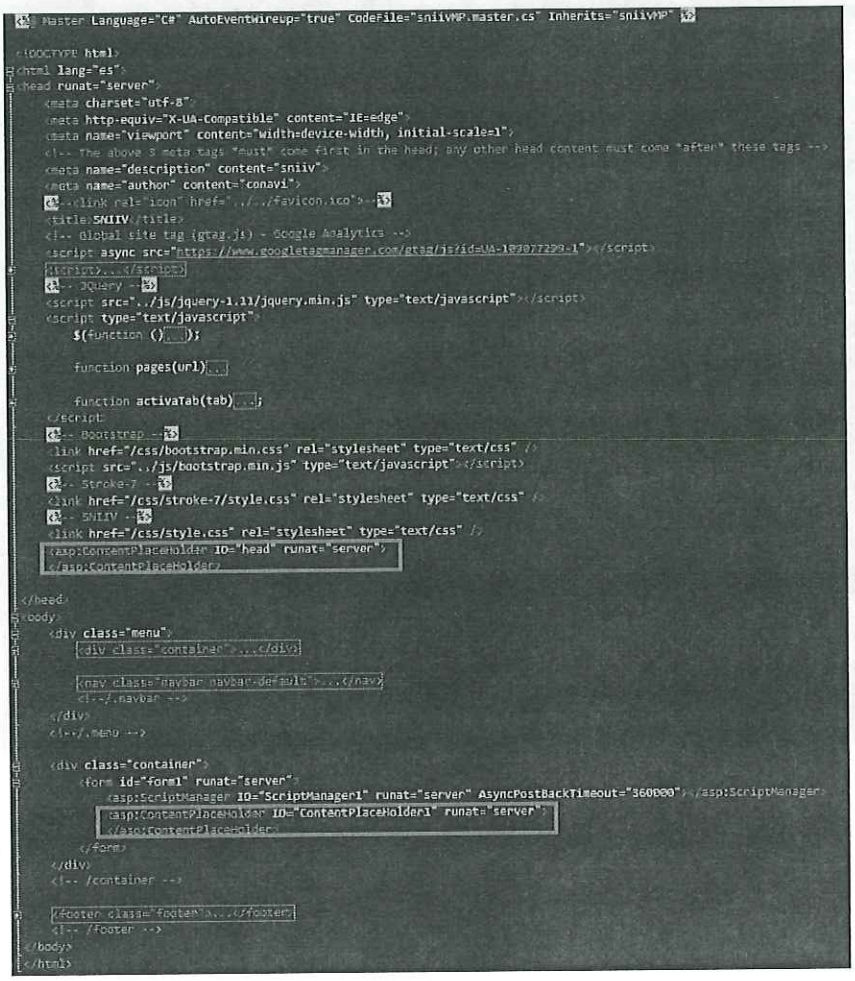
Imagen 2.34 Archivo cubo/Infonavit.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class cubo_infonavit : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            mostrarFecha();
            cargarListas();
        }
    }

    protected void mostrarFecha()
    {
        lbl_fecha.Text = Util.instancia().getLeyendaFecha(InfonavitDAO.instancia().seleccionarFecha());
    }

    protected void cargarListas()
    {
        List<CatalogoVO> anios = InfonavitDAO.instancia().seleccionarAnio();
        ddl_inicio.DataValueField = "id";
        ddl_inicio.DataTextField = "descripcion";
        ddl_inicio.DataSource = anios;
        ddl_inicio.DataBind();

        ddl_fin.DataValueField = "id";
        ddl_fin.DataTextField = "descripcion";
        ddl_fin.DataSource = anios;
        ddl_fin.DataBind();

        List<CatalogoVO> estados = CatalogoDAO.instancia().seleccionarEntidadFederativa();
        ddl_estado.DataValueField = "id";
        ddl_estado.DataTextField = "descripcion";
        ddl_estado.DataSource = estados;
        ddl_estado.DataBind();
        ddl_estado.Items.Insert(0, new ListItem("...", "0"));
        ddl_estado.Items.Insert(1, new ListItem("Todos los estados", Constante.FORMATO_ESTATAL));
    }

    protected void ddl_estado_SelectedIndexChanged(object sender, EventArgs e)...
}
```

- sniivMP.master, sniivMP2.master y sniivMPPrNV.master

Cuando se crea un sitio web se tiene la necesidad de repetir ciertas partes de una página en todo el sitio o en parte del sitio. Para no estar copiando y pegando las mismas estructuras en todas las páginas hay la posibilidad de crear una Master Page y referenciarla en las otras páginas. Para poder cargar cualquier objeto web con el marco especificado por la Master Page, se utiliza un control llamado **Content Placeholder.** Este control define en qué lugar, dentro de la Master Page, estará ubicado el contenido de las Transacciones Web y los Web Panels que la utilizarán como su página maestra, ejemplo;

Imagen 2.35 Archivo sniivMP.master



El SNIIV implementa tres tipos de Master Page; sniivMP.master, Master page para cualquier pagina general. La Master Page sniivMP2.master se utiliza como plantilla únicamente para páginas de la intranet en el SNIIV, ya que requiere de una estructura especifica (los módulos se crean de manera dinámica dependiendo usuario y rol). Y sniivMPPrNV.master se implementa específicamente para el sistema PrNV (Premio Nacional de Vivienda). Ejemplo;

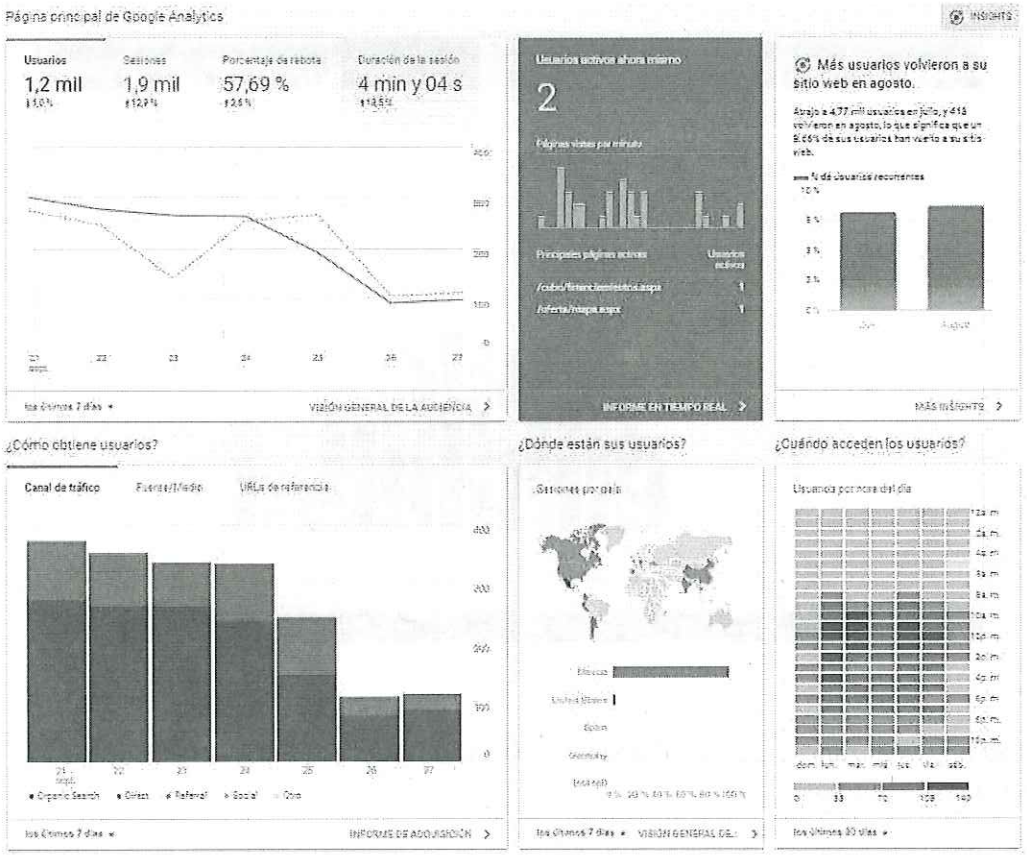
Imagen 2.36 sniivMP.master

Para monitorear y analizar el sitio web del SNIIV, se utiliza la herramienta Google Analytics incorporada en la Master page principal.

Google Analytics

Es una herramienta de analítica web muy completa que facilita información básica como: número de visitantes y de visitas en el SNIIV, duración media de la visita, la media de páginas vistas por cada usuario, informes geográficos, sociodemográficos (lenguaje, ubicación, proveedor de Internet, dispositivo móvil), etc. Pero también proporciona información más compleja como reportes por periodos, comportamientos, además de muchas otras variables.

Imagen 2.37 Página de Google Analytics del SNIIV



Nota: *Para la implementación de la herramienta de Google Analytics, se requiere de una cuenta Gmail y la configuración correspondiente en los archivos con extensión "master".*

WCF

WCF (Windows Communication Foundation) es un modelo de programación para el desarrollo de aplicaciones con arquitectura orientada a servicios (SOA). Aplicaciones distribuidas basadas en la comunicación mediante mensajes. El SNIIV cuenta con web services para la compartición de datos en formato JSON, esto mediante WCF. Ejemplo de Aplicación de servicios WCF;

Imagen 2.38 Clase InfonavitVO.cs

```
using System.Runtime.Serialization;

/// <summary> Descripción breve de InfonavitVO
[DataContract]
public class InfonavitVO : CuboVO
{
    [DataMember(EmitDefaultValue = false)]
    public string estado_civil { get; set; }
    [DataMember(EmitDefaultValue = false)]
    public string linea_credito { get; set; }
    [DataMember(EmitDefaultValue = false)]
    public string tipo_credito { get; set; }
    [DataMember(EmitDefaultValue = false)]
    public string intermediario_financiero { get; set; }
    [DataMember(EmitDefaultValue = false)]
    public string vivienda_sustentable { get; set; }

    public InfonavitVO()...
}
```

[DataContract]: Es un acuerdo formal entre un servicio y un cliente que describe de manera abstracta los datos que se intercambiarán.

[DataMember]: Cuando se aplica al miembro de un tipo, especifica que el miembro forma parte de un DataContract y es serializable por el DataContractSerializer.

Imagen 2.39 Clase InfonavitDAO.cs

```
public List<InfonavitVO> getCubo(string anios, string clave_estado, string clave_municipio, string dimensiones)
{
    string anio_inicio = anios.Split(',').First();
    string anio_fin = anios.Split(',').Last();

    string[] lstDimensiones = dimensiones.Split(',');
    string[] lst = new string[3];

    StringBuilder field = new StringBuilder();
    StringBuilder subField = new StringBuilder();
    StringBuilder table = new StringBuilder();
    foreach (string dimension in lstDimensiones)
    {
        lst = crearConsulta(dimension);
        field.Append(lst[0]);
        subField.Append(lst[1]);
        table.Append(lst[2]);
    }
    string strField = limpiarConsulta(field.ToString(), ",");
    string strSubField = limpiarConsulta(subField.ToString(), ",");
    string strTable = limpiarConsulta(table.ToString(), " ");

    List<InfonavitVO> cubo = new List<InfonavitVO>();
    StringBuilder query = new StringBuilder();
    query.Append("select ");
    query.Append(strField);
    query.Append(",acciones,monto");
    query.Append(" from (select ");
    query.Append(strSubField);
    query.Append(",sum(acciones) as acciones,sum(monto) as monto");
    query.Append(" from cubo_infonavit_bak ");
    if (anio_inicio.Equals(anio_fin))
        query.Append("where anio = " + anio_inicio);
    else
        query.Append("where anio between " + anio_inicio + " and " + anio_fin);
    if (isEstatal(clave_estado))
        query.Append(" and clave_estado = '" + clave_estado + "'");
    if (isMunicipal(clave_municipio))
        query.Append(" and clave_municipio = '" + clave_municipio + "'");
    query.Append(" group by ");
    query.Append(strSubField);
    query.Append(") f ");
    query.Append(strTable);
    try
    {
        DataTable dt = Generico.instancia().seleccionar(query.ToString(), Constante.BD_SNIIV);
        cubo = Util.instancia().ConvertDataTable<InfonavitVO>(dt);
    }
    catch (Exception ex) { Util.instancia().setLogError(ex); }
    return cubo;
}
```

En la clase InfonavitDAO.cs se realizará la consulta (consulta dinámica) a la base de datos (modelo multidimensional) instanciando la clase Generico.cs, al obtener los datos se almacenará en una lista de objetos tipo InfonavitVO para ser retornada.

Imagen 2.40 Clase SNIIV.cs

```
public class SNIIV : ISNIIV
{
    //INFONAVIT
    [WebInvoke(Method = "GET", UriTemplate = "getINFONAVIT/{anios}/{clave_estado}/{clave_municipio}/{dimensiones}", ResponseFormat = WebMessageFormat.Json)]
    public List<InfonavitVO> getINFONAVIT(string anios, string clave_estado, string clave_municipio, string dimensiones)
    {
        return InfonavitDAO.instancia().getCubo(anios, clave_estado, clave_municipio, dimensiones);
    }
```

Imagen 2.41 interface ISNIIV.cs

```
[ServiceContract]
public interface ISNIIV
{
    //INFONAVIT
    [OperationContract]
    List<InfonavitVO> getINFONAVIT(string anios, string clave_estado, string clave_municipio, string dimensiones);
```

Como se mencionó anteriormente el directorio App_Code cuenta con dos archivos; SNIIV.cs y ISNIIV.cs.

SNIIV.cs contiene la implementación predeterminada del contrato de servicio, contendrá todos los métodos que contendrá el web service, métodos que serán instancia de las clases DAO, InfonavitDAO.cs en este caso, mientras que ISNIIV.cs es la interface contiene la definición predeterminada del contrato de servicio.

[ServiceContract]: Indica que una interfaz o una clase define un contrato de servicio en una aplicación WCF.

[OperationContract]: Indica que un método define una operación que forma parte de un contrato de servicio en una aplicación WCF.

[WebInvoke]: Se aplica a una operación de servicio además de OperationContract y asocia la operación con un UriTemplate así como un verbo de transporte subyacente que representa una invocación (por ejemplo, HTTP POST, PUT o DELETE).

Y finalmente SNIIV.svc que contiene una directiva de procesamiento específica de WCF (@ ServiceHost) que permite a la infraestructura de hospedaje de WCF activar servicios hospedados en respuesta a los mensajes entrantes. Ejemplo;

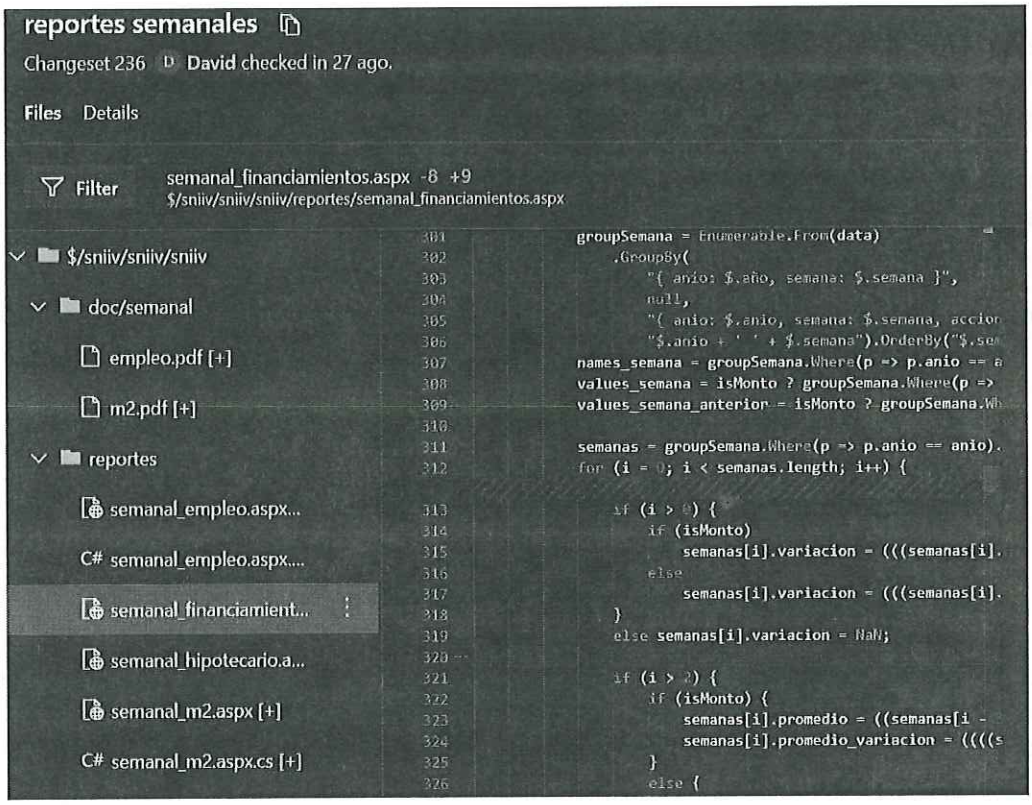
Imagen 2.42 Respuesta del método getINFONAVIT en el navegador (formato JSON)

Finalmente, se recomienda la implementación de herramientas para la gestión del proyecto, control de código fuente, repositorios y control de versiones principalmente, el SNIIV actualmente está desarrollado en tecnología .NET, por consiguiente, la implementación de TFS (Team Foundation Server) será de gran ayuda.

TFS

Team Foundation Server es un producto de Microsoft que proporciona administración de código fuente (ya sea a través de Team Foundation Version Control o Git), informes, administración de requisitos, gestión de proyectos (para equipos de desarrollo de software ágil y en cascada), compilaciones automatizadas, capacidades de gestión de laboratorio, pruebas y gestión de lanzamientos. Cubre todo el ciclo de vida de la aplicación. TFS se puede usar como un back-end para numerosos entornos de desarrollo integrado, pero está diseñado para Microsoft Visual Studio y Eclipse.

Imagen 2.43 Pantalla de cambios de Team Foundation Version Control del SNIIV



Nota: *Para la implementación de la herramienta TFS se requiere de cuentas Microsoft por parte del equipo de desarrollo del SNIIV para su gestión y seguimiento.*

La explicación de la estructura del proyecto web se concluye con el Servidor Aplicativo, por otra parte, también se cuenta con GeoServer, un servidor para gestionar datos espaciales.

GeoServer

Para la publicación de capas geográficas dentro del SNIIV se publican WMS. Un servicio web de mapas o Web Map Service (WMS) es un protocolo estándar definido por el OGC que sirve imágenes de mapas a partir de información geográfica.

GeoServer es un servidor de datos espaciales que permite la publicación de los WMS del SNIIV. Es una aplicación web de código abierto, escrito en Java, que permite a los usuarios compartir y editar datos geoespaciales.

Para el correcto funcionamiento de la aplicación GeoServer se requiere de un servidor de aplicaciones. GlassFish es un servidor de aplicaciones para tecnología Java, es gratuito, de código libre y se distribuye bajo un licenciamiento dual a través de la licencia CDDL y la GNU GPL.

Imagen 2.44 GlassFish en el navegador

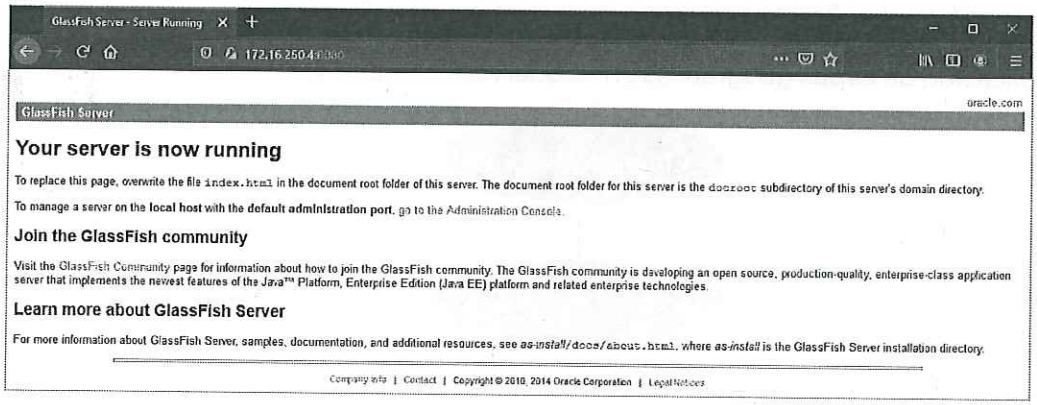
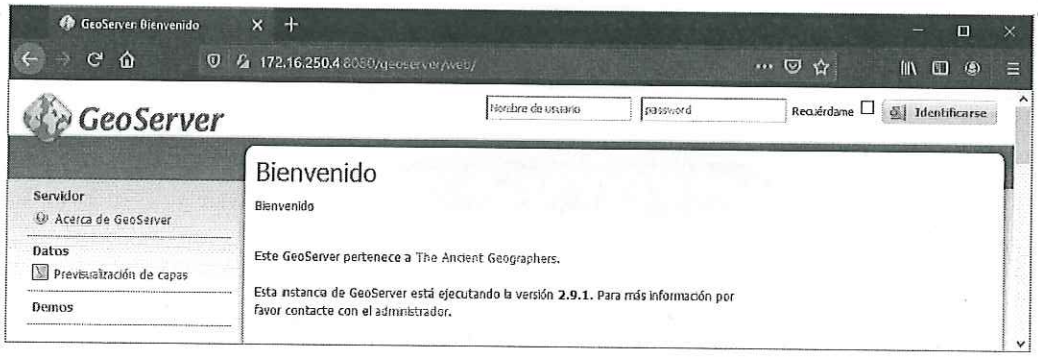


Imagen 2.45 GeoServer en el navegador

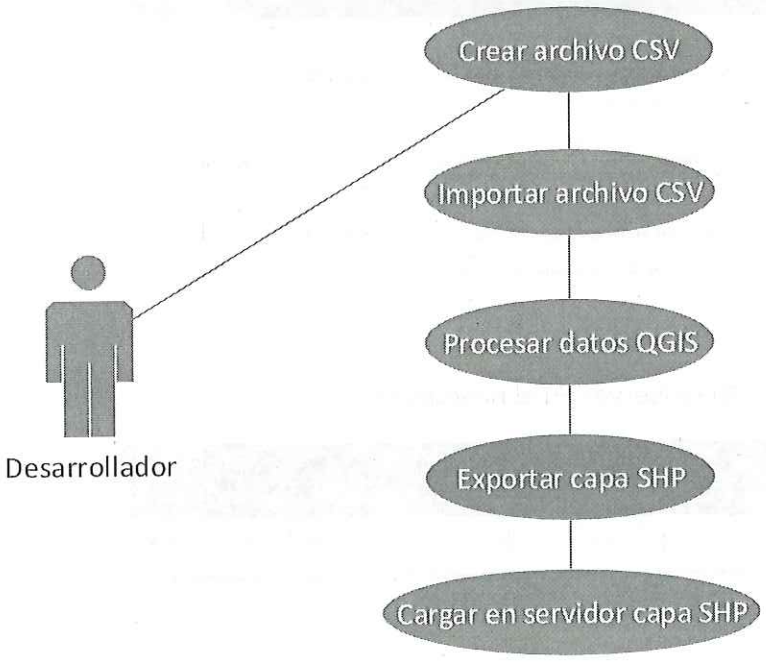


Implementación GeoServer

1. El primer paso para realizar una capa SHP, es la creación de un archivo CSV con los datos que se desee publicar, por ejemplo, para mostrar los puntos (latitud y longitud) e información de PMU 2019, se deberá ejecutar el siguiente query en el servidor de Base de Datos:

```
select t.id, l.descripcion as linea, cuv, t.edad, g.descripcion
as genero, t.longitud, t.latitud
from programas_conavi t
join c_programa_presupuestal pp on pp.id =
t.id_programa_presupuestal
join c_linea_apoyo_presidencia l on l.id = t.id_linea_apoyo
join c_genero g on g.id = t.id_genero
where t.id_programa_presupuestal = 2 and t.periodo_rep = 2019
and status = 1 and t.id_status = 1 and t.longitud is not null
and t.longitud != 0;
```

Diagrama 5 Casos de uso para la publicación de capas geográficas en GeoServer



2. El siguiente paso será importar el archivo CSV utilizando la herramienta QGIS para su visualización.

Imagen 2.46 Herramienta QGIS

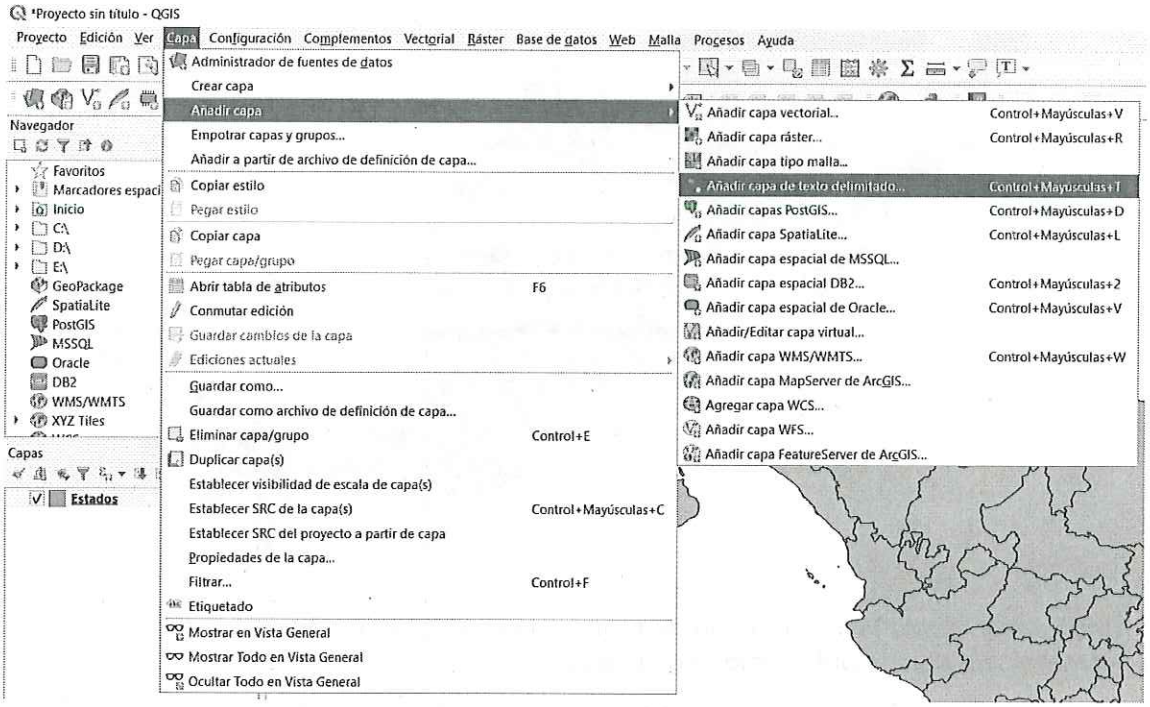


Imagen 2.47 Importar archivo CSV en QGIS

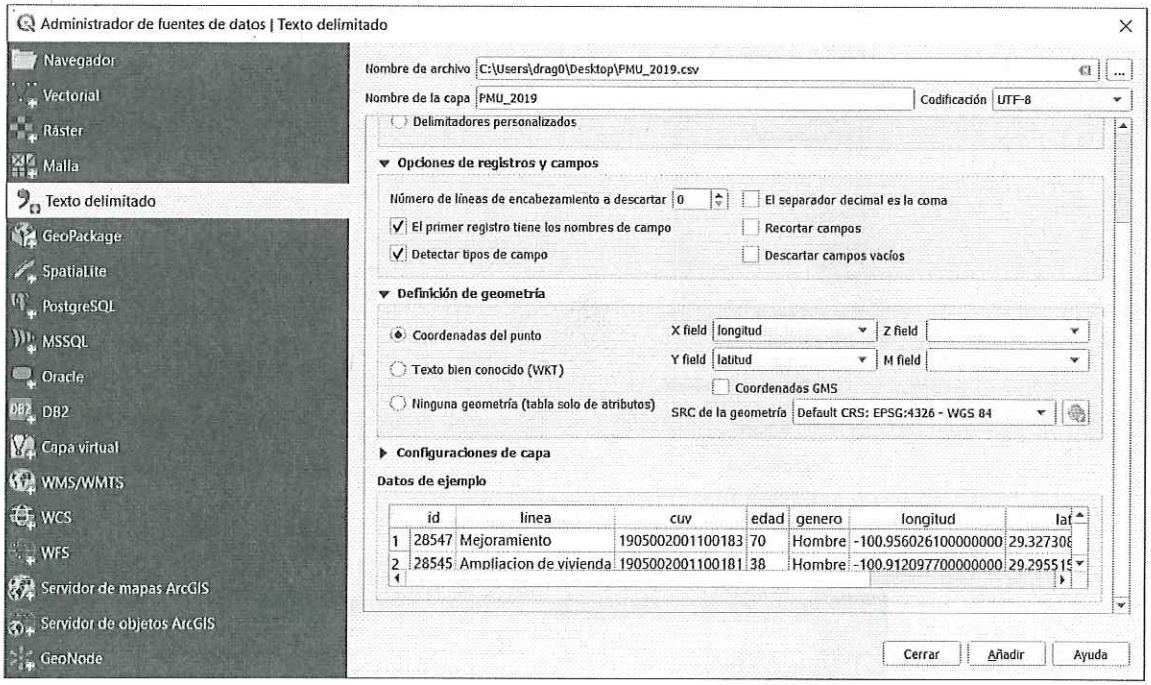


Imagen 2.48 Visualización de la capa "Entidades federativas" y "PMU 2019" en QGIS

3. Una vez mostrada la información de manera correcta, lo siguiente será exportar los datos en formato SHP para la publicación en el SNIIV.

Imagen 2.49 Exportar capa SHP en QGIS

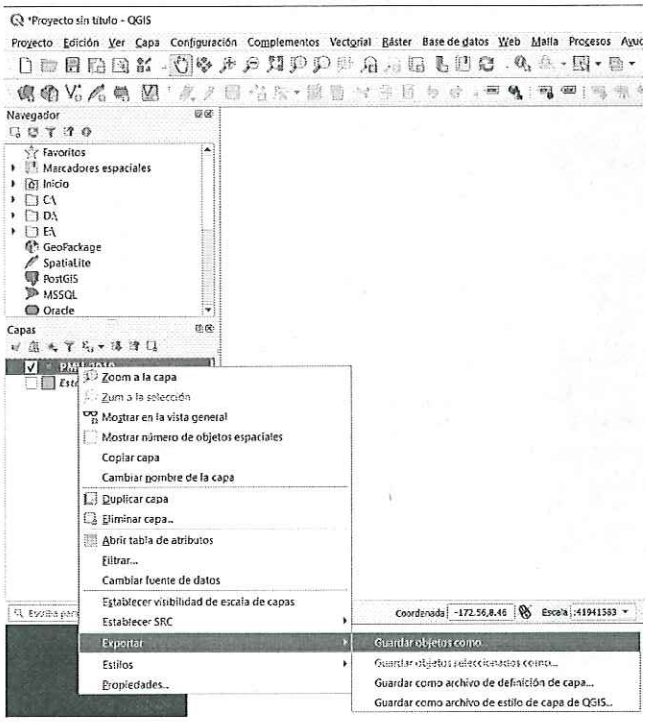
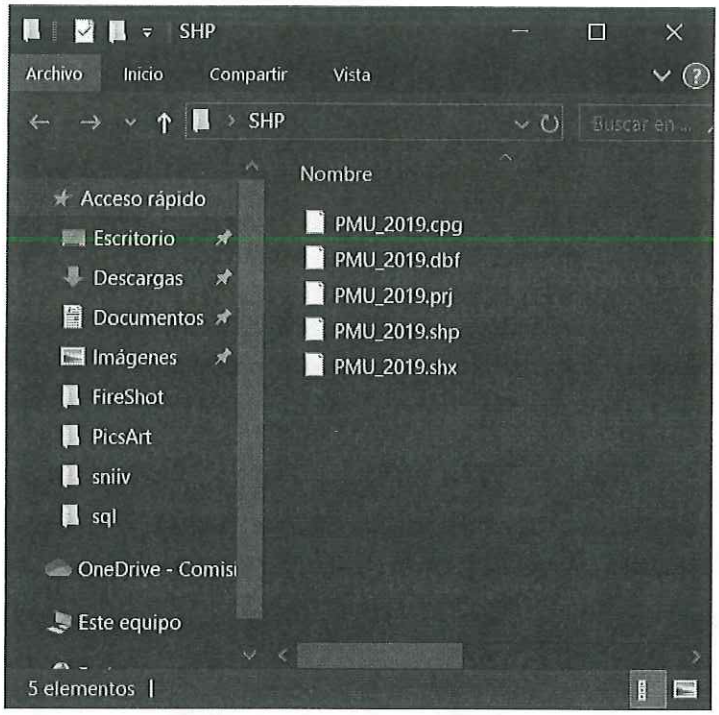


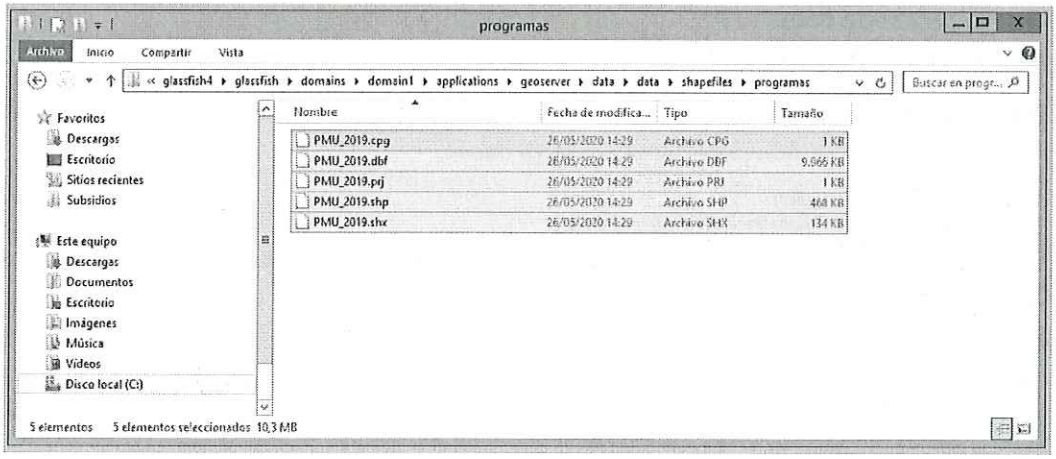
Imagen 2.50 Archivos creados al exportar capa SHP



4. Publicar las capas en el servidor Aplicativo almacenándolos en el siguiente directorio;

C:\Aplicativos\glassfish4\glassfish\domains\domain1\applications\geoserver\data\data\shapefiles\programas

Imagen 2.51 Directorio para archivos shapefiles



Nota: *Ver contraseñas del Servidor Aplicativo del archivo credenciales.*

5. Una vez se encuentren las capas almacenadas en el servidor, se procede a configurar el GeoServer para la implementación de WMS, en el caso que no exista un almacén de datos, el paso siguiente será crearlo.

Imagen 2.52 Almacén de datos en GeoServer

Nota: *Ver contraseñas del GeoServer del archivo credenciales.*

Imagen 2.53 Agregar un nuevo almacén

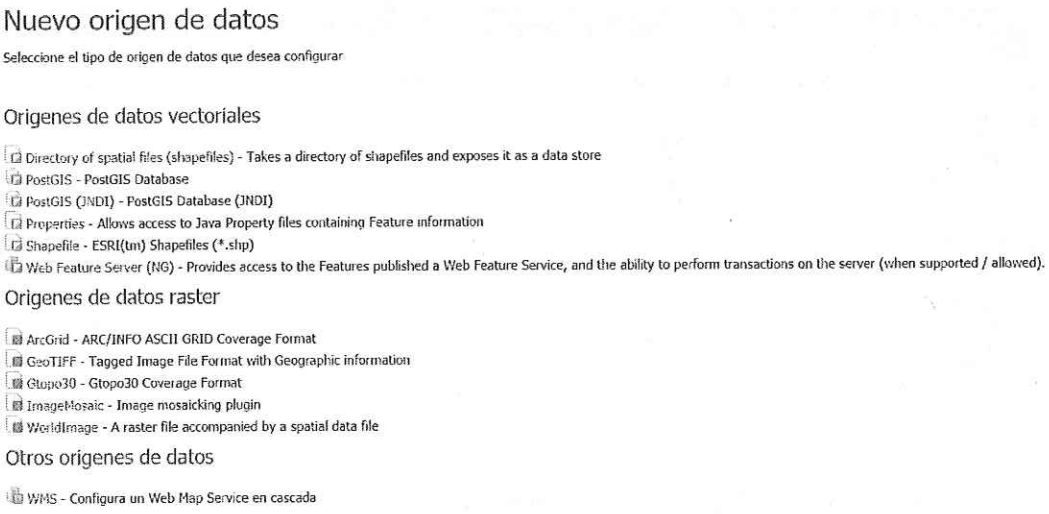


Imagen 2.54 Agregar nuevo origen de datos Shapefile

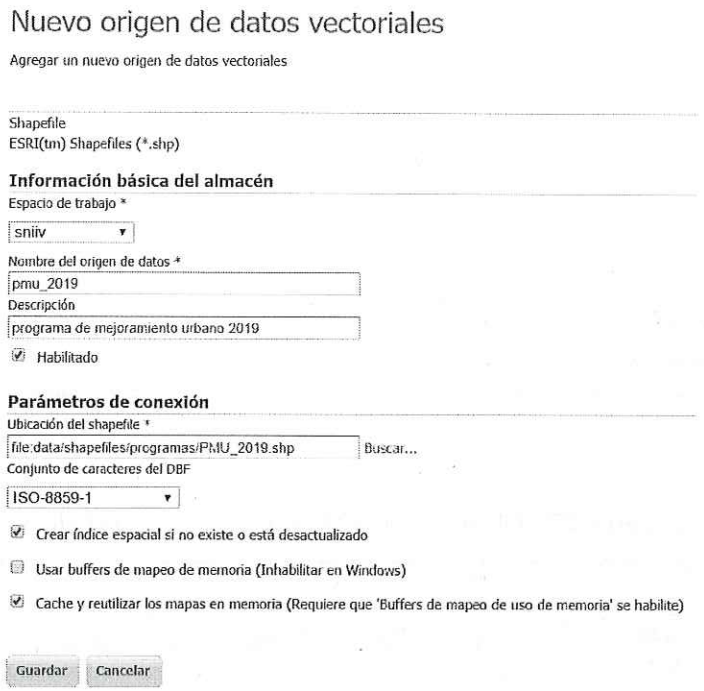
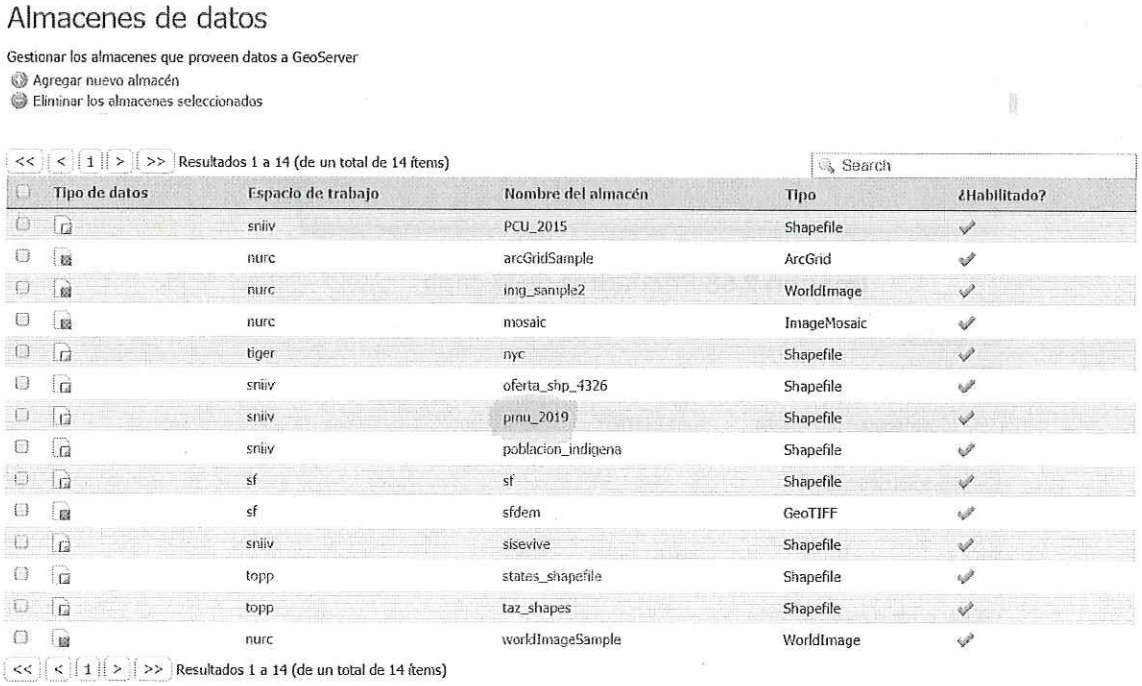


Imagen 2.55 Nuevo almacén de datos PMU 2019



6. Lo siguiente será agregar una nueva capa eligiendo el almacén de datos creado anteriormente y seleccionar la opción publicación, posteriormente seleccionar el sistema de coordenadas **WGS 84** y los encuadres en la opción **Calcular desde los datos** y finalmente seleccionar **Guardar**.

Imagen 2.56 Agregar una nueva capa

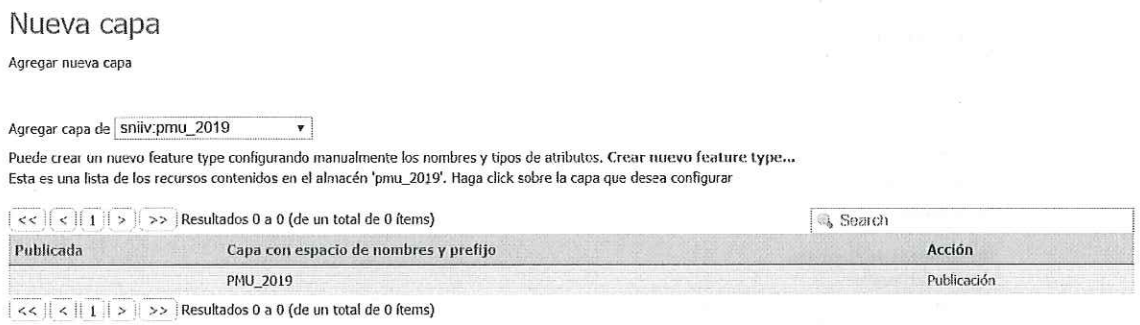


Imagen 2.57 Sistema de coordenadas WGS 84 de la capa

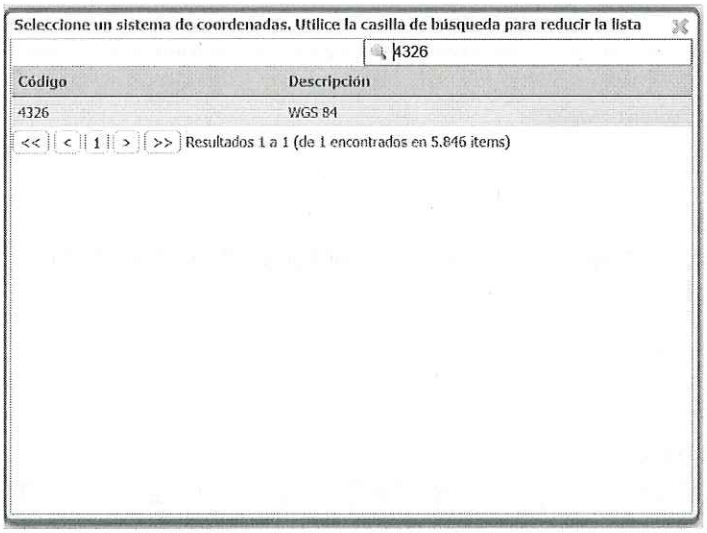
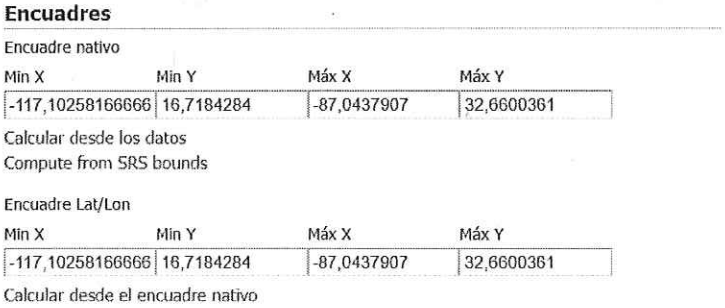
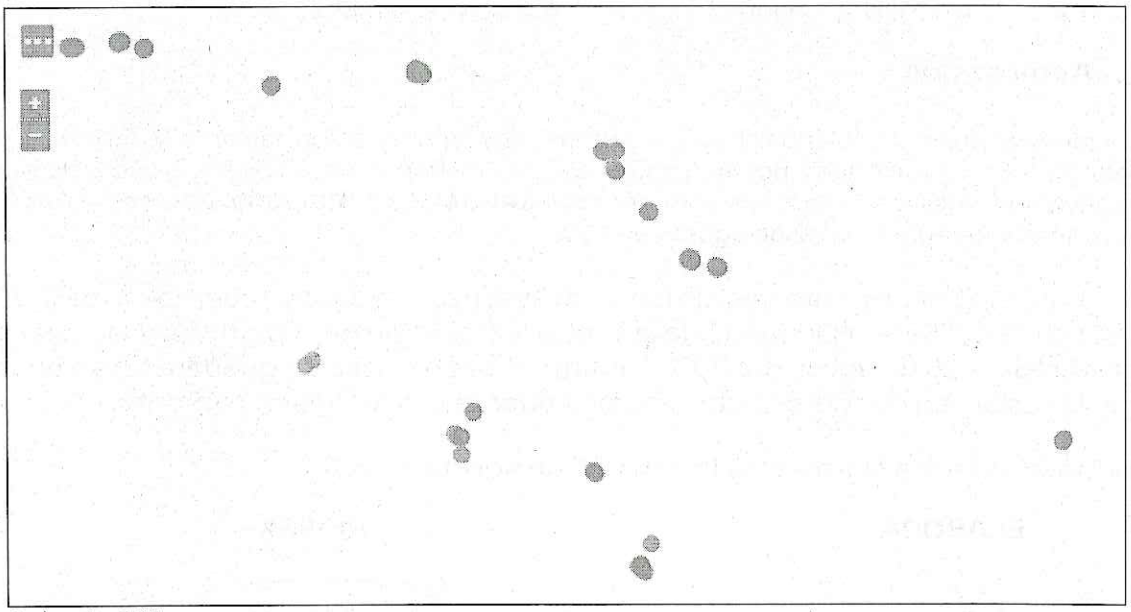


Imagen 2.58 Encuadres de la capa



7. El paso final será visualizar la capa y la información en WMS seleccionando la opción **Previsualización de capas** y seleccionando la opción **Open Layers**.

Imagen 2.59 Previsualización de la capa PMU 2019 en Open Layers



Scale = 1 : 17M

PMU_2019

fid	id	linea	cuv	edad	genero	longitud	latitud
PMU_2019.233	27778	Ampliacion de vivienda	1912001001101815	51	Hombre	-99.84967	16.9111933333333
PMU_2019.234	27773	Ampliacion de vivienda	1912001001101810	37	Hombre	-99.8480894806536	16.9115734569129
PMU_2019.235	27765	Ampliacion de vivienda	1912001001101802	66	Hombre	-99.84945905	16.90952223
PMU_2019.236	27749	Ampliacion de vivienda	1912001001101786	69	Hombre	-99.84892073	16.90974134
PMU_2019.237	27719	Ampliacion de vivienda	1912001001101757	81	Hombre	-99.8544585682493	16.9104241614213
PMU_2019.238	27716	Ampliacion de vivienda	1912001001101755	66	Hombre	-99.8515054817886	16.9107308047772
PMU_2019.239	27714	Ampliacion de vivienda	1912001001101753	70	Hombre	-99.84830094	16.91004164
PMU_2019.240	27710	Ampliacion de vivienda	1912001001101749	62	Hombre	-99.8510945678583	16.9113365821521
PMU_2019.241	27708	Ampliacion de vivienda	1912001001101747	45	Hombre	-99.8501279366277	16.9078799839539

VI.1. Transitorios

Único. – El instrumento que nos ocupa, sustituye el Manual Técnico: Sistema Nacional de Indicadores de Vivienda (SNIIV) anterior, publicado en la Normateca Interna de la Institución en 2020. Lo anterior debido a la actualización de la documentación técnica y de servidores del Sistema.

VI.1. Autorización

Las modificaciones al presente "Manual Técnico: Sistema Nacional de Información e Indicadores de Vivienda (SNIIV)", fueron aprobadas por el Comité de Mejora Regulatoria Interna de la Comisión Nacional de Vivienda, mediante Acuerdo número **COMERI-04-3EXTORD-20082021**, de la Tercera Sesión Extraordinaria, celebrada el 20 de agosto de 2021.

Asimismo, la emisión del presente instrumento, se autoriza mediante Acuerdo número JG-62-250821-861, de la 62ª Sesión Ordinaria de la Junta de Gobierno de la Comisión Nacional de Vivienda, celebrada el 25 de agosto de 2021. Las firmas de autorización quedarán plasmadas en el acta de la sesión, en donde se dictamine procedente el acuerdo en comento.

Fecha de publicación en la Normateca Interna: 26 de agosto de 2021.

ELABORA

María Fernanda Borja Ruiz
Subdirectora de Logística Territorial

REVISA

Urb. Daniel González Escobar
Director de Análisis de Información Territorial y Seguimiento de Programas de Vivienda

AUTORIZA

Lic. Silvia Circe Díaz Duarte
Subdirectora General de Análisis de Vivienda, Prospectiva y Sustentabilidad

V. HISTORIAL DE CAMBIOS

REVISIÓN NO	FECHA DE EMISIÓN	DESCRIPCIÓN DEL CAMBIO	MOTIVO
01	Diciembre, 2020	Alta del procedimiento	Integración del Manual Técnico del Sistema Nacional de Información e Indicadores de Vivienda.
02	Agosto, 2021	Modificación del Proyecto	Actualización de normativa y adición de material para enriquecer el documento