In [1]:
```
import pandas as pd, matplotlib.pyplot as plt #geopandas as gpd
```

In [2]:
```
datos_completos = pd.read_csv("PEMEX PLANTILA.csv", encoding='ISO-8859-1')
```

In [3]:
```
datos_completos.head(2)
```

Out[3]:

	ID_REGISTRO DE MANTENIMIENTO	TIPO DE MANTENIMIENTO	FECHA DE REPORTE	ID_TAD	NOMBRE_TAD	NUM_ETAPA	AVISO AVERIA SAP	ID_EQUIPO	NOMBRE_EQUIPO	ID_PAR
0	1	Correctivo	12/14/2023	622	TAD ESCAMELA	7	12101520.0	4.0	HMI	
1	2	Correctivo	11/6/2023	610	TAD MORELIA	5	12087624.0	8.0	SKID BOMBAS DE INYECCIÓN DE ADITIVOS	

2 rows × 32 columns

In [4]:
```
fallas_por_tad = datos_completos.groupby('NOMBRE_TAD').size().reset_index(name='cantidad_fallas')
```

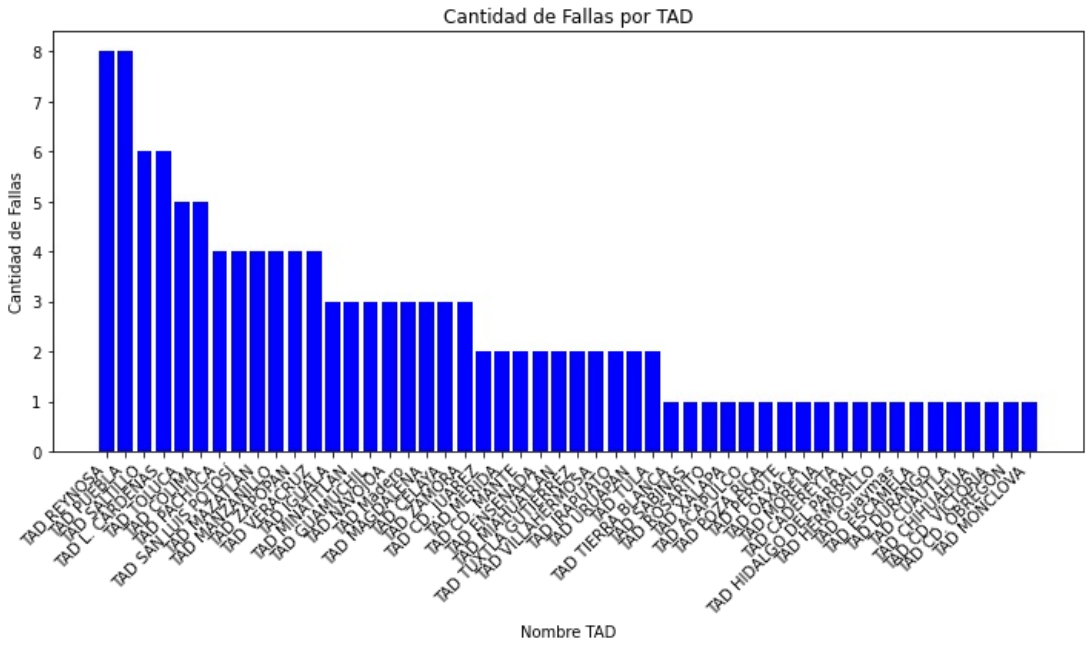
In [6]:
```
fallas_por_tad = fallas_por_tad.sort_values(by='cantidad_fallas', ascending=False)
```

In [7]:
```
fallas_por_tad.head(10)
```

Out[7]:

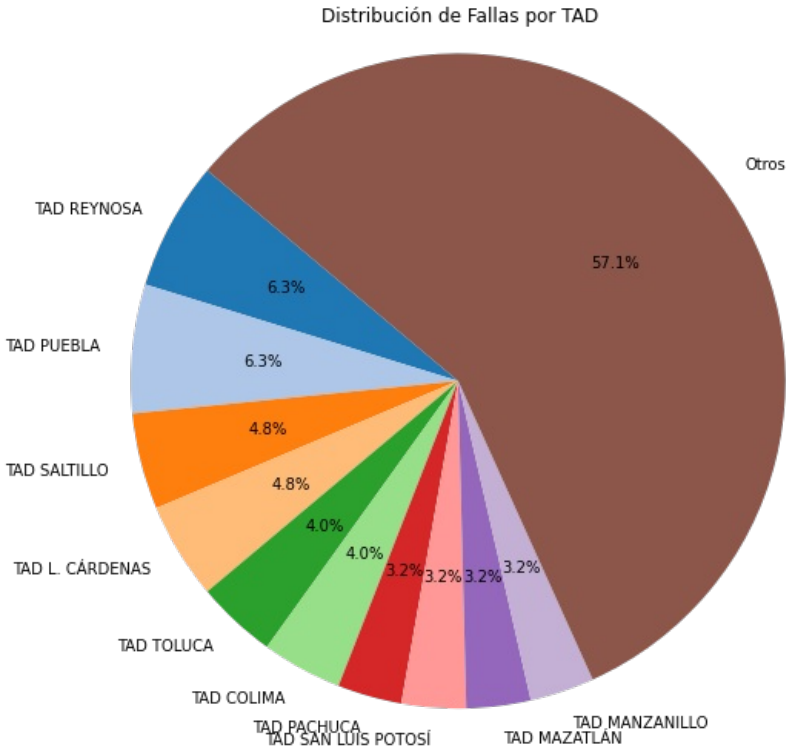
	NOMBRE_TAD	cantidad_fallas
35	TAD REYNOSA	8
34	TAD PUEBLA	8
38	TAD SALTILLO	6
19	TAD L. CÁRDENAS	6
41	TAD TOLUCA	5
8	TAD COLIMA	5
31	TAD PACHUCA	4
39	TAD SAN LUIS POTOSÍ	4
22	TAD MAZATLÁN	4
21	TAD MANZANILLO	4

In [8]:
```
plt.figure(figsize=(10, 6))
plt.bar(fallas_por_tad['NOMBRE_TAD'], fallas_por_tad['cantidad_fallas'], color='blue')
plt.xlabel('Nombre TAD')
plt.ylabel('Cantidad de Fallas')
plt.title('Cantidad de Fallas por TAD')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



In [9]:
```
top_10 = fallas_por_tad.head(10)
otros = pd.DataFrame({
    'NOMBRE_TAD': ['Otros'],
```

```
    'cantidad_fallas': [fallas_por_tad['cantidad_fallas'].iloc[10:].sum()]
})

# Concatenar los datos de los 10 primeros con el de "otros"
nuevos_datos = pd.concat([top_10, otros])

# Lista de colores
paleta_colores = plt.get_cmap('tab20')
colores = paleta_colores(range(len(nuevos_datos)))


# Crear el gráfico de pastel con los 10 primeros y "otros"
plt.figure(figsize=(9.9, 6.9))
plt.pie(nuevos_datos['cantidad_fallas'], labels=nuevos_datos['NOMBRE_TAD'], autopct='%1.1f%%', startangle=140,
plt.title('Distribución de Fallas por TAD')
plt.axis('equal')  # Hacer que el gráfico de pastel sea circular
plt.tight_layout()
plt.show()
```



In [10]:
```
fallas_por_equipo = datos_completos.groupby('NOMBRE_EQUIPO').size().reset_index(name='total_fallas')
```

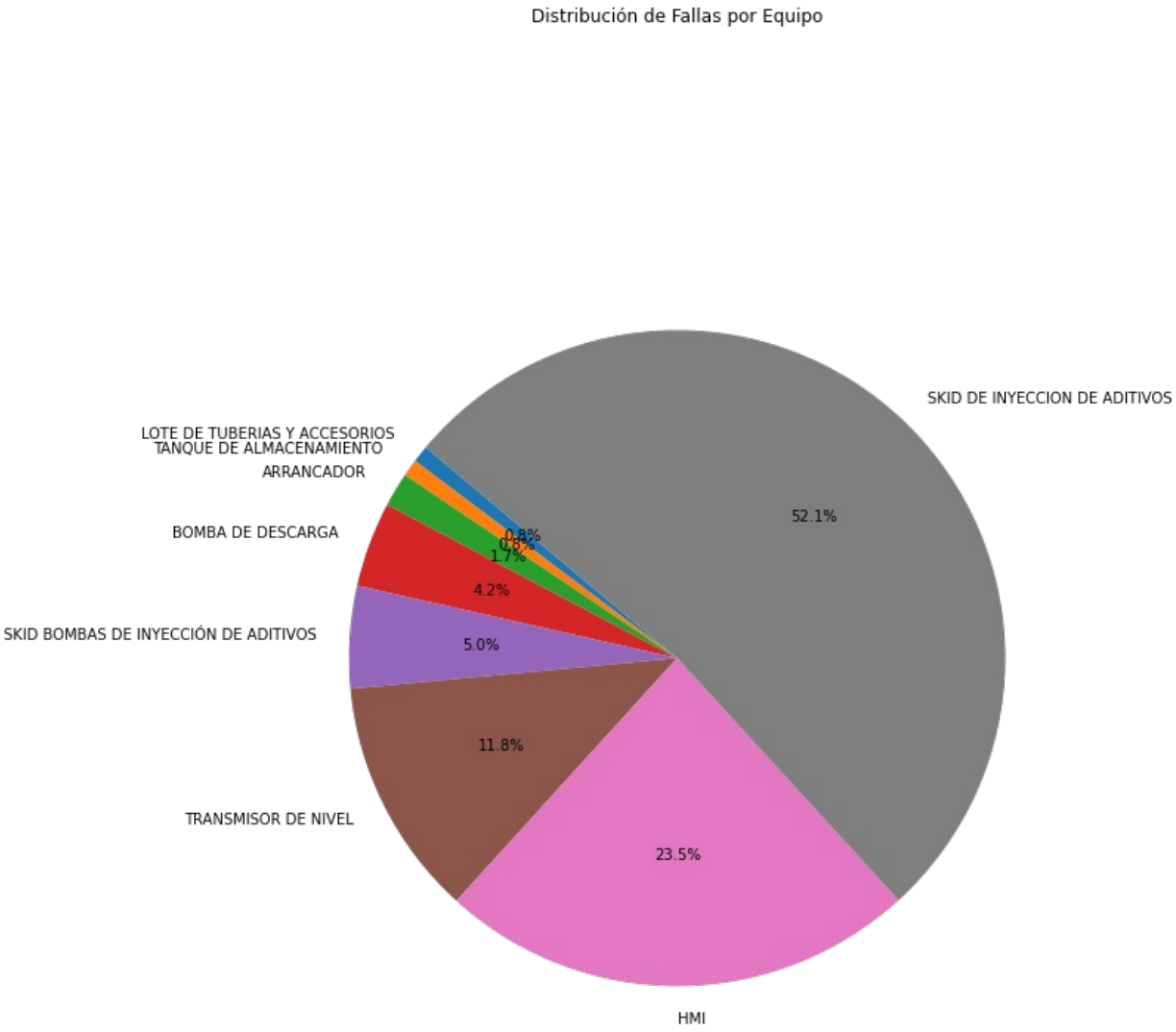
In [11]:
```
fallas_por_equipo = fallas_por_equipo.sort_values(by='total_fallas', ascending=True)
```

In [12]:
```
fallas_por_equipo
```

Out[12]:

	NOMBRE_EQUIPO	total_fallas
3	LOTE DE TUBERIAS Y ACCESORIOS	1
6	TANQUE DE ALMACENAMIENTO	1
0	ARRANCADOR	2
1	BOMBA DE DESCARGA	5
4	SKID BOMBAS DE INYECCIÓN DE ADITIVOS	6
7	TRANSMISOR DE NIVEL	14
2	HMI	28
5	SKID DE INYECCION DE ADITIVOS	62

In [13]:
```
plt.figure(figsize=(11, 12.3))
plt.pie(fallas_por_equipo['total_fallas'], labels=fallas_por_equipo['NOMBRE_EQUIPO'], autopct='%1.1f%%', starta
plt.title('Distribución de Fallas por Equipo')
plt.axis('equal')  # Hacer que el gráfico de pastel sea circular
plt.tight_layout()
plt.show()
```



In [14]:
```
# Calcular el número total de fallas por etapa
fallas_por_etapa = datos_completos.groupby('NUM_ETAPA').size().reset_index(name='total_fallas')
fallas_por_etapa = fallas_por_etapa.set_index('NUM_ETAPA')
```

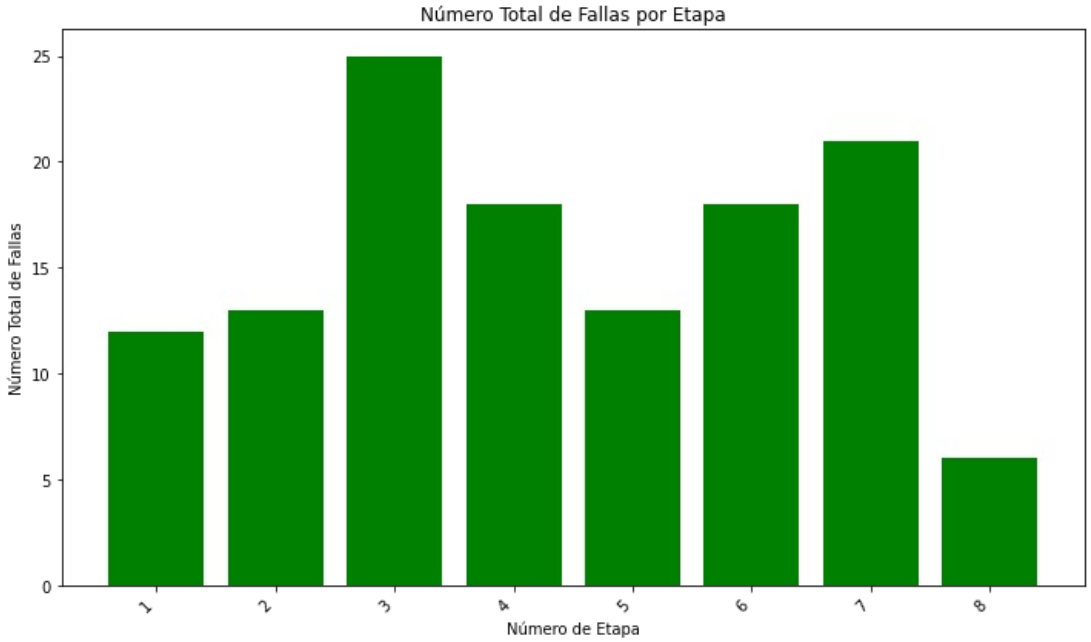
In [15]:
```
# Ordenar el resultado por el número total de fallas en orden ascendente
fallas_por_etapa
```

Out[15]:

NUM_ETAPA	total_fallas
1	12
2	13
3	25
4	18
5	13
6	18
7	21
8	6

In [16]:
```
plt.figure(figsize=(10, 6))
plt.bar(fallas_por_etapa.index, fallas_por_etapa['total_fallas'], color='green')
plt.xlabel('Número de Etapa')
plt.ylabel('Número Total de Fallas')
plt.title('Número Total de Fallas por Etapa')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
```

```
plt.show()
```



In [17]:
```
#formato fecha y mes
datos_completos['FECHA DE REPORTE'] = pd.to_datetime(datos_completos['FECHA DE REPORTE'])
```

In [18]:
```
reportes_por_fecha = datos_completos.groupby(datos_completos['FECHA DE REPORTE']).size().reset_index(name='tota
reportes_por_fecha = reportes_por_fecha.sort_values(by='total_reportes', ascending=False)

reportes_por_fecha.head(10)
```

Out[18]:

	FECHA DE REPORTE	total_reportes
40	2023-11-14	10
59	2024-01-24	6
42	2023-11-27	5
51	2024-01-08	5
67	2024-02-12	5
65	2024-02-07	4
32	2023-10-10	3
70	2024-02-19	3
60	2024-01-26	2
44	2023-12-01	2

In [19]:
```
reportes_por_proveedor = datos_completos.groupby('ID_Proveedor').size().reset_index(name='total_reportes')
```

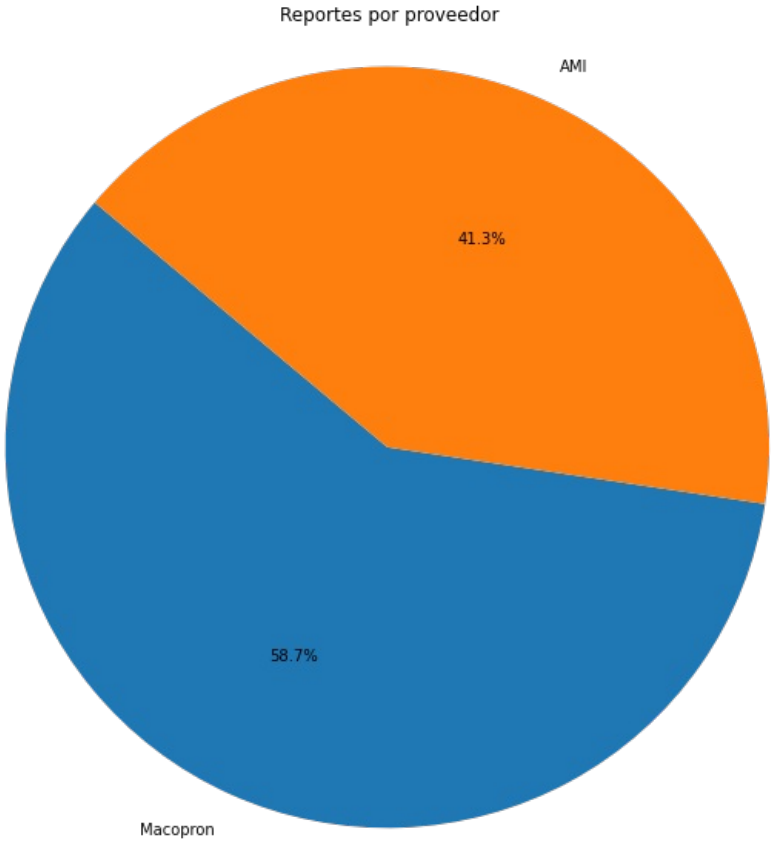
In [20]:
```
# Ordenar los resultados por el número total de reportes en orden descendente
reportes_por_proveedor = reportes_por_proveedor.sort_values(by='total_reportes', ascending=False)
reportes_por_proveedor
```

Out[20]:

	ID_Proveedor	total_reportes
0	111111	74
1	222222	52

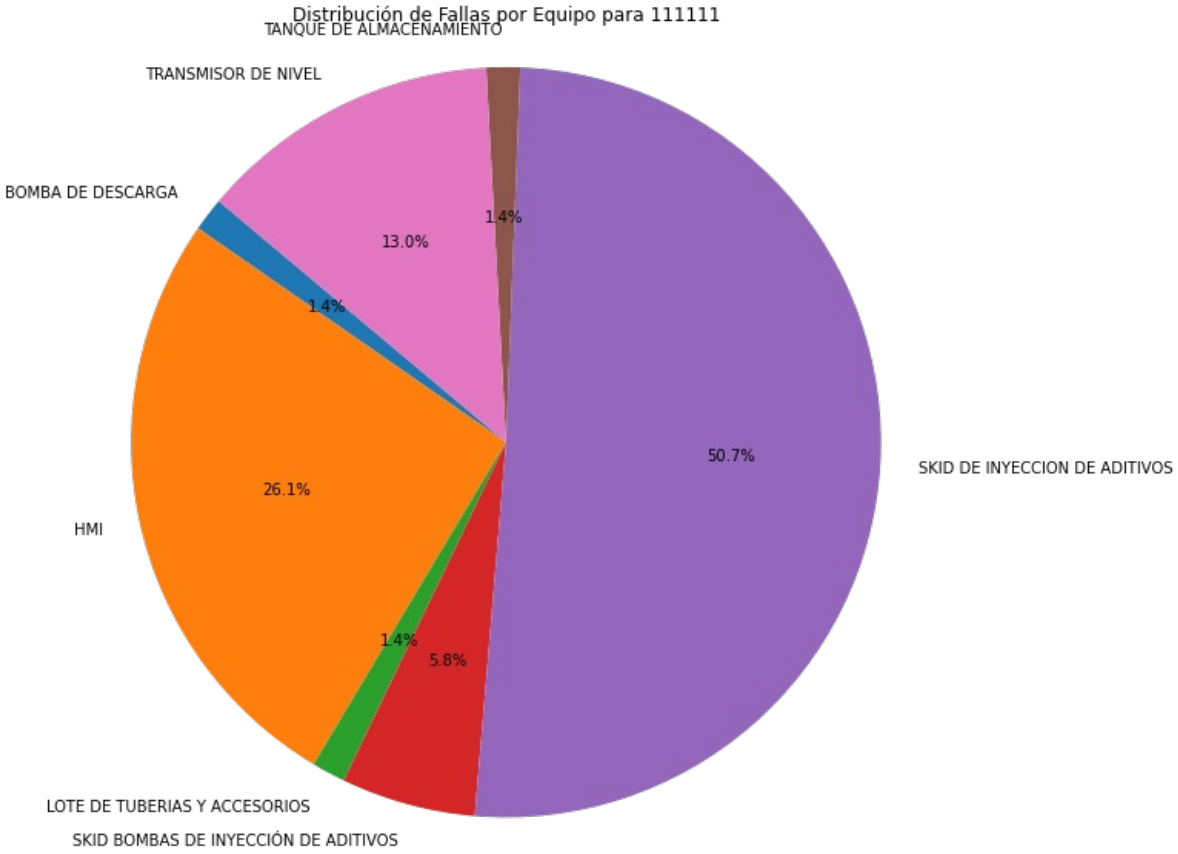
In [21]:
```
nombres_proveedores = ['Macopron', 'AMI']
```

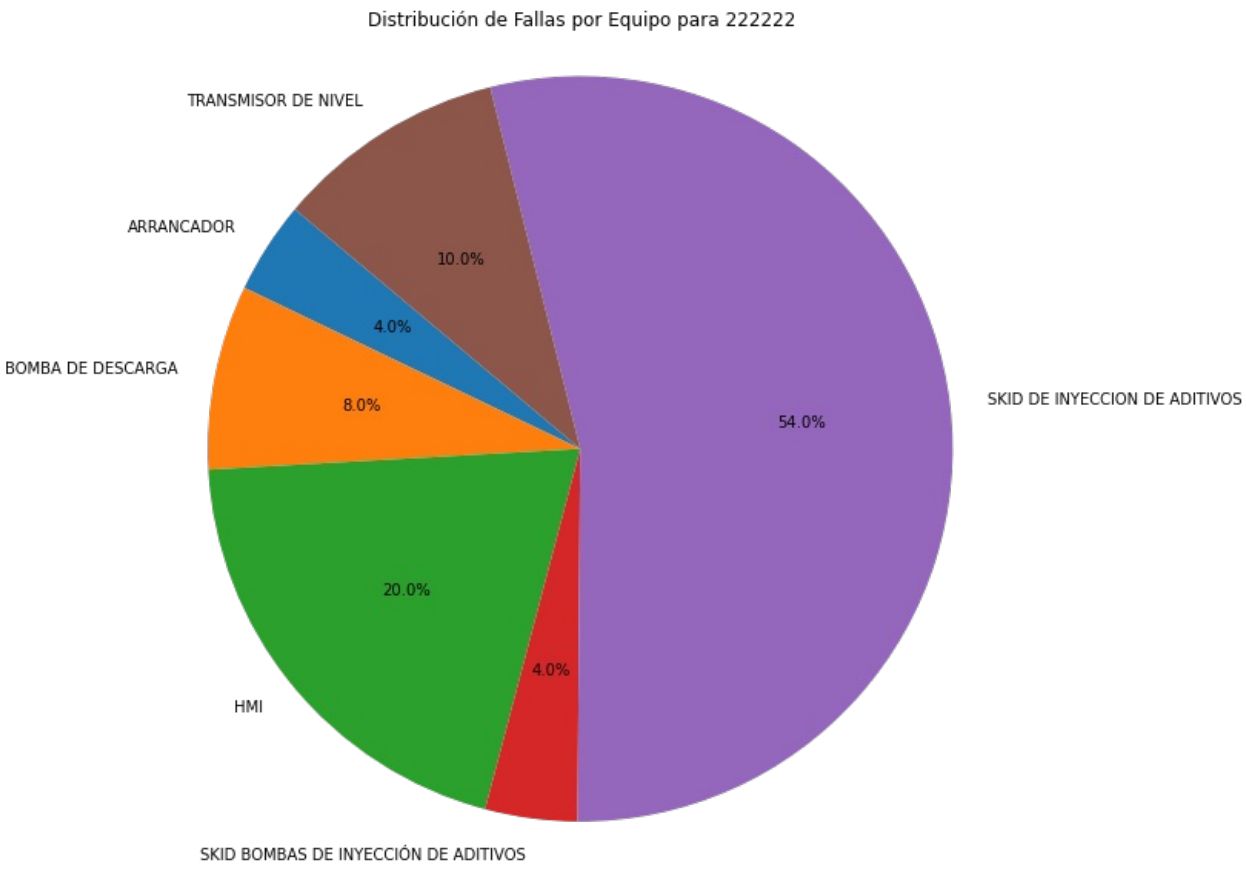
In [22]:
```
plt.figure(figsize=(8, 8))
plt.pie(reportes_por_proveedor['total_reportes'], labels=nombres_proveedores, autopct='%1.1f%%', startangle=140
plt.title('Reportes por proveedor')
plt.axis('equal')  # Hacer que el gráfico de pastel sea circular
plt.tight_layout()
plt.show()
```



In [23]:
```
#datos_por_proveedor = datos_completos.groupby('ID_Proveedor').apply(lambda x: x.reset_index(drop=True))
#datos_por_proveedor
```

In [24]:
```
fallas_por_equipo_y_proveedor = datos_completos.groupby(['ID_Proveedor', 'NOMBRE_EQUIPO']).size().reset_index(n

# Lista de proveedores a considerar
proveedores = fallas_por_equipo_y_proveedor['ID_Proveedor'].unique()

# Graficar un gráfico de pastel por cada proveedor
for nombre_proveedores in proveedores:
# Filtrar los datos para el proveedor actual
    datos_proveedor = fallas_por_equipo_y_proveedor[fallas_por_equipo_y_proveedor['ID_Proveedor'] == nombre_pro

# Preparar los datos para el gráfico de pastel
    labels = datos_proveedor['NOMBRE_EQUIPO']
    sizes = datos_proveedor['total_fallas']

# Crear el gráfico de pastel
    plt.figure(figsize=(10, 8))
    plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140)
    plt.title(f'Distribución de Fallas por Equipo para {nombre_proveedores}')
    plt.axis('equal')  # Hacer que el gráfico de pastel sea circular
    plt.tight_layout()
    plt.show()
```

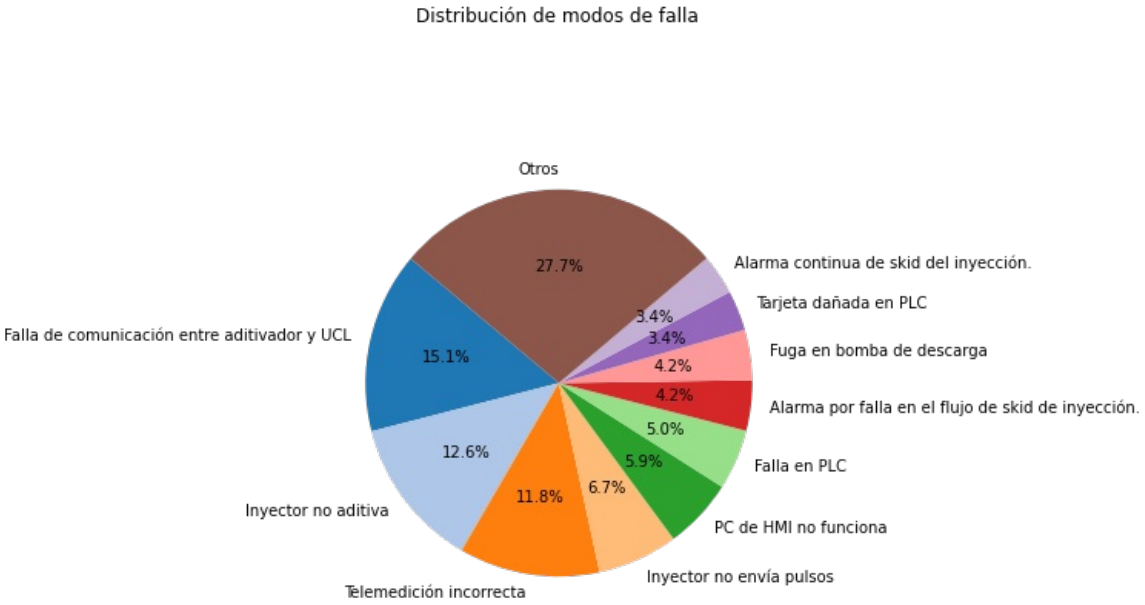




In [25]:
```
modo_falla = datos_completos.groupby('AVERÍA (MODO DE FALLA)').size().reset_index(name='total_fallas')

modo_falla = modo_falla.sort_values(by='total_fallas', ascending=False)
```

In [26]:
```
top_10 = modo_falla.head(10)
otros = pd.DataFrame({
    'AVERÍA (MODO DE FALLA)': ['Otros'],
    'total_fallas': [modo_falla['total_fallas'].iloc[10:].sum()]
})

# Concatenar los datos de los 10 primeros con el de "otros"
nuevos_datos = pd.concat([top_10, otros])

# Lista de colores
paleta_colores = plt.get_cmap('tab20')
colores = paleta_colores(range(len(nuevos_datos)))
```

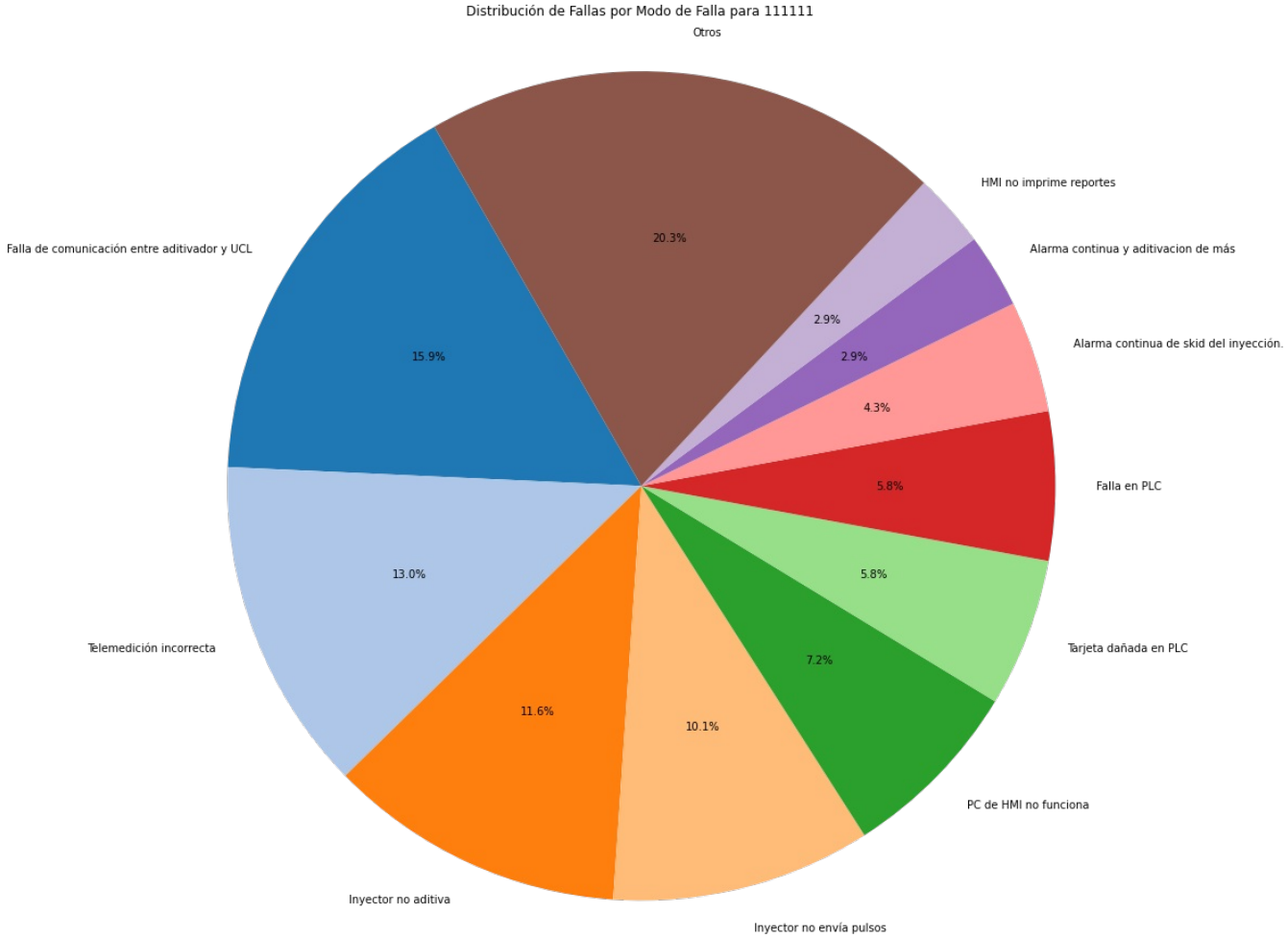
```
# Crear el gráfico de pastel con los 10 primeros y "otros"
plt.figure(figsize=(9.9, 6.9))
plt.pie(nuevos_datos['total_fallas'], labels=nuevos_datos['AVERÍA (MODO DE FALLA)'], autopct='%1.1f%%', startan
plt.title('Distribución de modos de falla')
plt.axis('equal')  # Hacer que el gráfico de pastel sea circular
plt.tight_layout()
plt.show()
```

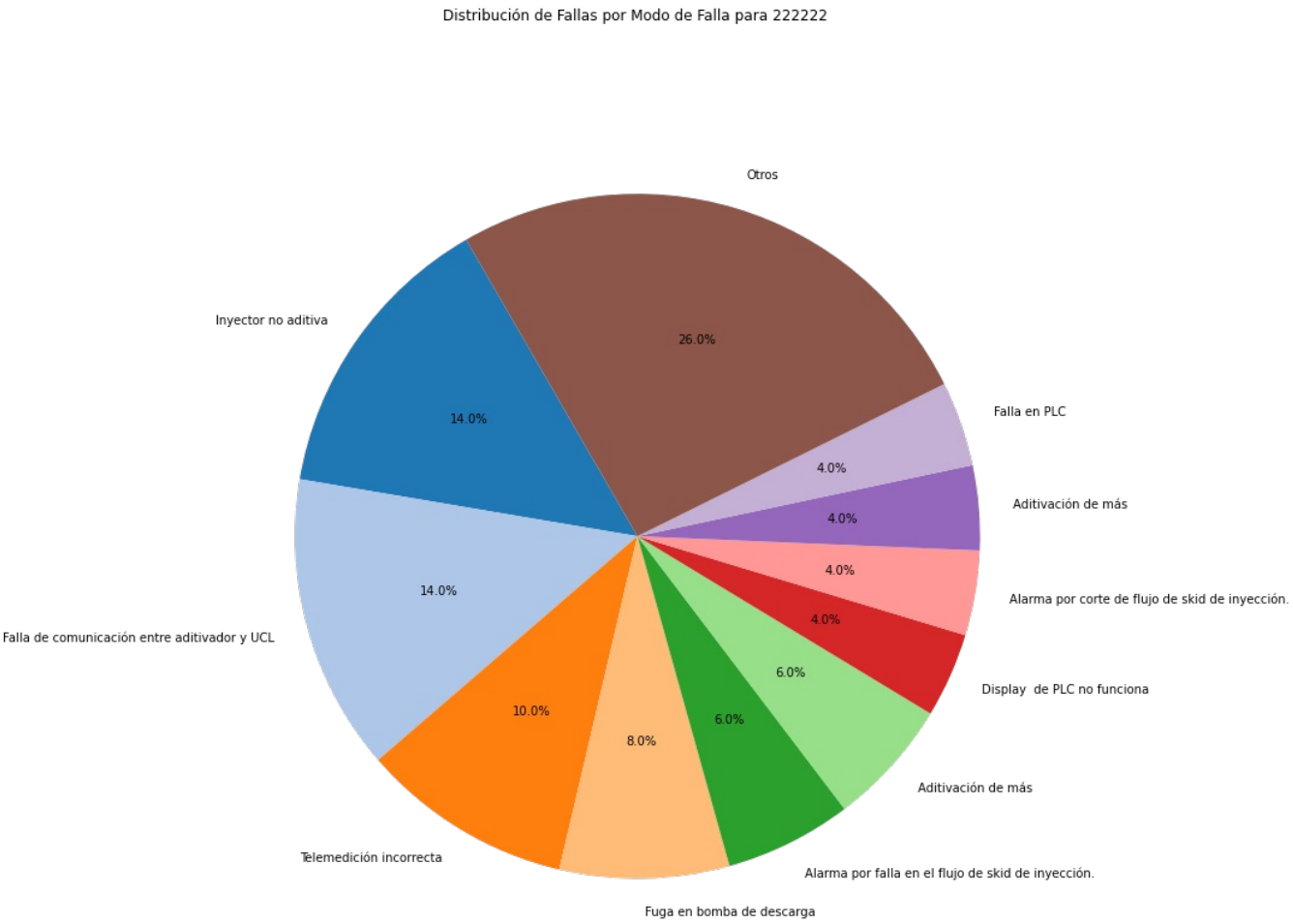


In [27]:
```
#fallas_por_modo_y_proveedor = datos_completos.groupby(['ID_Proveedor', 'AVERÍA (MODO DE FALLA)']).size().reset

#proveedores = fallas_por_modo_y_proveedor['ID_Proveedor'].unique()

# Graficar un gráfico de pastel por cada proveedor
#for proveedor in proveedores:
    # Filtrar los datos para el proveedor actual
 #   datos_proveedor = fallas_por_modo_y_proveedor[fallas_por_modo_y_proveedor['ID_Proveedor'] == proveedor]

    # Preparar los datos para el gráfico de pastel
  #  labels = datos_proveedor['AVERÍA (MODO DE FALLA)']
   # sizes = datos_proveedor['total_fallas']

    # Crear el gráfico de pastel
    #plt.figure(figsize=(15, 12.3))
    #plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=120)
    #plt.title(f'Distribución de Fallas por Modo de Falla para {proveedor}')
    #plt.axis('equal')  # Hacer que el gráfico de pastel sea circular
    #plt.tight_layout()

    #plt.show()
```

In [28]:
```
# Obtener el recuento de fallas por modo de falla y por proveedor
fallas_por_modo_y_proveedor = datos_completos.groupby(['ID_Proveedor', 'AVERÍA (MODO DE FALLA)']).size().reset_

# Lista de proveedores a considerar
proveedores = fallas_por_modo_y_proveedor['ID_Proveedor'].unique()

# Graficar un gráfico de pastel por cada proveedor
for proveedor in proveedores:
    # Filtrar los datos para el proveedor actual
    datos_proveedor = fallas_por_modo_y_proveedor[fallas_por_modo_y_proveedor['ID_Proveedor'] == proveedor]

    # Ordenar los datos por la cantidad de fallas en orden descendente
    datos_proveedor = datos_proveedor.sort_values(by='total_fallas', ascending=False)

    # Lista de colores
    paleta_colores = plt.cm.tab20

    # Seleccionar los 10 primeros y calcular la suma de los demás
    top_10 = datos_proveedor.head(10)
    otros = pd.DataFrame({
        'AVERÍA (MODO DE FALLA)': ['Otros'],
        'total_fallas': [datos_proveedor['total_fallas'].iloc[10:].sum()]
    })
```

```
    # Concatenar los datos de los 10 primeros con el de "otros"
    nuevos_datos = pd.concat([top_10, otros])
    colores = paleta_colores(range(len(nuevos_datos)))

    # Preparar los datos para el gráfico de pastel
    labels = nuevos_datos['AVERÍA (MODO DE FALLA)']
    sizes = nuevos_datos['total_fallas']

    # Crear el gráfico de pastel
    plt.figure(figsize=(15, 12.3))
    plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=120,colors=colores)
    plt.title(f'Distribución de Fallas por Modo de Falla para {proveedor}')
    plt.axis('equal')  # Hacer que el gráfico de pastel sea circular
    plt.tight_layout()

    plt.show()
```





In [29]:
```
fallas_por_equipo_y_modo = datos_completos.groupby(['NOMBRE_EQUIPO', 'AVERÍA (MODO DE FALLA)']).size().reset_in
fallas_por_equipo_y_modo.head(10)
```

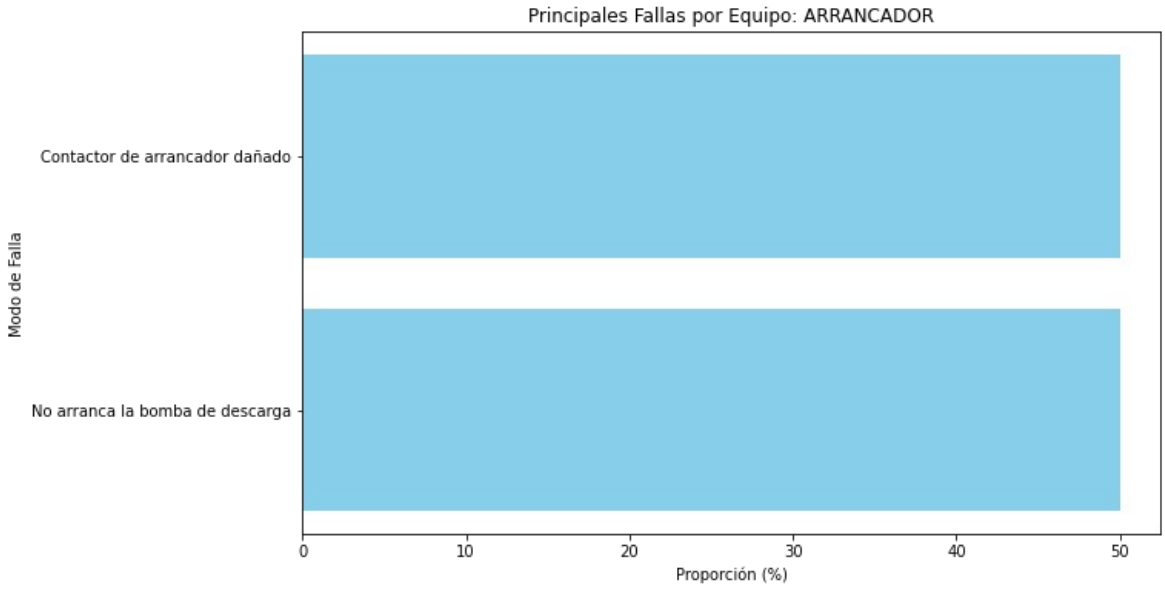
Out[29]:

	NOMBRE_EQUIPO	AVERÍA (MODO DE FALLA)	total_fallas
0	ARRANCADOR	Contactor de arrancador dañado	1
1	ARRANCADOR	No arranca la bomba de descarga	1
2	BOMBA DE DESCARGA	Fuga en bomba de descarga	5
3	HMI	Cable USB/AXIAL de HMI no funciona.	1
4	HMI	Control remoto de UCL no funciona	1
5	HMI	Falla de comunicación entre aditivador y UCL	14
6	HMI	Falla de conexión a la red de HMI	1
7	HMI	HMI no imprime reportes	4
8	HMI	PC de HMI no funciona	7
9	LOTE DE TUBERIAS Y ACCESORIOS	Falla en brida	1

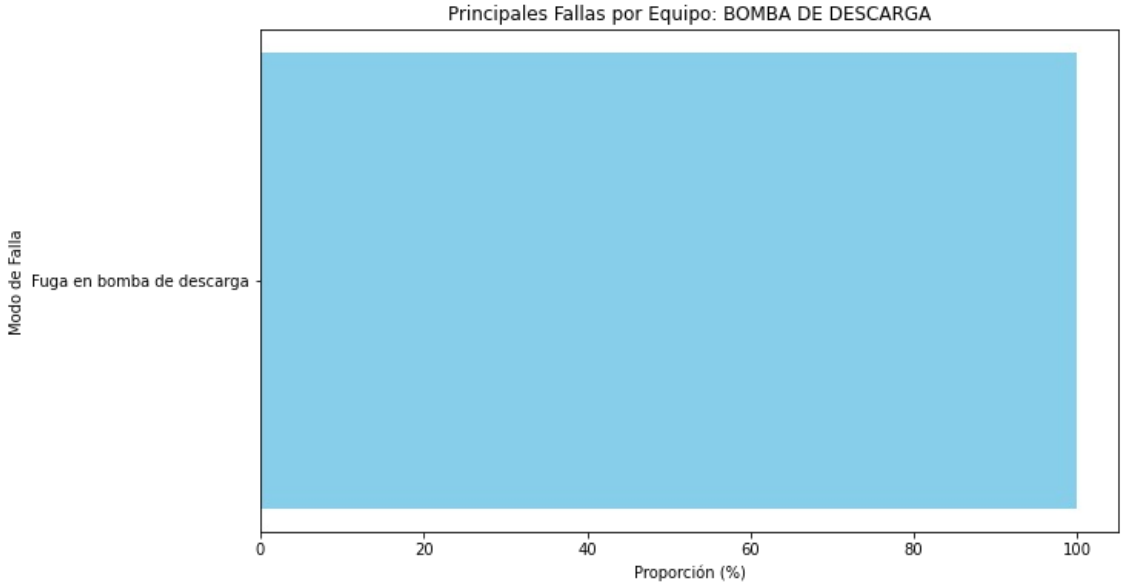
In [30]:

```
# Obtener el recuento de fallas por equipo y por modo de falla
fallas_por_equipo_y_modo = datos_completos.groupby(['NOMBRE_EQUIPO', 'AVERÍA (MODO DE FALLA)']).size().reset_in

# Lista de equipos a considerar
equipos = fallas_por_equipo_y_modo['NOMBRE_EQUIPO'].unique()

# Definir la cantidad de principales fallas a mostrar por equipo
principales_fallas_por_equipo = 5

for equipo in equipos:
    # Filtrar los datos para el equipo actual
    datos_equipo = fallas_por_equipo_y_modo[fallas_por_equipo_y_modo['NOMBRE_EQUIPO'] == equipo]

# Ordenar los datos por la cantidad de fallas en orden descendente
    datos_equipo = datos_equipo.sort_values(by='total_fallas', ascending=False)

    # Seleccionar las principales fallas para el equipo actual
    principales_fallas = datos_equipo.head(principales_fallas_por_equipo)

    # Calcular el total de fallas para el equipo actual
    total_fallas_equipo = principales_fallas['total_fallas'].sum()

    # Calcular la proporción de cada modo de falla para el equipo actual
    principales_fallas['proporcion'] = principales_fallas['total_fallas'] / total_fallas_equipo * 100


    plt.figure(figsize=(10, 6))
    plt.barh(principales_fallas['AVERÍA (MODO DE FALLA)'], principales_fallas['proporcion'], color='skyblue')
    plt.xlabel('Proporción (%)')
    plt.ylabel('Modo de Falla')
    plt.title(f'Principales Fallas por Equipo: {equipo}')
    plt.gca().invert_yaxis()  # Invertir el eje y para mostrar las barras más grandes arriba
    plt.show()
```

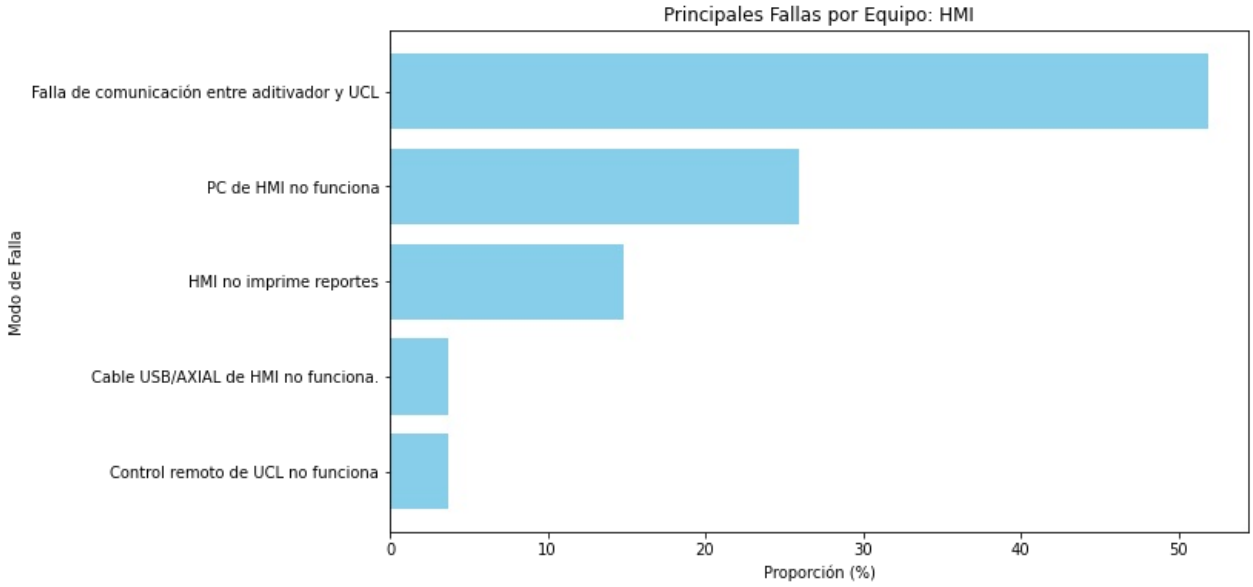
```
C:\Users\kevin\AppData\Local\Temp\ipykernel_137540\832426784.py:24: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  principales_fallas['proporcion'] = principales_fallas['total_fallas'] / total_fallas_equipo * 100
```



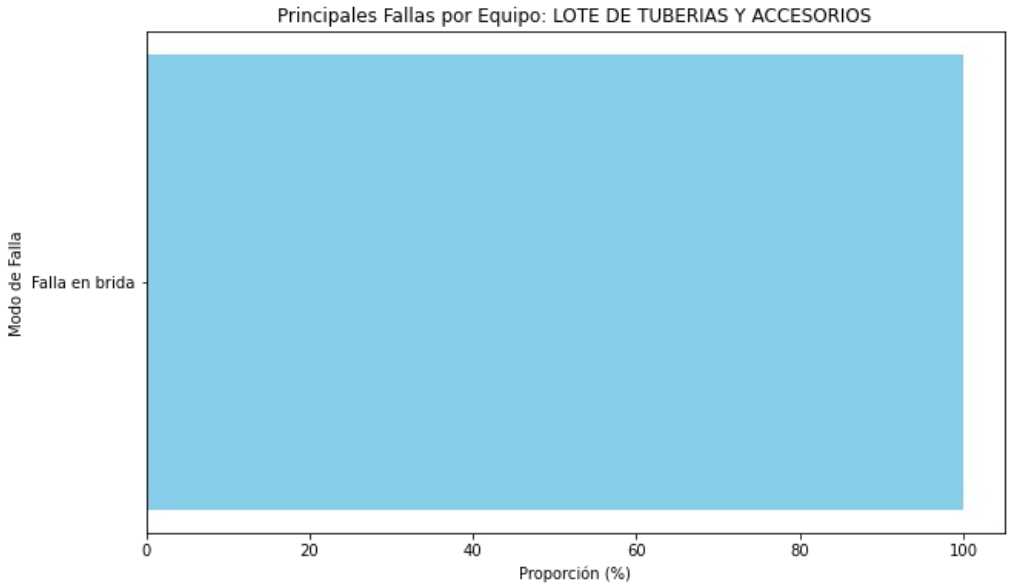
```
C:\Users\kevin\AppData\Local\Temp\ipykernel_137540\832426784.py:24: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead


See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  principales_fallas['proporcion'] = principales_fallas['total_fallas'] / total_fallas_equipo * 100
```



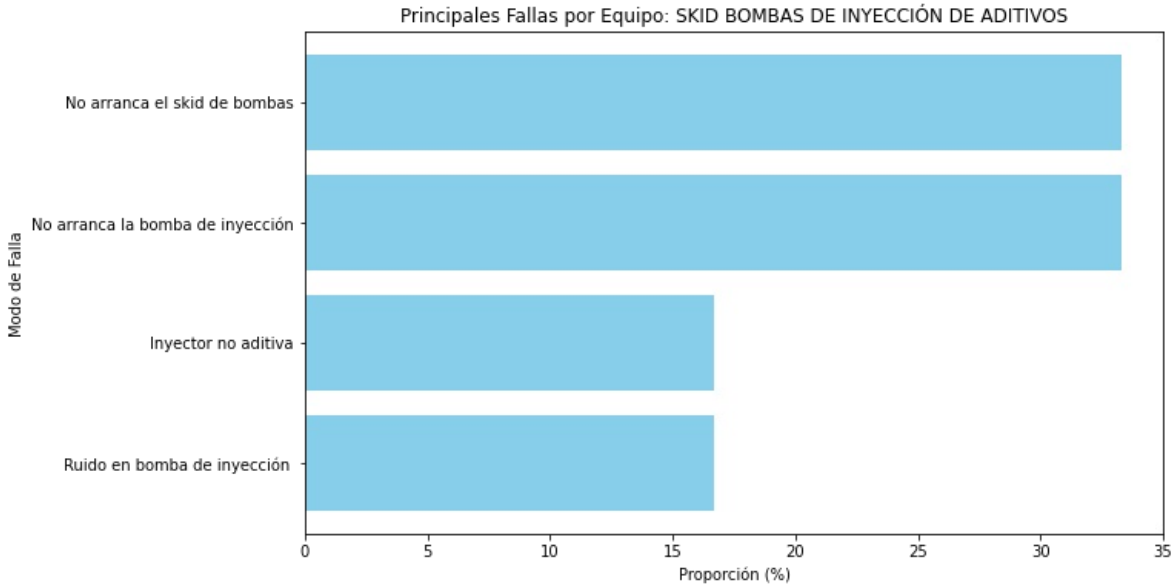

```
C:\Users\kevin\AppData\Local\Temp\ipykernel_137540\832426784.py:24: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  principales_fallas['proporcion'] = principales_fallas['total_fallas'] / total_fallas_equipo * 100
```



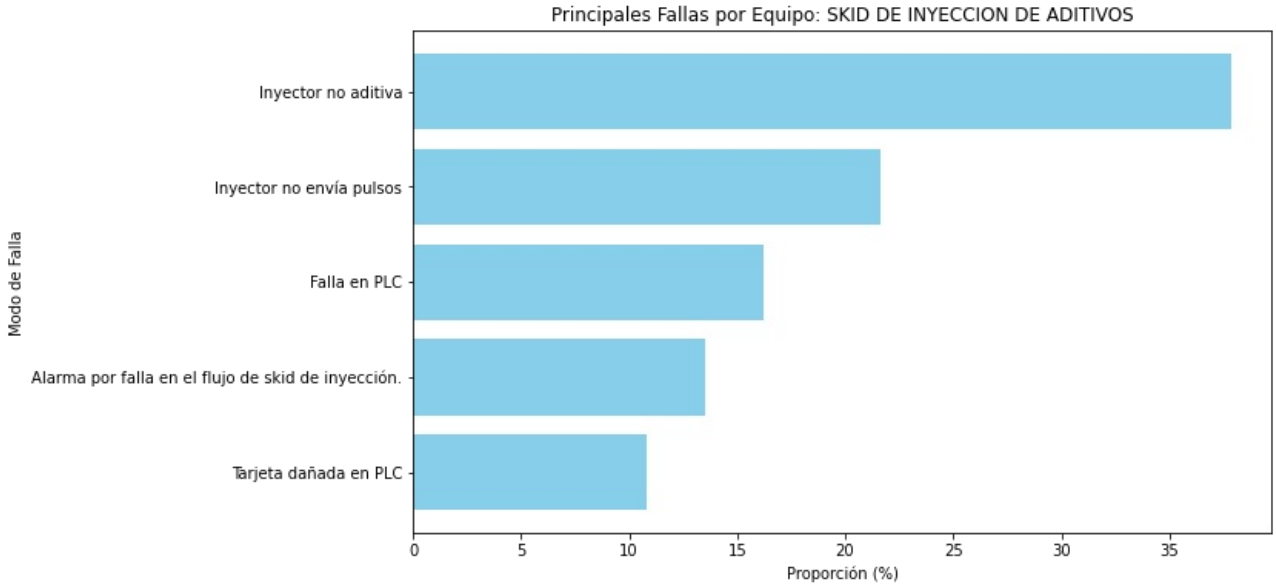

```
C:\Users\kevin\AppData\Local\Temp\ipykernel_137540\832426784.py:24: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  principales_fallas['proporcion'] = principales_fallas['total_fallas'] / total_fallas_equipo * 100
```



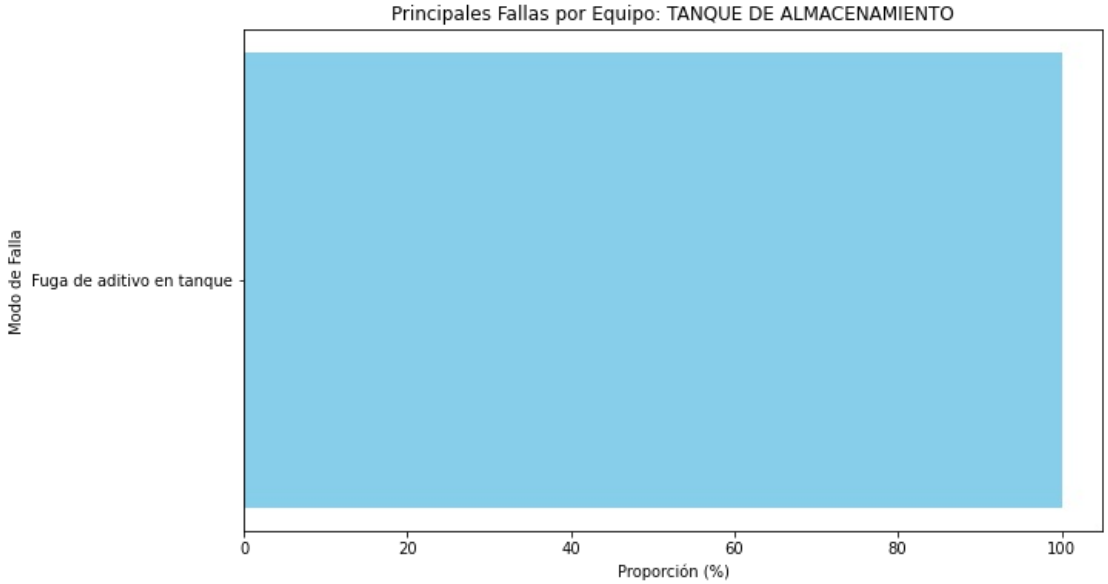

```
C:\Users\kevin\AppData\Local\Temp\ipykernel_137540\832426784.py:24: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  principales_fallas['proporcion'] = principales_fallas['total_fallas'] / total_fallas_equipo * 100
```



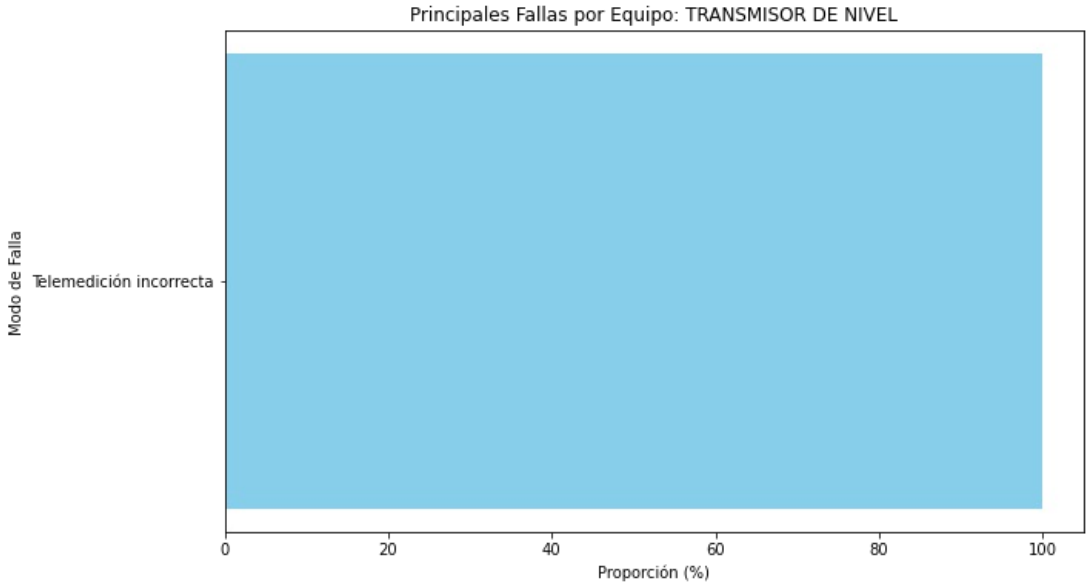

```
C:\Users\kevin\AppData\Local\Temp\ipykernel_137540\832426784.py:24: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  principales_fallas['proporcion'] = principales_fallas['total_fallas'] / total_fallas_equipo * 100
```



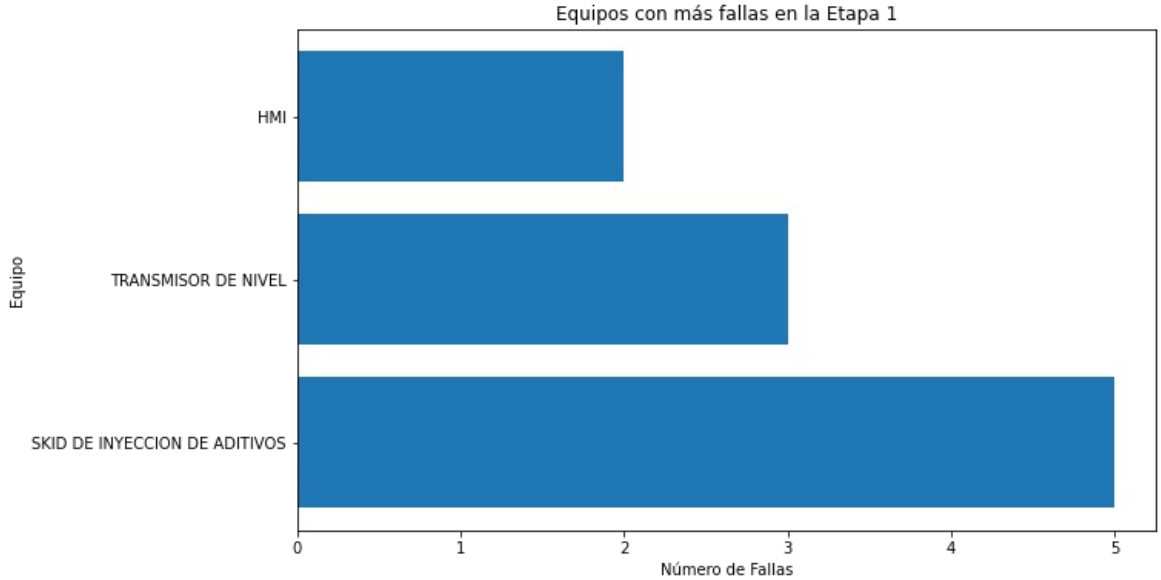

```
C:\Users\kevin\AppData\Local\Temp\ipykernel_137540\832426784.py:24: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  principales_fallas['proporcion'] = principales_fallas['total_fallas'] / total_fallas_equipo * 100
```

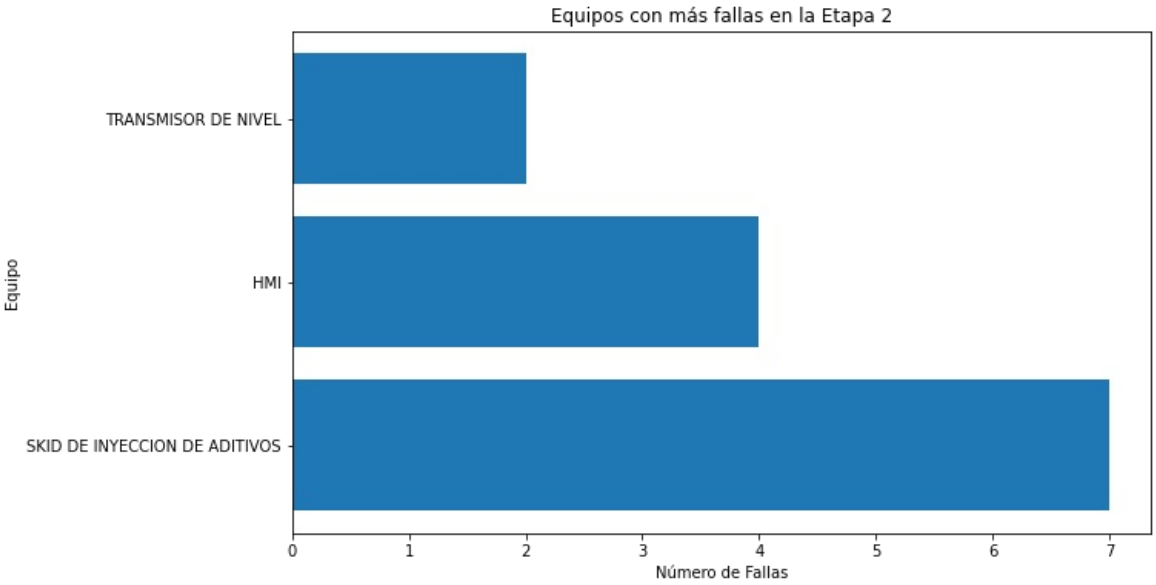


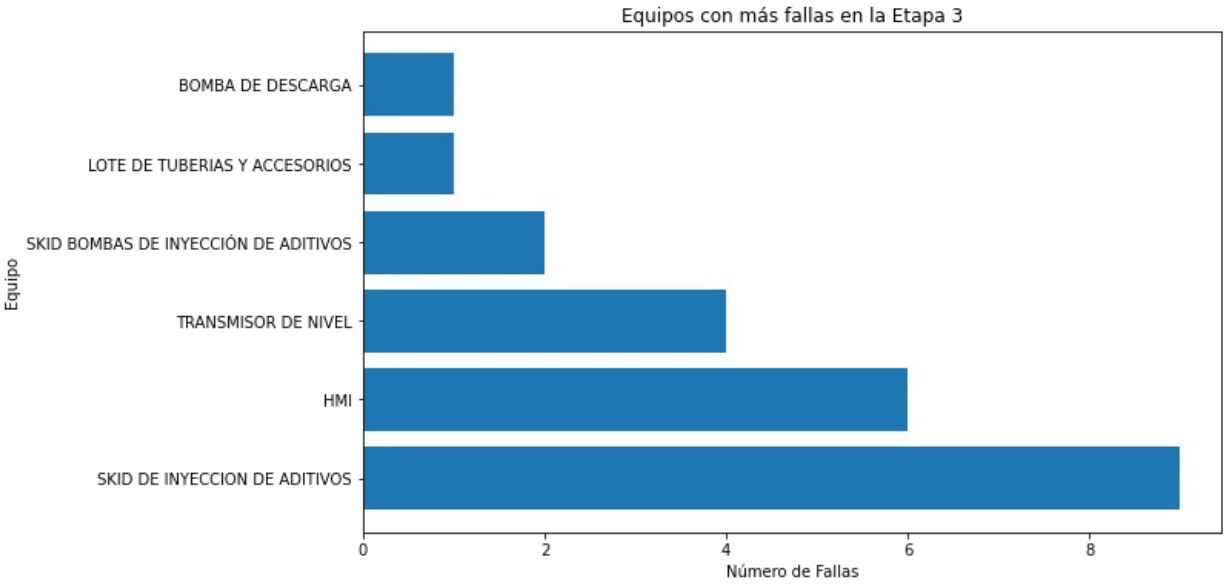

```
C:\Users\kevin\AppData\Local\Temp\ipykernel_137540\832426784.py:24: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  principales_fallas['proporcion'] = principales_fallas['total_fallas'] / total_fallas_equipo * 100
```

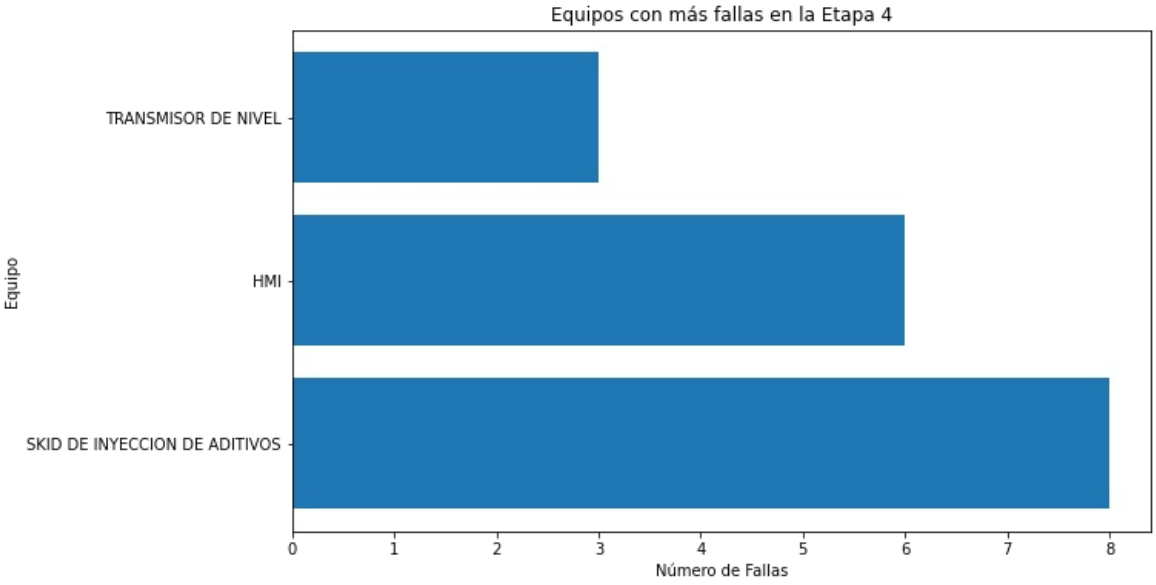


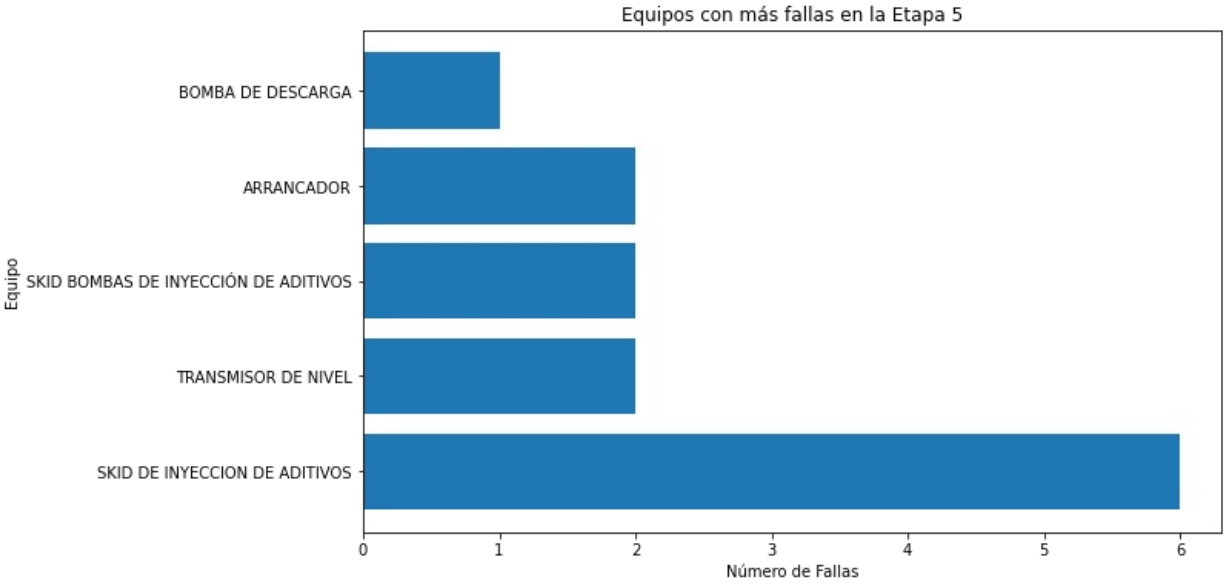
In [55]:
```
fallas_por_etapa_y_equipo = datos_completos.groupby(['NUM_ETAPA', 'NOMBRE_EQUIPO']).size().reset_index(name='to

# Ordena los resultados para cada etapa según el recuento de fallas
fallas_por_etapa_y_equipo_sorted = fallas_por_etapa_y_equipo.sort_values(by=['NUM_ETAPA', 'total_fallas'], asce

# Visualiza los resultados
for etapa in fallas_por_etapa_y_equipo_sorted['NUM_ETAPA'].unique():
    datos_etapa = fallas_por_etapa_y_equipo_sorted[fallas_por_etapa_y_equipo_sorted['NUM_ETAPA'] == etapa].head
    plt.figure(figsize=(10, 6))
    plt.barh(datos_etapa['NOMBRE_EQUIPO'], datos_etapa['total_fallas'])
    plt.xlabel('Número de Fallas')
    plt.ylabel('Equipo')
    plt.title(f'Equipos con más fallas en la Etapa {etapa}')
    plt.gca().invert_yaxis()  # Invierte el eje y para mostrar el equipo con más fallas arriba
    plt.show()
```

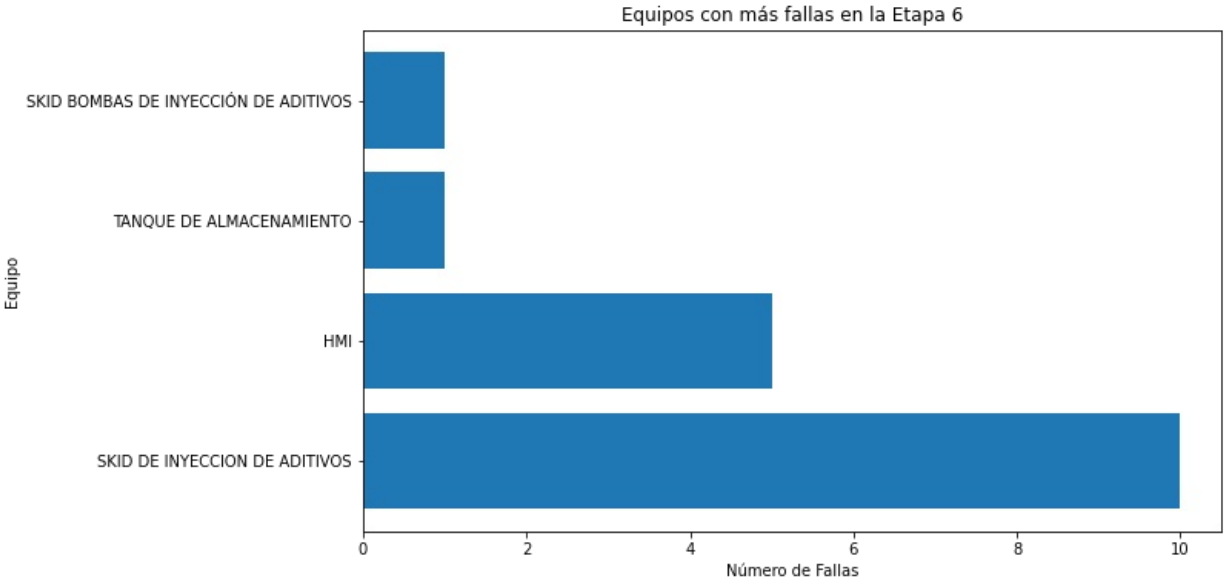


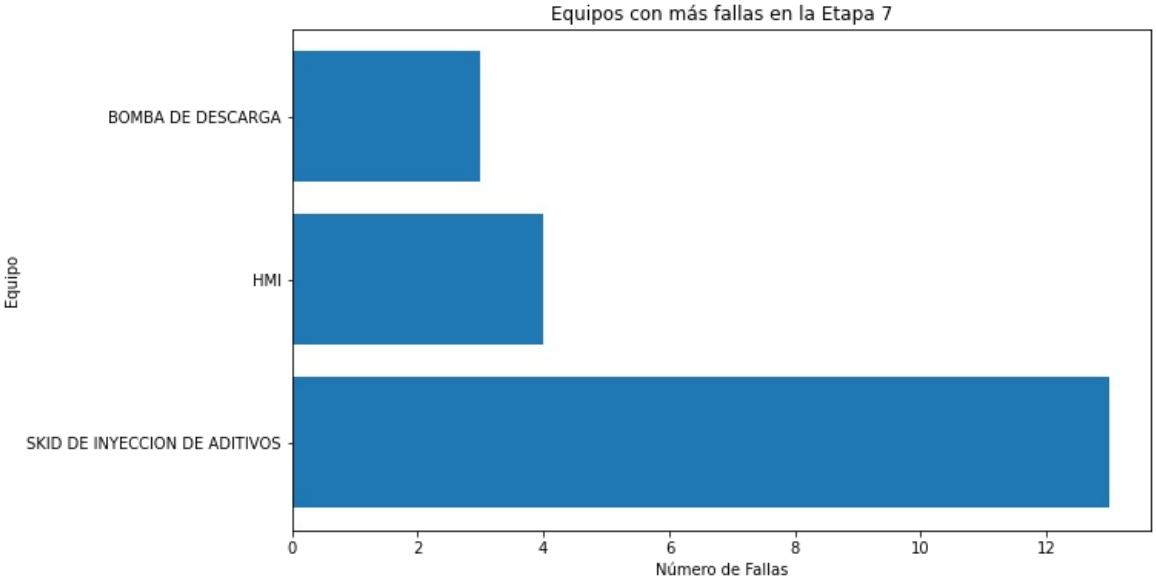

Equipos con más fallas en la Etapa 2
TRANSMISOR DE NIVEL
HMI
SKID DE INYECCION DE ADITIVOS
Equipo
0
1
2
3
4
5
6
7
Número de Fallas

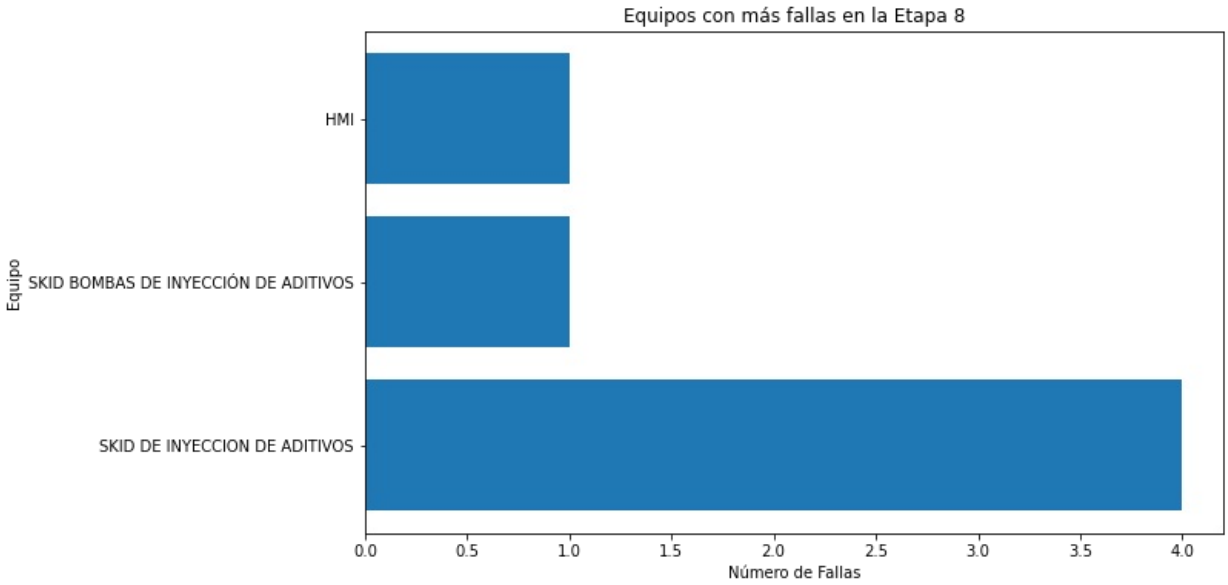

Equipos con más fallas en la Etapa 3
BOMBA DE DESCARGA
LOTE DE TUBERIAS Y ACCESORIOS
SKID BOMBAS DE INYECCIÓN DE ADITIVOS
TRANSMISOR DE NIVEL
HMI
SKID DE INYECCION DE ADITIVOS
Equipo
0
2
4
6
8
Número de Fallas


Equipos con más fallas en la Etapa 4
TRANSMISOR DE NIVEL
HMI
SKID DE INYECCION DE ADITIVOS
Equipo
0
1
2
3
4
5
6
7
8
Número de Fallas

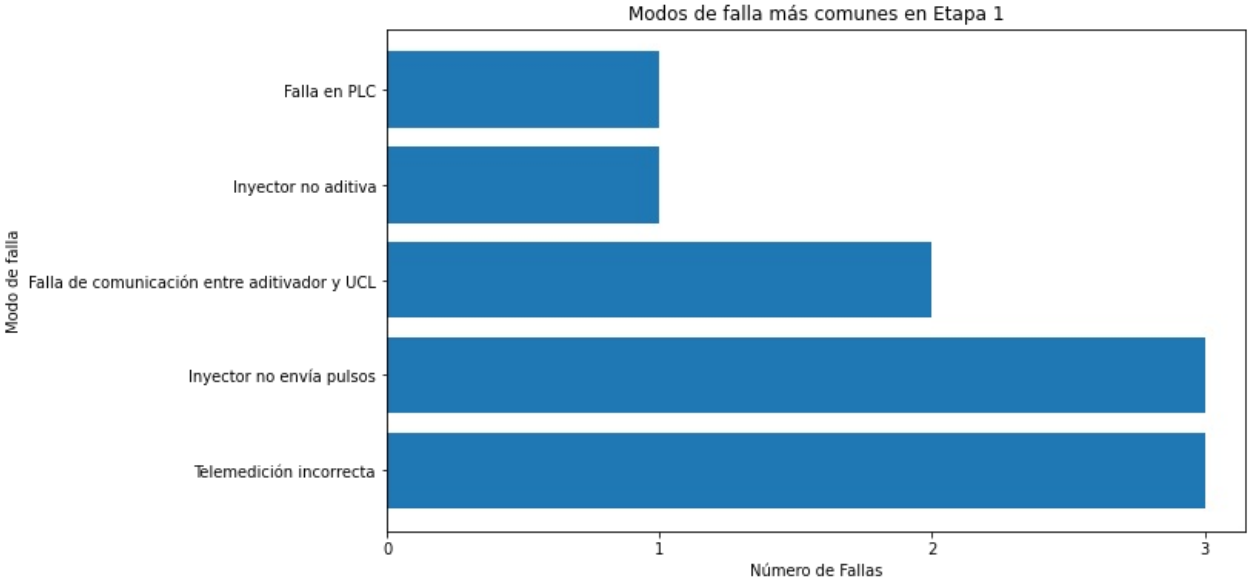

Equipos con más fallas en la Etapa 5
BOMBA DE DESCARGA
ARRANCADOR
SKID BOMBAS DE INYECCIÓN DE ADITIVOS
TRANSMISOR DE NIVEL
SKID DE INYECCION DE ADITIVOS
Equipo
0
1
2
3
4
5
6
Número de Fallas

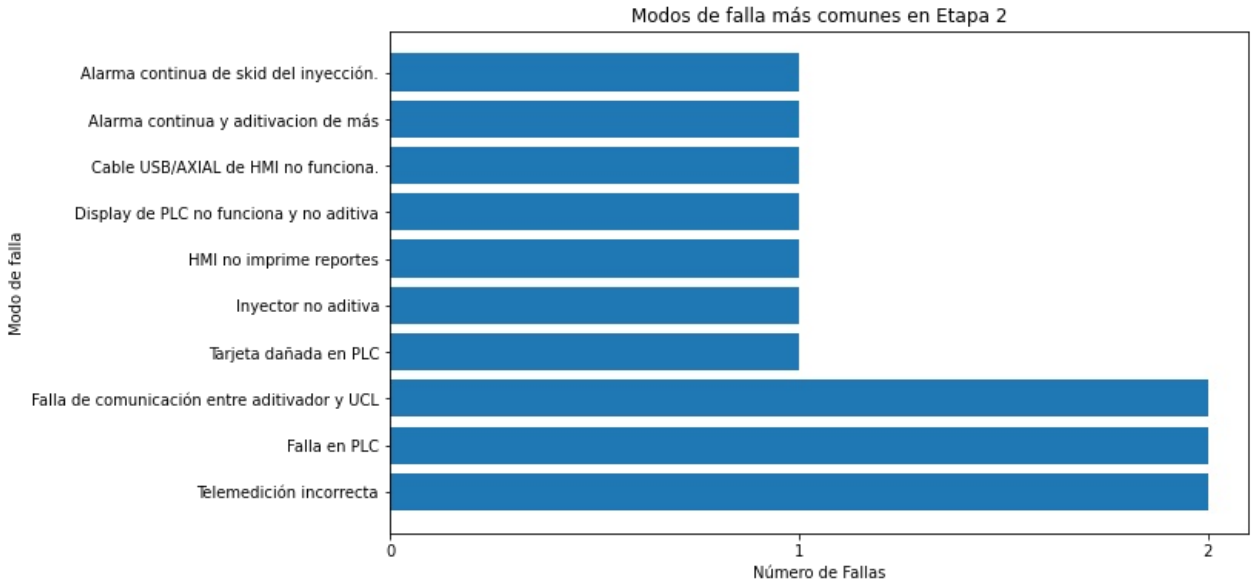

Equipos con más fallas en la Etapa 6
SKID BOMBAS DE INYECCIÓN DE ADITIVOS
TANQUE DE ALMACENAMIENTO
HMI
SKID DE INYECCION DE ADITIVOS
Equipo
0
2
4
6
8
10
Número de Fallas

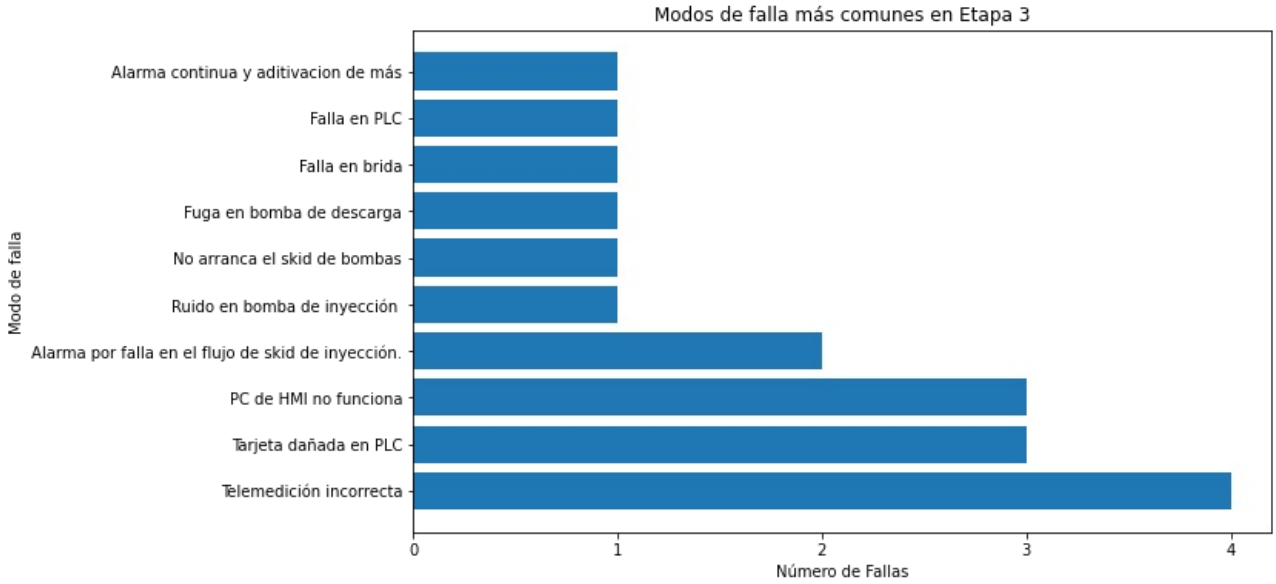

Equipos con más fallas en la Etapa 7
BOMBA DE DESCARGA
HMI
SKID DE INYECCION DE ADITIVOS
Equipo
0
2
4
6
8
10
12
Número de Fallas

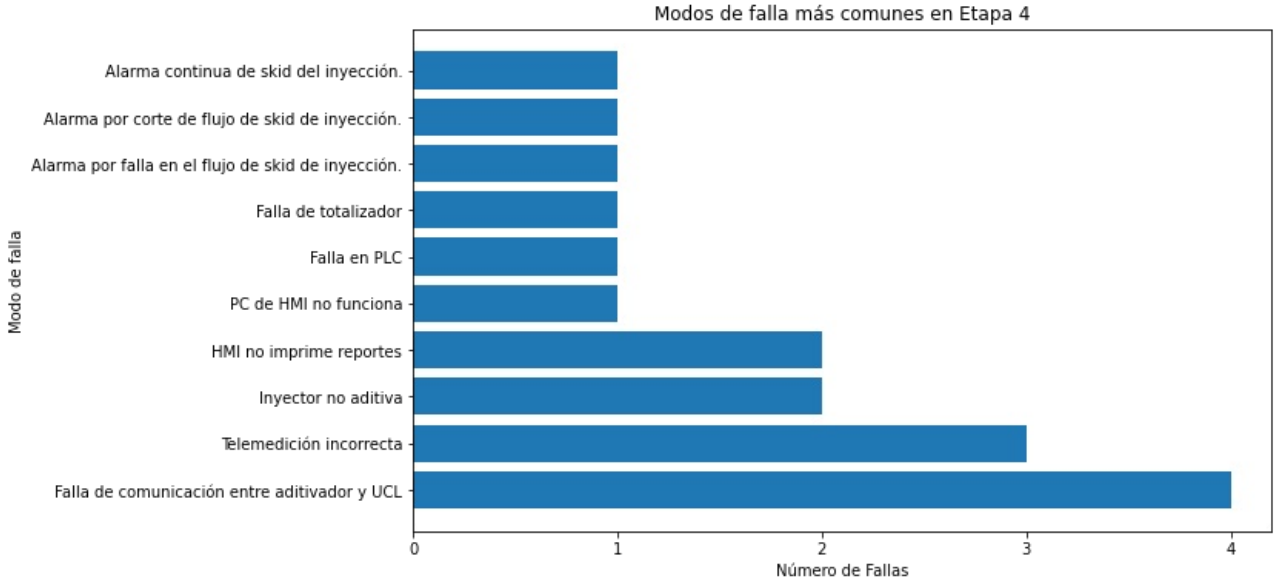


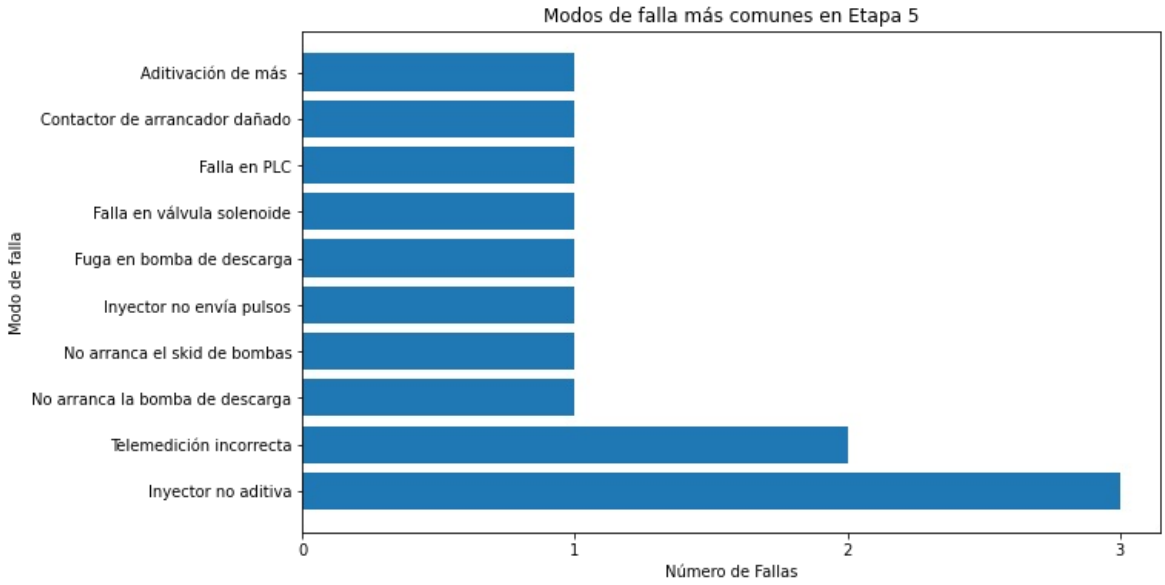
In [53]:
```
fallas_por_etapa_y_modo = datos_completos.groupby(['NUM_ETAPA', 'AVERÍA (MODO DE FALLA)']).size().reset_index(n

# Ordena los resultados para cada etapa según el recuento de fallas
fallas_por_etapa_y_modo_sorted = fallas_por_etapa_y_modo.sort_values(by=['NUM_ETAPA', 'total_fallas'], ascendin

# Visualiza los resultados
for etapa in fallas_por_etapa_y_modo_sorted['NUM_ETAPA'].unique():
    datos_etapa = fallas_por_etapa_y_modo_sorted[fallas_por_etapa_y_modo_sorted['NUM_ETAPA'] == etapa].head(10)
    plt.figure(figsize=(10, 6))
    plt.barh(datos_etapa['AVERÍA (MODO DE FALLA)'], datos_etapa['total_fallas'])
    plt.xlabel('Número de Fallas')
    plt.ylabel('Modo de falla')
    plt.title(f'Modos de falla más comunes en Etapa {etapa}')
    plt.gca().invert_yaxis()  # Invierte el eje y para mostrar el equipo con más fallas arriba
    plt.xticks(range(int(datos_etapa['total_fallas'].max()) + 1))
    plt.show()
```

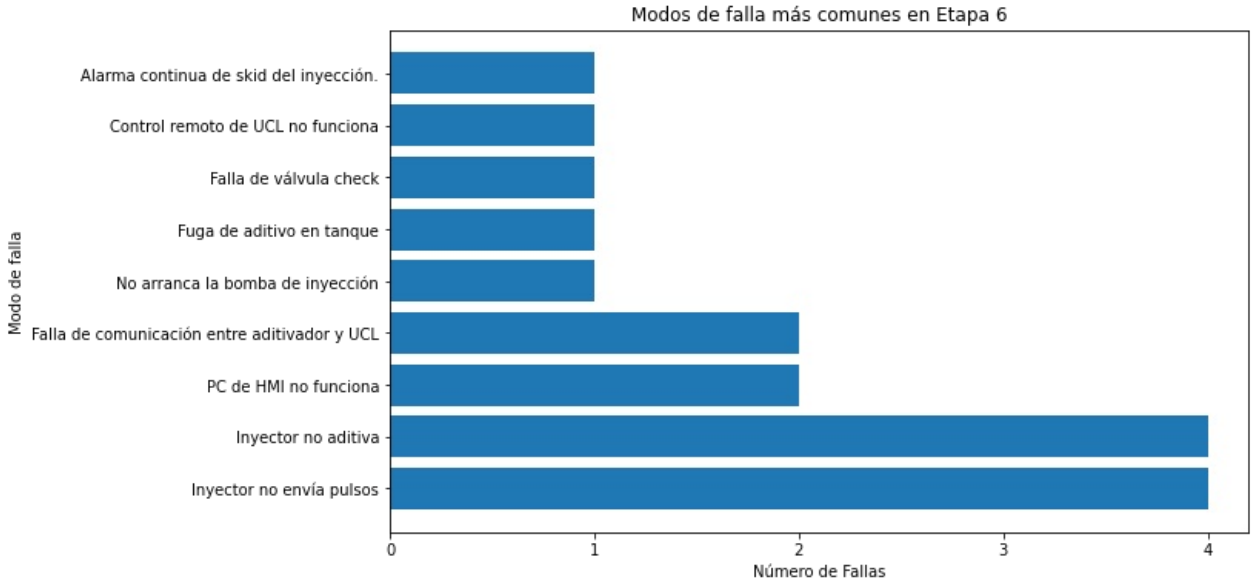


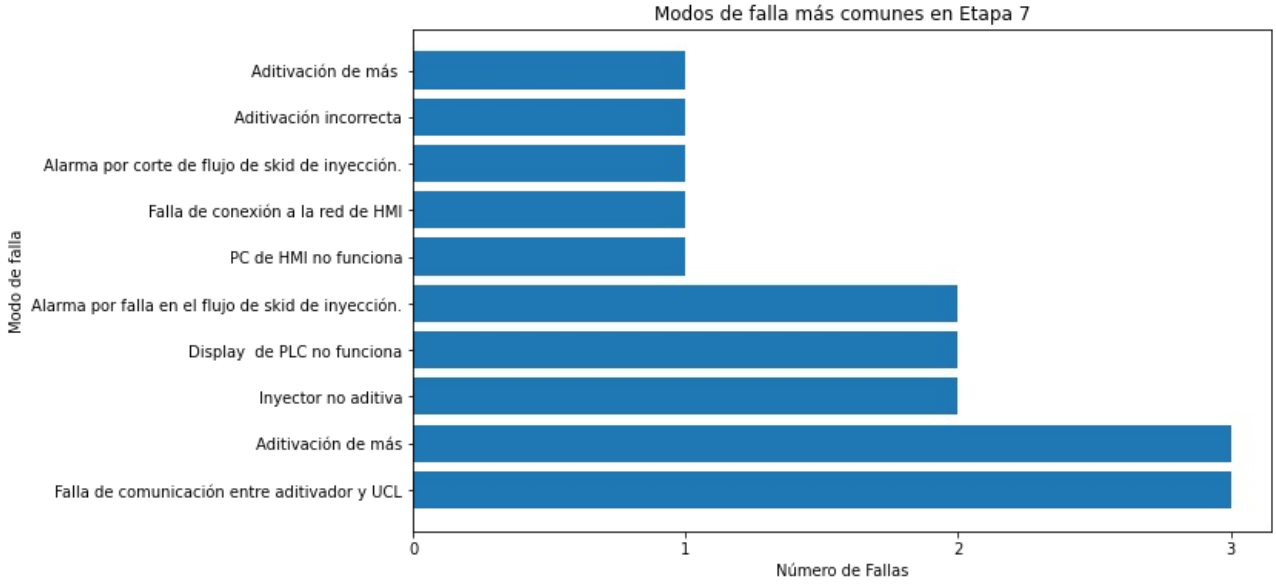

Modos de falla más comunes en Etapa 2
Alarma continua de skid del inyección.
Alarma continua y aditivacion de más
Cable USB/AXIAL de HMI no funciona.
Display de PLC no funciona y no aditiva
HMI no imprime reportes
Inyector no aditiva
Tarjeta dañada en PLC
Falla de comunicación entre aditivador y UCL
Falla en PLC
Telemedición incorrecta
Modo de falla
0
1
2
Número de Fallas

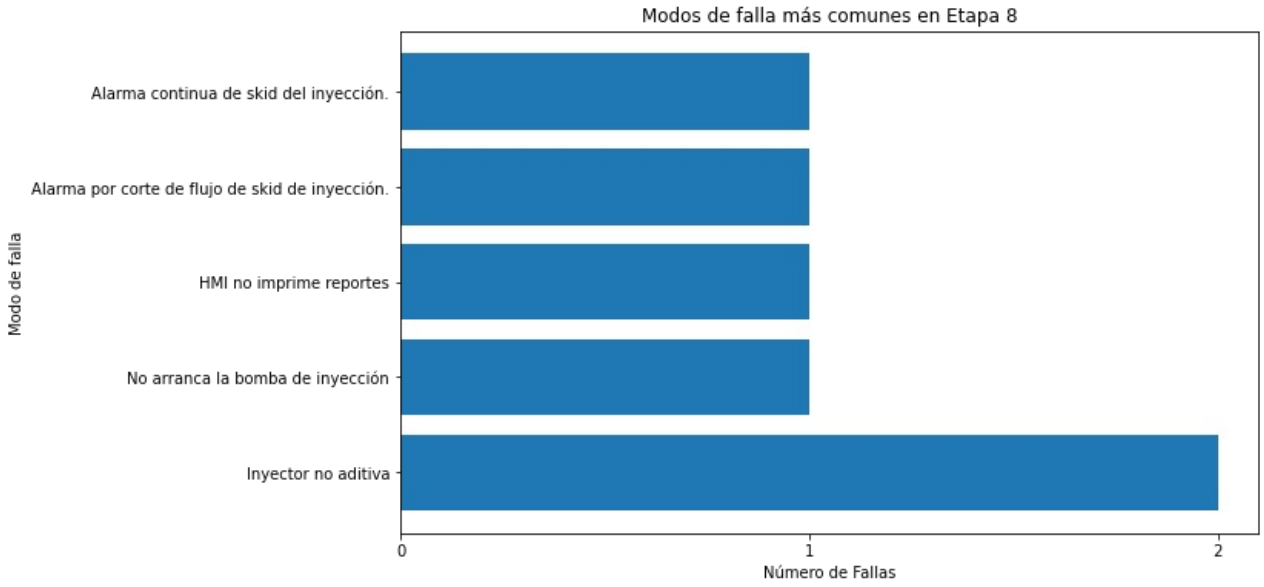

Modos de falla más comunes en Etapa 3
Alarma continua y aditivacion de más
Falla en PLC
Falla en brida
Fuga en bomba de descarga
No arranca el skid de bombas
Ruido en bomba de inyección
Alarma por falla en el flujo de skid de inyección.
PC de HMI no funciona
Tarjeta dañada en PLC
Telemedición incorrecta
Modo de falla
0
1
2
3
4
Número de Fallas


Modos de falla más comunes en Etapa 4
Alarma continua de skid del inyección.
Alarma por corte de flujo de skid de inyección.
Alarma por falla en el flujo de skid de inyección.
Falla de totalizador
Falla en PLC
PC de HMI no funciona
HMI no imprime reportes
Inyector no aditiva
Telemedición incorrecta
Falla de comunicación entre aditivador y UCL
Modo de falla
0
1
2
3
4
Número de Fallas


Modos de falla más comunes en Etapa 5
Aditivación de más
Contactor de arrancador dañado
Falla en PLC
Falla en válvula solenoide
Fuga en bomba de descarga
Inyector no envía pulsos
No arranca el skid de bombas
No arranca la bomba de descarga
Telemedición incorrecta
Inyector no aditiva
Modo de falla
0
1
2
3
Número de Fallas


Modos de falla más comunes en Etapa 6
Alarma continua de skid del inyección.
Control remoto de UCL no funciona
Falla de válvula check
Fuga de aditivo en tanque
No arranca la bomba de inyección
Falla de comunicación entre aditivador y UCL
PC de HMI no funciona
Inyector no aditiva
Inyector no envía pulsos
Modo de falla
0
1
2
3
4
Número de Fallas


Modos de falla más comunes en Etapa 7
Aditivación de más
Aditivación incorrecta
Alarma por corte de flujo de skid de inyección.
Falla de conexión a la red de HMI
PC de HMI no funciona
Alarma por falla en el flujo de skid de inyección.
Display de PLC no funciona
Inyector no aditiva
Aditivación de más
Falla de comunicación entre aditivador y UCL
Modo de falla
0
1
2
3
Número de Fallas



Inteligencia Artificial

Se presenta el modelo de *machine learning* que va a ayudar a realizar mantenimiento preventivo en las TAD.

In [ ]: