## appAnálise

Esse documento objetiva registrar os detalhes relacionados às principais funcionalidades implantadas no appAnálise, uma aplicação desenvolvida no Matlab.

As informações serão apresentadas nos seguintes tópicos:

1. **Funcionalidades** já implantadas na aplicação, com telas ilustrativas.
2. Forma de organização das principais **variáveis** na aplicação.
3. **Fluxos** de algumas tarefas na aplicação.
4. Da criação do projeto a partir do **GitHub**.

**Notas importantes!**

* A aplicação está sendo desenvolvida na versão **Matlab 2021a**.
* Em implementação, no modo FISCALIZA, a integração appAnálise/Fiscaliza, o que possibilitará a carga automática do relatório de monitoração no SEI e o preenchimento da inspeção do Fiscaliza, alterando o seu estado de “Rascunho” para “Relatada”.
* Em implementação funcionalidade que possibilita criação de máscara espectral, a qual será útil no âmbito do PMEC 2021.

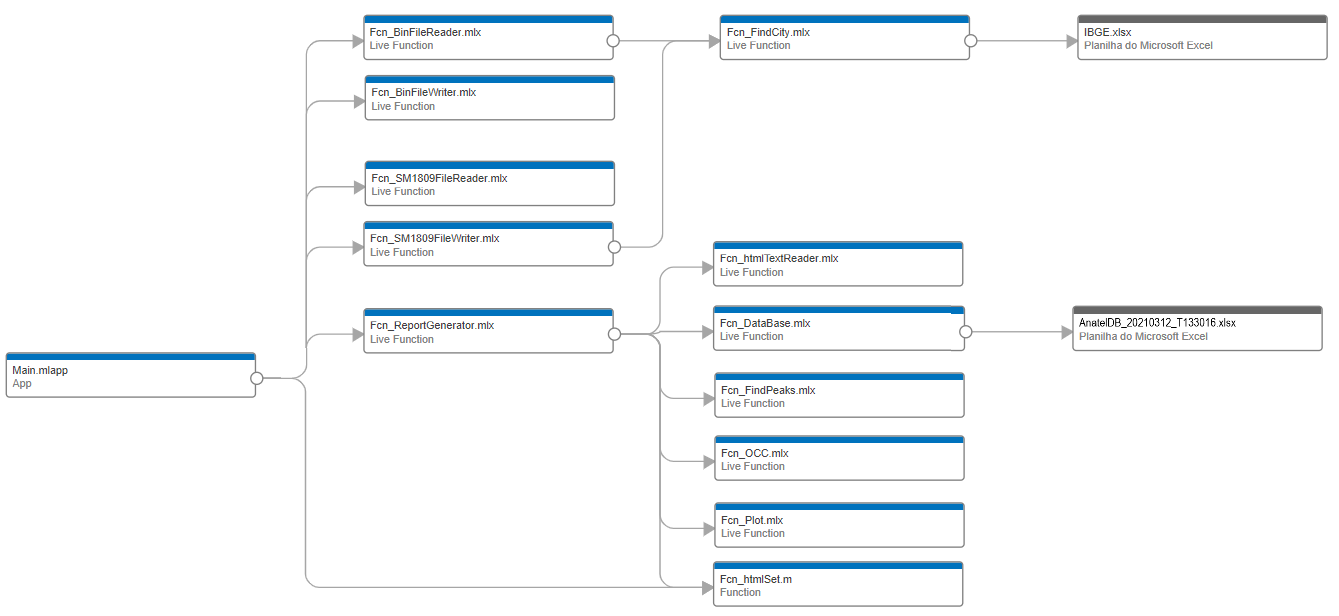
## Item 1: Funcionalidades já implantadas na aplicação, com telas ilustrativas.

Os modos já implantados na aplicação constam na tabela apresentada a seguir.

|  |  |  |  |
| --- | --- | --- | --- |
| **Modo** | **Layout** | **Principais funcionalidades** | **Principais funções e arquivos (1)** |
| ARQUIVO | Painéis 1 e 2.  Ambos podem ser omitidos para melhorar a visualização do Painel 3 (que pode possuir duas colunas, a depender do modo). | Leitura de arquivos nos formatos CRFS BIN, SM1809, ARGUS CSV e MAT.   * CRFS BIN, SM1809 e ARGUS CSV: arquivos de espectro, para os quais são lidos inicialmente os metadados (apresentados no Painel 1) e posteriormente os dados de espectro (apresentados no Painel 2). Quando da leitura dos dados de espectro, há uma prévia organização dos fluxos de dados (inclusive a relação com os fluxos de ocupação). * MAT: arquivos de “Projeto appAnálise”, relacionado ao processo de geração do relatório de monitoração que não pode ser finalizado numa única sessão. | **Fcn\_BinFileReader**  **Fcn\_SM1809FileReader**  **Fcn\_csvFileReader** |
| SALVAR | Painel 2 | Possibilita salvar fluxos de dados selecionados em arquivos nos formatos CRFS BIN e SM1809 com as seguintes ressalvas:   * Os fluxos devem estar relacionados a um mesmo sensor porque a informação de GPS é única (ou num bloco comum a todos os fluxos, no caso do CRFS BIN, ou no cabeçalho, no caso do SM1809). * O arquivo no formato SM1809 não aceita fluxo de ocupação. * É obrigatória a informação do GPS nos fluxos de dados selecionados. | **Fcn\_BinFileWriter**  **Fcn\_SM1809FileWriter** |
| EDIÇÃO | Painel 3. Uma única coluna. | Edição de coordenadas geográficas, quando não presentes no(s) arquivo(s).  Edição do fluxo de ocupação que será considerado no modo RELATÓRIO, em especial na busca de emissões na curva de MaxHold (Critério 2). | - |
| PLAYBACK | Painel 3 e figura.  Como o painel possui duas colunas, **recomenda-se omitir o Painel 1**. | Visualização tanto dos fluxos de espectro quanto dos fluxos de ocupação.  Processamento dos traços (média e MaxHold, por exemplo), identificação automática de picos, aferição em “tempo real” da ocupação e *Waterfall*. | Fcn\_Plot(app)  Fcn\_Plot\_Layout1(app)  Fcn\_Plot\_Layout2(app)  Fcn\_Plot\_StartUp1(app)  Fcn\_Plot\_StartUp2(app) |
| RELATÓRIO | Painel 3, uma figura por fluxo e um visualizador HTML.  Como o painel possui duas colunas, **recomenda-se omitir o Painel 1**. | Geração de relatório.  Se não for possível concluir a análise das emissões indicadas no relatório, submetendo-o ao SEI, então o fiscal pode salvar o “Projeto appAnálise” no formato MAT. | **Fcn\_ReportGenerator**  **Fcn\_OCC**  **Fcn\_FindPeaks**  **Fcn\_DataBase**  **Fcn\_Plot**  “AnatelDB\_YYYYMMDD\_THHMMSS.xlsx” (incluso na subpasta “DataBase”)  Arquivos textuais inclusos na subpasta “Template”, os quais correspondem aos pedaços do relatório que será gerado no formato HTML. |

(1) Em negrito “funções externas” à aplicação.

A seguir é apresentado diagrama com relação entre os principais arquivos do projeto.



A seguir a relação de produtos que precisam estar instalados para que todas as funcionalidades estejam operacionais:

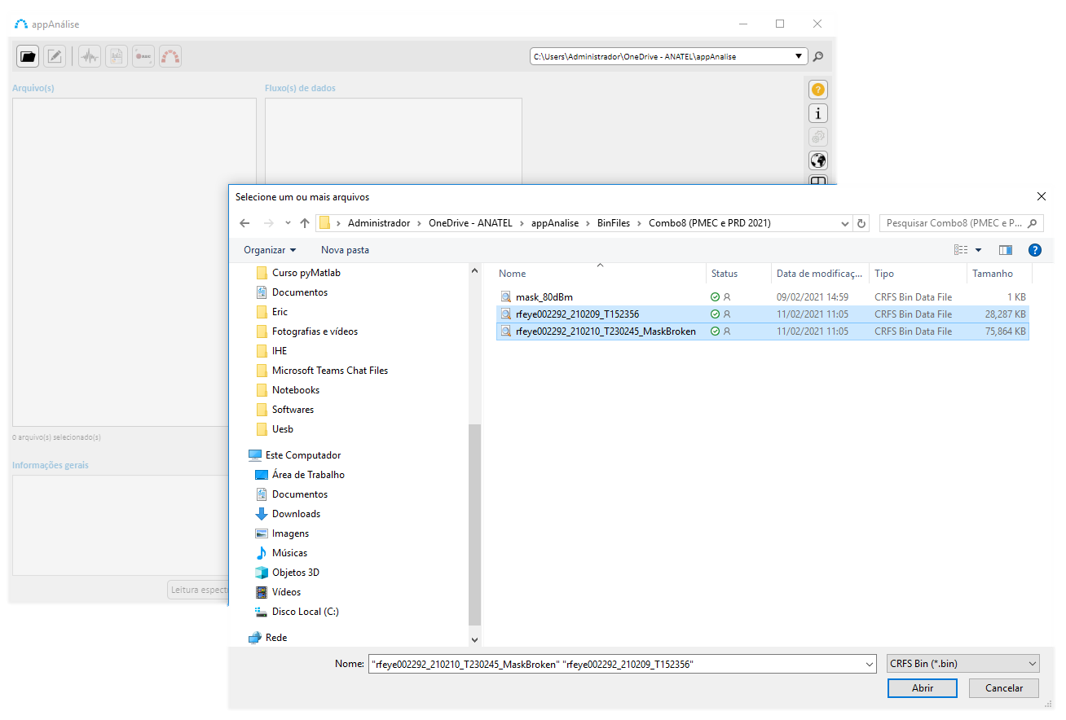
**②**

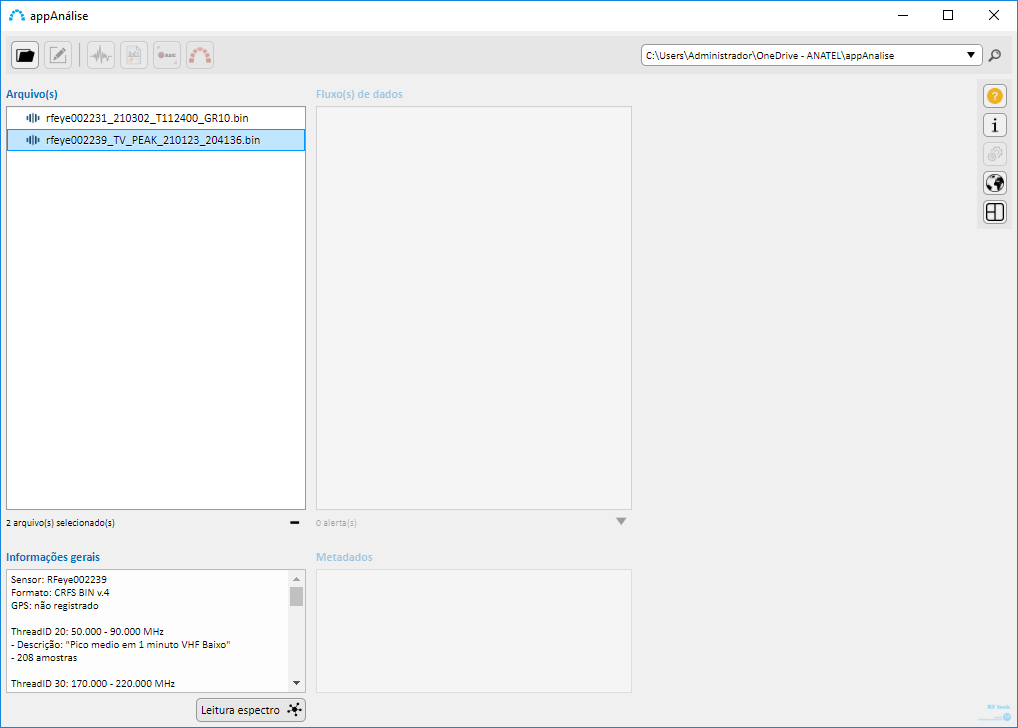
* MATLAB 2020b
  + Signal Processing Toolbox
  + Image Processing Toolbox
  + MATLAB Report Generator
* Python 3.8

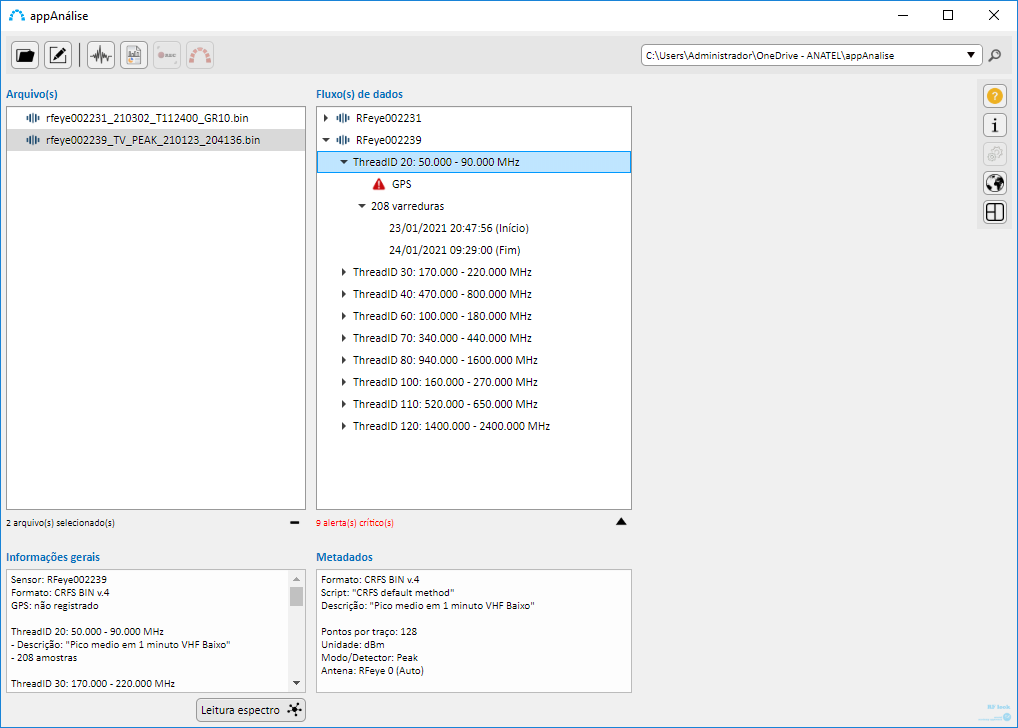
|  |  |
| --- | --- |
| **⑥**  **①**  **⑤**  **④**  **③** | |
| **①** Menu gráfico que possibilita a escolha do modo ARQUIVO, SALVAR, EDIÇÃO, PLAYBACK ou RELATÓRIO.  **②** Pasta onde serão salvos os arquivos (relatórios, imagens, dados espectrais etc).  **③** Barra de ferramentas, com destaque para as seguintes funcionalidades:   * Define as principais variáveis da aplicação como variáveis globais, possibilitando a sua visualização na área de trabalho do Matlab. * Possibilita alteração do layout da aplicação, tornando invisível os paineis 1 e 2, o que melhora a visualização do Painel 3. | **④** “Painel 1”, no qual consta a relação de arquivos cujos metadados foram lidos. Ao selecionar um dos arquivos são apresentados algumas das suas informações na caixa de texto inferior.  **⑤** “Painel 2”, no qual consta a relação de fluxos de dados (tanto fluxos de espectro quanto fluxos de ocupação). Ao selecionar um dos fluxos são apresentadas algumas das suas informações na caixa de texto inferior.  **⑥** “Painel 3”, no qual consta as funcionalidades implantadas para os modos EDIÇÃO, PLAYBACK e RELATÓRIO. |

**②**

### Modo **ARQUIVO**.





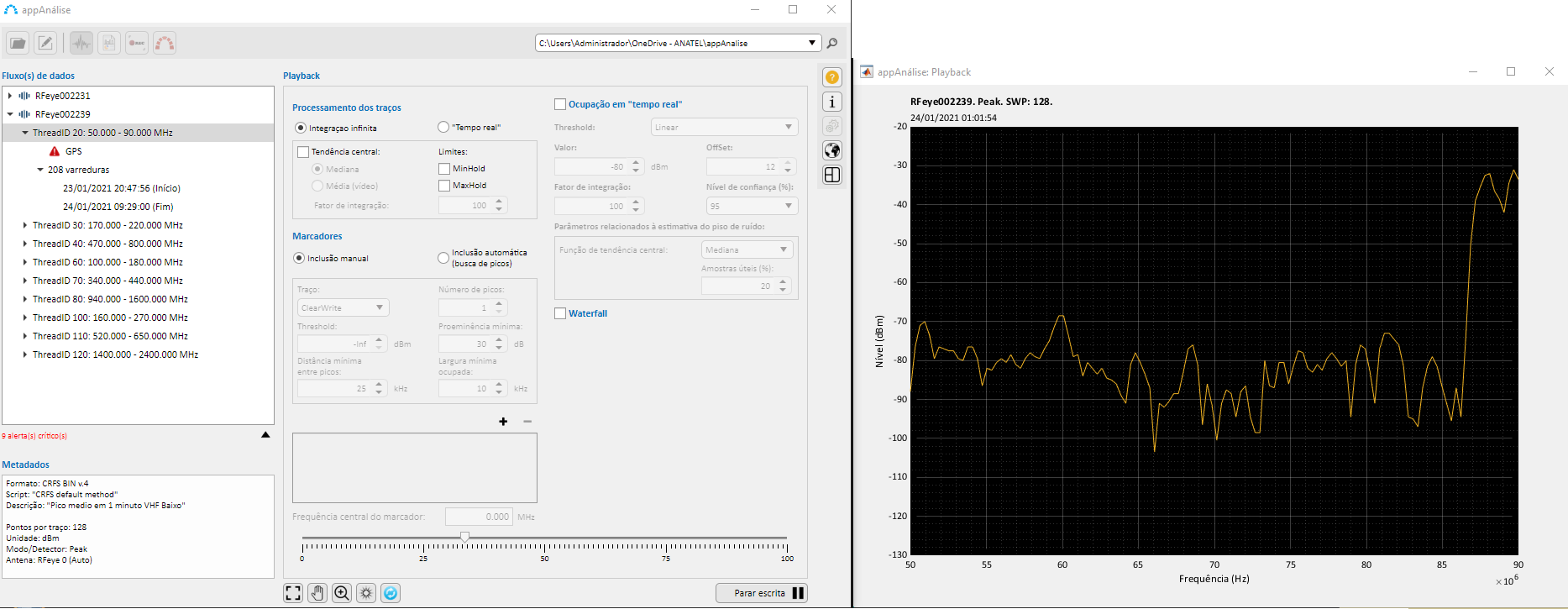


### Modo **PLAYBACK**.

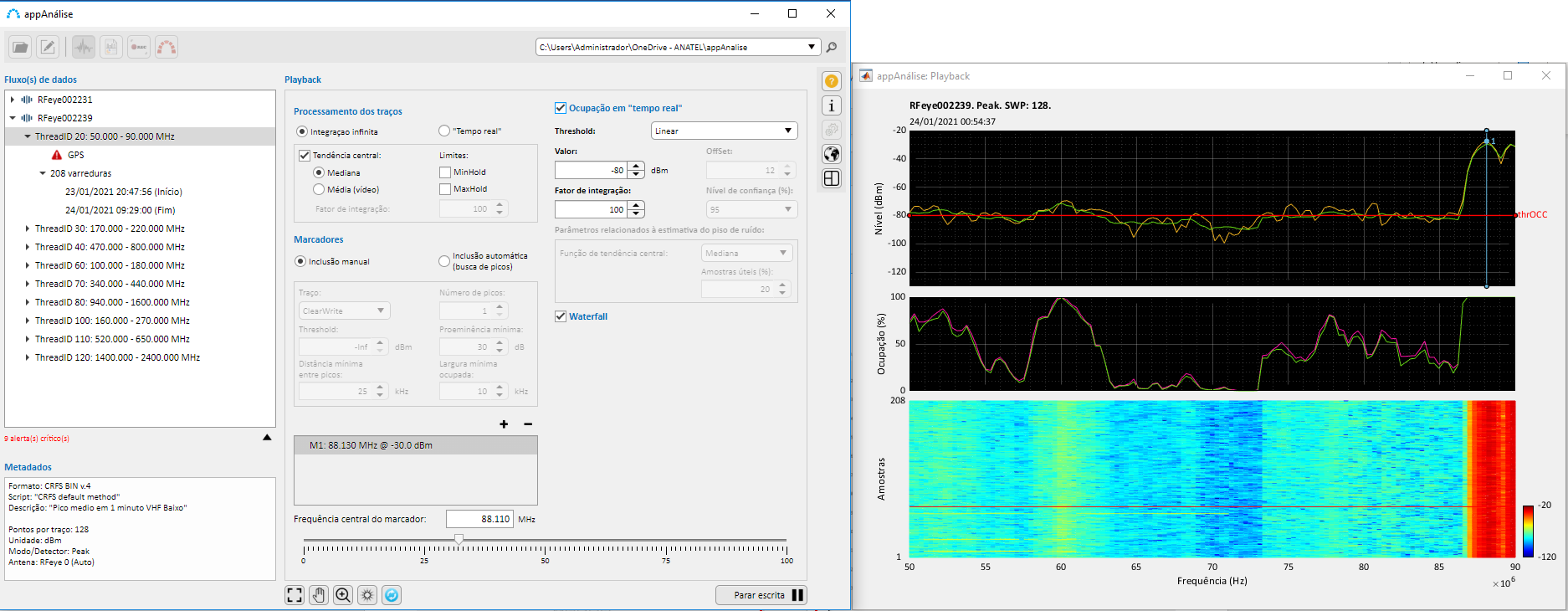
Nas telas apresentadas a seguir é possível visualizar todas as funcionalidades implantadas no modo, o que somente foi possível após tornar invisível o Painel 1.

Para que seja possível visualizar os dados dos fluxos de espectro ou ocupação, faz-se necessária prévia seleção de um fluxo de dados na árvore.

As funcionalidades “Processamento dos traços” e “Ocupação” são aplicáveis apenas aos fluxos de espectro, ficando desabilitadas quando selecionado um fluxo de ocupação.



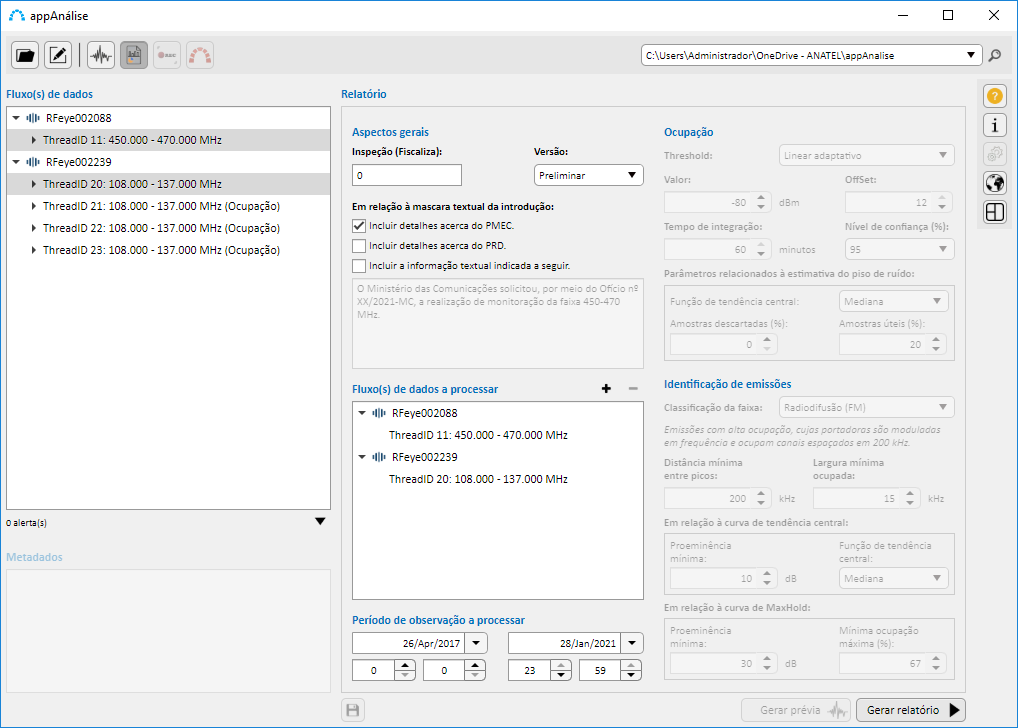
Botão que ajusta a visibilidade dos painéis 1 e 2.

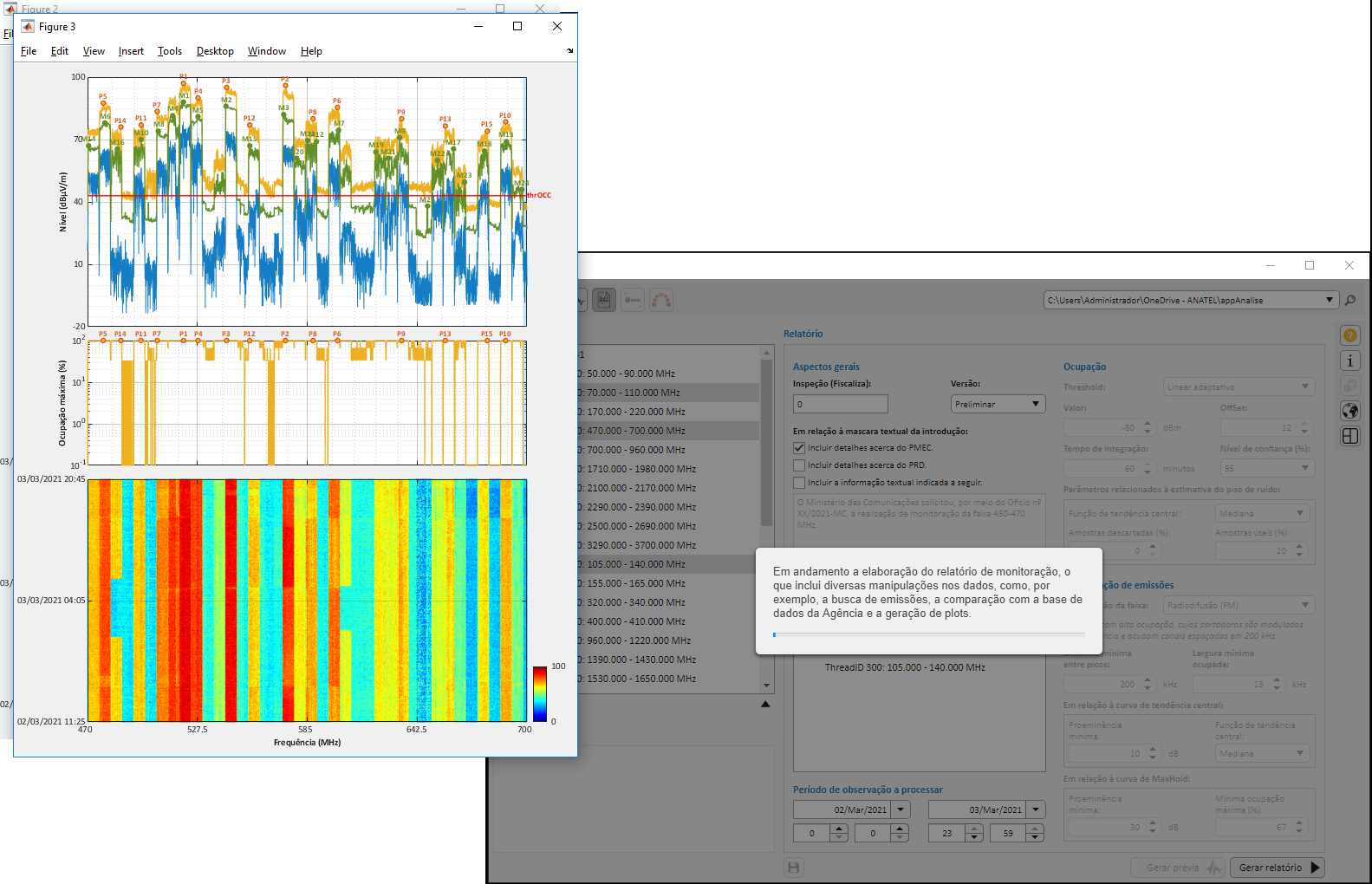


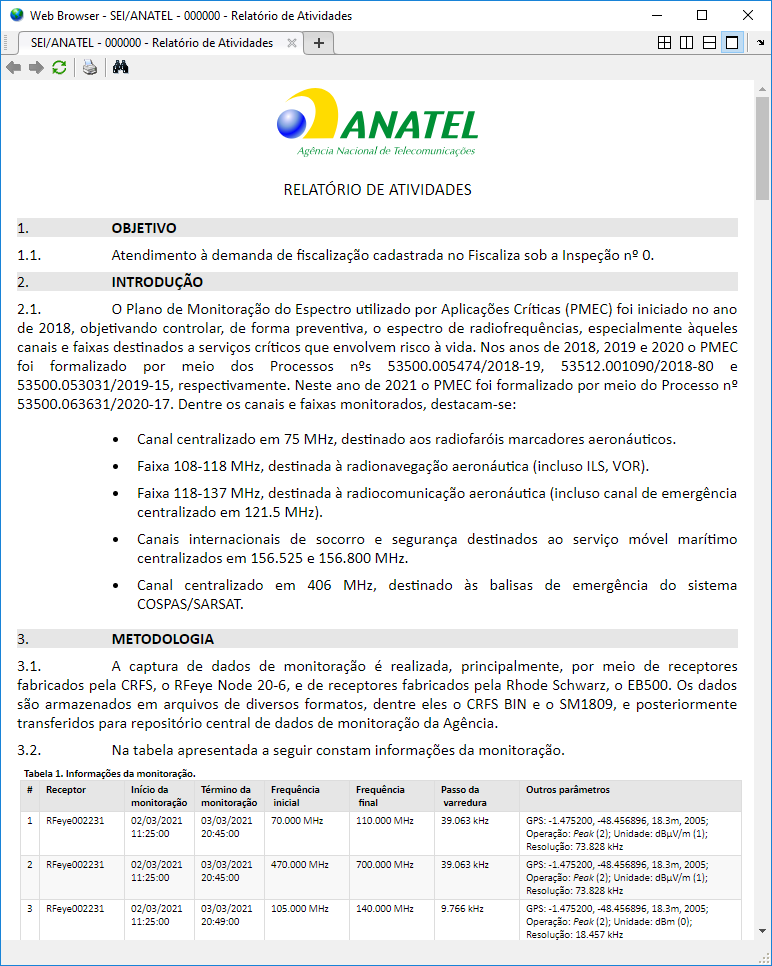
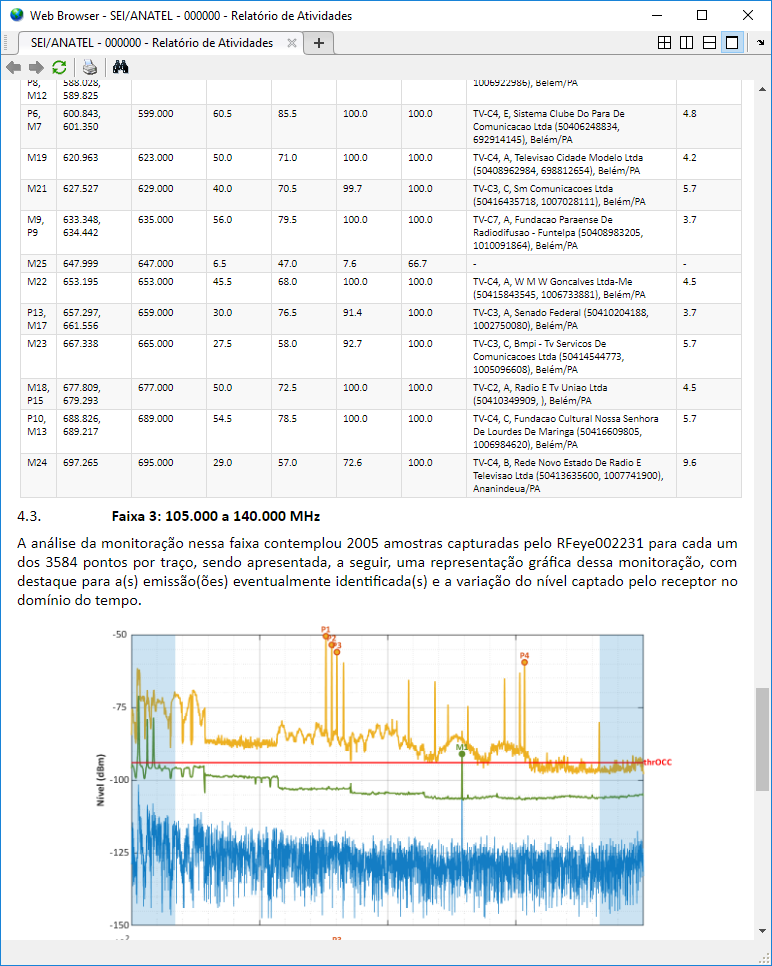
### Modo **RELATÓRIO**.

Para que seja possível gerar um relatório, deve ser inserido um ou mais fluxos de espectro usando a imagem **+**, que fica sobre a árvore do Painel 3. A imagem **–** possibilita a exclusão de um ou mais fluxos.

A multiseleção da árvore do Painel 2 somente é possível nesse modo. Nos outros modos, por conta da sua natureza, é desativada a multiseleção.





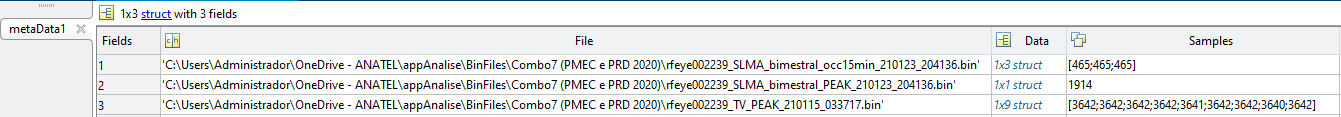


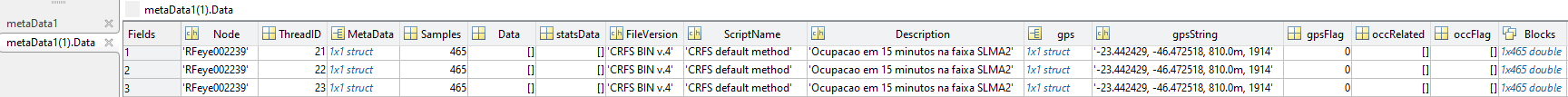
## Item 2: Forma de organização das principais variáveis na aplicação.

A seguir são descritas as principais variáveis da aplicação.

**metaData**

metaData = struct('File', [], 'Data', [], 'Samples', [])



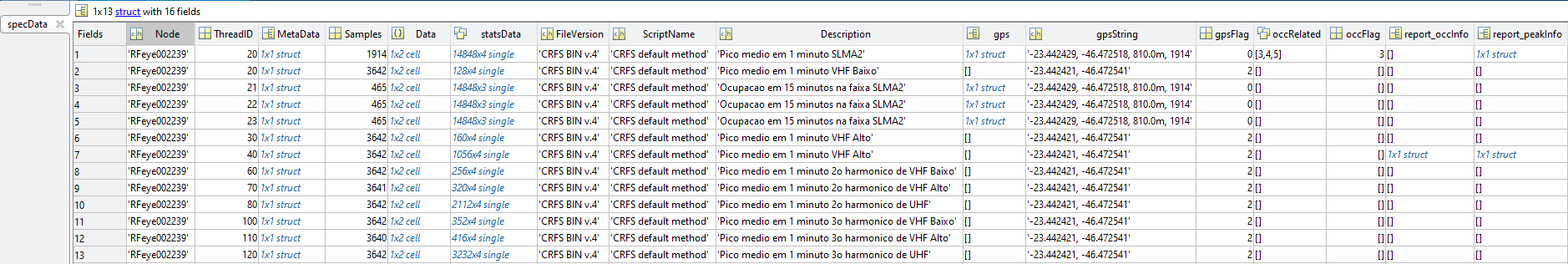


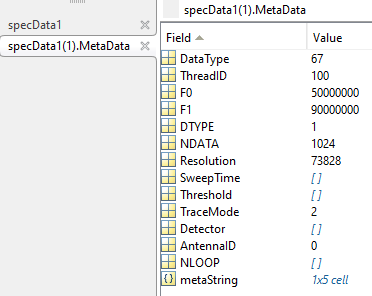
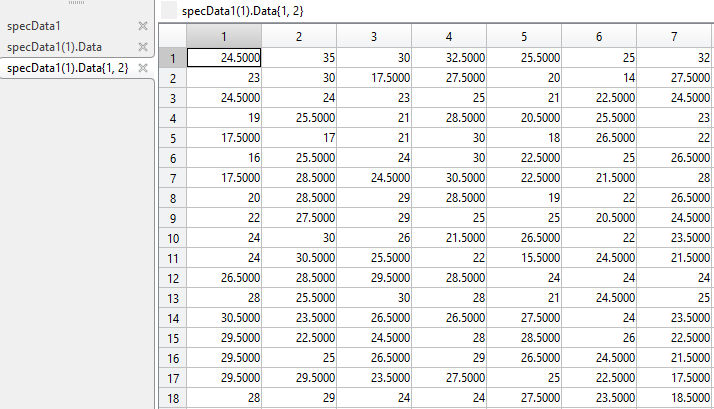
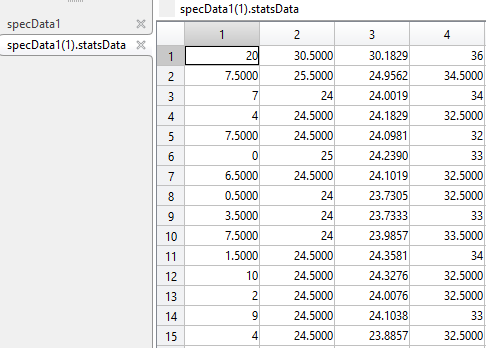
Notas:

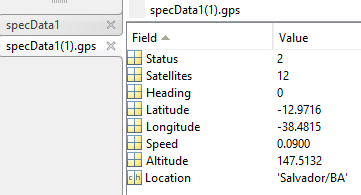
* O campo “Samples” armazena o número de traços por fluxo. Logo, considerando a imagem que exemplifica a organização da variável metaData, perceba que essa informação corresponde à leitura de três arquivos. O primeiro arquivo tem três fluxos (de espectro ou ocupação), sendo que cada fluxo possui 465 traços.

**specData**

specData = struct('Node', [], 'ThreadID', [], 'MetaData', [], 'Samples', [], 'Data', [], 'statsData', [], 'FileVersion', [], 'ScriptName', [], 'Description', [], 'gps', [], 'gpsString', [], 'gpsFlag', [], 'occRelated', [], 'occFlag', [], 'report\_occInfo', [], 'report\_peakInfo', []);





Notas:

* Caso se trate de um arquivo CRFS BIN, a função Fcn\_BinFileReader criará a estrutura specData, editando as informações dos campos ‘Node’, ‘ThreadID’, ‘MetaData’, ‘Samples’, ‘Data’, ‘FileVersion’, ‘ScriptName’, ‘Description’, ‘gps’, ‘gpsString’, ‘gpsFlag’.

|  |  |
| --- | --- |
| **Campo** | **Possíveis valores** |
| gpsFlag | Se o arquivo possuir informação válida de GPS, então o valor desse campo será igual a zero.  Caso contrário, o valor será igual a 1, podendo ser editada na aplicação, quando assumirá o valor igual a 2. |

* A aplicação, posteriormente, ao analisar os fluxos de vários arquivos, editará as informações dos campos ‘occRelated’ e ‘occFlag’ da seguinte forma:

|  |  |
| --- | --- |
| **Campo** | **Possíveis valores** |
| occRelated | Um vetor de índices de fluxos de ocupação que estão relacionados ao fluxo de espectro.  Por exemplo, na imagem que exemplifica a organização da variável specData consta que o fluxo de espectro da Linha 1 possui occRelated = [3,4,5]. Ou seja, os fluxos de ocupação das Linhas 3, 4 e 5 possuem as mesmas características do fluxo de espectro de Linha 1.  Caso não existe fluxo de ocupação relacionado ao de espectro, o valor desse campo será vazio.  Por exemplo,  specData(i).occRelated = [];  ou  specData(i).occRelated = [2,3,4]; |
| occFlag | Escolha do fluxo de ocupação que deverá ser considerado quando da busca por emissões na curva de MaxHold (Critério 2), no modo RELATÓRIO.   * Modo automático: é escolhido o primeiro fluxo de ocupação indicado no campo occRelated.   specData(i).occFlag = specData(i).occRelated(1);   * Modo manual: é escolhido fluxo de ocupado diverso do primeiro indicado no campo occRelated, ou, opcionalmente, indicado que a ocupação será aferida no pós-processamento.   specData(i).occFlag = [];  ou  specData(i).occFlag = specData(i).occRelated(j);  com j ≠ 1. |

* A aplicação criará outros dois campos – ‘report\_occInfo’ e ‘report\_peakInfo’ -, os quais somente serão editados para aqueles fluxos que forem inclusos no modo RELATÓRIO.

SEÇÃO DE COISAS ALEATÓRIAS QUE AINDA PRECISAM SER ORGANIZADAS.

* Unidade:

switch app.specData(app.traceInfo.SelectedNode).MetaData.DTYPE

case 0; app.Unit = 'dBm';

case 1; app.Unit = 'dBμV/m';

end

* Detector p/ outros equipamentos:

switch app.specData(app.traceInfo.SelectedNode).MetaData.Detector

case 0; Detector = 'AutoPeak';

case 1; Detector = 'PositivePeak';

case 2; Detector = 'NegativePeak';

case 3; Detector = 'Sample';

case 4; Detector = 'RMS';

case 5; Detector = 'Average';

case 6; Detector = 'QuasiPeak';

end

* Trace mode p/ RFeye

switch app.specData(app.traceInfo.SelectedNode).MetaData.TraceMode

case 0; traceMode = 'Single Measurement';

case 1; traceMode = 'Mean';

case 2; traceMode = 'Peak';

case 3; traceMode = 'Minimum';

end

* Trace mode p/ outros equipamentos:

switch app.specData(app.traceInfo.SelectedNode).MetaData.TraceMode

case 0; traceMode = 'ClearWrite';

case 1; traceMode = 'Average';

case 2; traceMode = 'MaxHold';

case 3; traceMode = 'MinHold';

end

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metadado |  |  |  |  |  |
| ThreadID |  |  |  |  |  |
| F0 |  |  |  |  |  |
| F1 |  |  |  |  |  |
| Resolution |  |  |  |  |  |
| Duration |  |  |  |  |  |
| TraceMode / Processing |  |  |  |  |  |
| DTYPE |  |  |  |  |  |
|  |  |  |  |  |  |
| FileVersion |  |  |  |  |  |
| ScriptName |  |  |  |  |  |
| Description |  |  |  |  |  |

Algoritmos a citar:

* Leitura de GPS.
* Processamento de traços (Média, MinHold e MaxHold).
* Ocupação em tempo real (Threshold fixo e OffSet do Piso de Ruído).
* Busca de picos.

Principais fluxos:

* Função Fcn\_ReportGenerator.
* Função Fcn\_Plot(app).

Peguei aquele arquivo que Lobão usou inicialmente p/ avaliar o tempo de leitura dos algoritmos em C++, Python e Matlab (SCAN\_M\_450470\_rfeye002088\_170426\_171908.bin). O arquivo possui 39MB.

Li o arquivo, registrando-o na specData (a variável do appAnalise). Depois escrevi essa variável num CRFS BIN (v.5) e num SM1809. E li esses novos arquivos.

|  |  |  |  |
| --- | --- | --- | --- |
| Formato | Tamanho do arquivo | Tempo de escrita | Tempo de leitura |
| CRFS BIN (v.5) | 39 MB | 1.5 s | 1.7 s |
| SM1809 | 232 MB | 129 s | 535 s |
| MAT v.7.3 (HDF5) | 39 MB | 3.4 s | 0.6 s |
| MAT (v.7) | 39 MB | 8 s | 0.7 s |

Inserir posteriormente às seguintes funcionalidades do THRESHOLD p/ aferição de ocupação na própria aplicação.

- Linear fixo

- Piso de ruído (Offset)

|  |  |  |
| --- | --- | --- |
| **Funcionalidade** | **Estado inicial** | **Movimentos excepcionais** |
| Processamento dos traços |  |  |
| Marcadores | app. Playback\_peakLabel0.Enable = 1;  app. Playback\_peakRadio1.Value = 1;  set(app. Playback\_peakRadioGroup.Children, ‘Enable’, 1)  set(app. Playback\_peakGrid.Children, ‘Enable’, 0) | Se **fluxo de ocupação** selecionado na Tree2:  app. Playback\_peakLabel0.Enable = 0;  set(app. Playback\_peakRadioGroup.Children, ‘Enable’, 0)  set(app. Playback\_peakGrid.Children, ‘Enable’, 0) |
| Ocupação em “tempo real” | app. Playback\_occCheckBox.Value = 0  set(app. Playback\_occGrid1.Children, ‘Enable’, 0)  set(app. Playback\_occGrid2.Children, ‘Enable’, 0) | Se **fluxo de ocupação** selecionado na Tree2:  set(app.Playback\_occCheckBox, ‘Enable’, 0, ‘Value’, 0)  set(app. Playback\_occGrid1.Children, ‘Enable’, 0)  set(app. Playback\_occGrid2.Children, ‘Enable’, 0) |
| Waterfall | app. Playback\_wfCheckBox.Value = 0 |  |

Rfeye:  
IF   
DataType in [62, 65, 69]: "OCC"  
ELSE  
0: "single measurement"  
1: "average"  
2: "peak"  
3: "minimum"  
  
Para os outros sensores:  
0: 'ClearWrite'  
1: 'Average'  
2: 'MaxHold'  
3: 'MinHold'

-1': Não posssui informação.  
0: 'AutoPeak'  
1: 'PositivePeak'  
2: 'NegativePeak'  
3: 'Sample'  
4: 'RMS'  
5: 'Average'  
6: 'QuasiPeak'

]

**rfLOOK BIN v.1**

|  |  |  |  |
| --- | --- | --- | --- |
| **File Header**  *String* com 16 caracteres, cada um representado por um byte.  **“rfLOOK BIN v.1 ”** | **DataType1**  **Data Block Header**  **Data Block Body**  Node ScriptName  Description  F0, F1  Resolution  SweepTime  SweepPoints  revisitTime  integrationTime  Threshold  TraceMode  Detector  Unit  Preamp  ATT  AntennaID  **Data Block Trailer** | **DataType3**  **Data Block Header**  **Data Block Body**  Type TimeStamp  Status  Heading  Latitude  Longitude  Speed  **Data Block Trailer** | **DataType5**  **Data Block Header**  **Data Block Body**  TimeStamp  Offset  Array of levels  **Data Block Trailer** |

**DataType1: Metadados**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parte** | **Campo** | **Representação** | **Comentários** |
| Data Block Header | ThreadID | uint16 |  |
| BytesBlock | uint32 |  |
| DataType | uint16 | 1 |
| Data Block Body | Node | *string* com 32 caracteres (1 byte por caractere) |  |
| ScriptName | *string* com 32 caracteres (1 byte por caractere) |  |
| Description | *string* com 32 caracteres (1 byte por caractere) |  |
| F0\_int | uint16 (parte inteira) | F0 em MHz  F0 = F0\_int + F0\_dec / 1000 |
| F0\_dec | uint32 (parte decimal) |
| F1\_int | uint16 (parte inteira) | F1 em MHz  F1 = F1\_int + F1\_dec / 1000 |
| F1\_dec | uint32 (parte decimal) |
| Resolution | uint32 | Resolução em Hertz |
| SweepTime | uint32 | Tempo de varredura em μs |
| SweepPoints | uint16 | Pontos por traço  Faixa de possíveis valores: 0-65535 pontos  Obs.: o EB500, o FSL e o FSVR possuem número máximo de 32001 pontos. |
| revisitTime | uint32 | Tempo de revisita em ms |
| integrationTime | uint32 | Tempo de integração em segundos  Faixa de possíveis valores: 0-49710 dias |
| Threshold | int16 | Se Threshold = -1: Threshold = []  Caso contrário: Threshold = Threshold - 1000 |
| TraceMode | int8 | Se TraceMode = 0: “ClearWrite”  Se TraceMode = 1: “Average”  Se TraceMode = 2: “MaxHold”  Se TraceMode = 3: “MinHold” |
| Detector | int8 | Se Detector = 0: “AutoPeak”  Se Detector = 1: “PositivePeak”  Se Detector = 2: “NegativePeak”  Se Detector = 3: “Sample”  Se Detector = 4: “RMS”  Se Detector = 5: “Average” |
| Unit | uint8 | Se Unit = 0: “dBm”  Se Unit = 1: “dBμV”  Se Unit = 2: “dBμV/m” |
| Preamp | uint8 | Se Preamp = 1: “Ligado”  Se Preamp = 0: “Desligado” ou “Inexistente” |
| ATT | uint8 | 1 ou 0 |
| AntennaID | uint8 | Se AntennaID = 0: “RFeye 0 (Auto)”  Se AntennaID = 1: “RFeye 1”  Se AntennaID = 2: “RFeye 2”  Se AntennaID = 3: “RFeye 3”  Se AntennaID = 4: “RFeye 4”  Se AntennaID = 5: “EB500 0 (Auto)”  Se AntennaID = 6: “R&S ADD107”  Se AntennaID = 7: “R&S ADD207”  Se AntennaID = 8: “Telescópica”  ... |
| - | 10 bytes de alinhamento e reserva para inclusão de outros campos. |  |
| Data Block Trailer | CheckSum | uint32 |  |
| EOF | *string* “UUUU”, a qual possui 4 caracteres, sendo representada por 1 byte por caractere) | “UUUU” |

**DataType3: GPS**

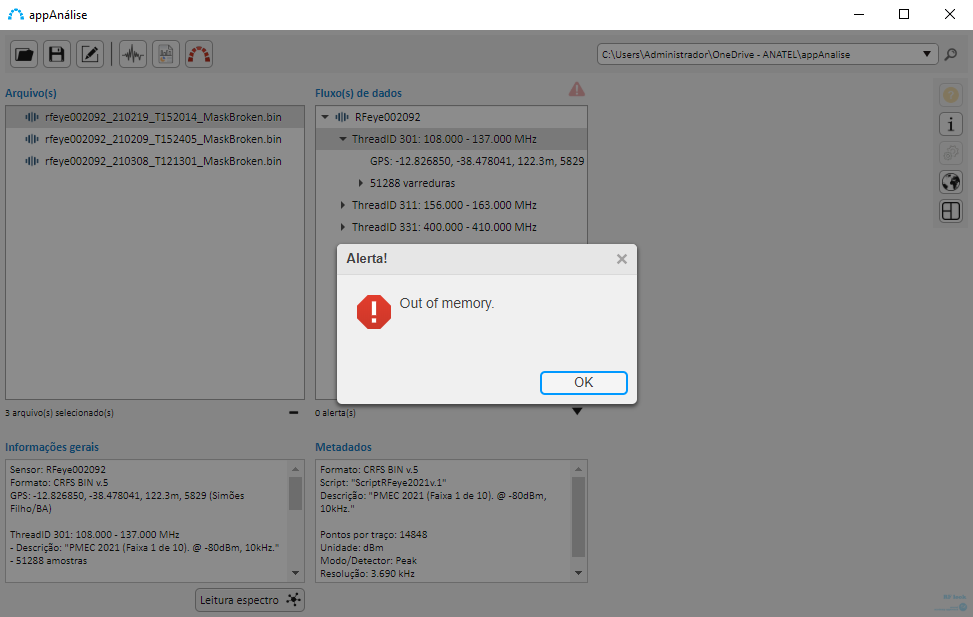
|  |  |  |  |
| --- | --- | --- | --- |
| Data Block Header | ThreadID | uint16 |  |
| BytesBlock | uint32 |  |
| DataType | uint16 | 3 |
| Data Block Body | Type | unit8 | Se Type = 0: inclusão das informações manual  Se Type = 1: inclusão das informações automática por meio de um dispositivo GPS |
| TimeStamp (Day) | uint8 | Se Type = 0: horário local  Se Type = 1: horário UTC |
| TimeStamp (Month) | uint8 |  |
| TimeStamp (Year) | uint8 | Subtraído 2000. |
| TimeStamp (Hours) | uint8 |  |
| TimeStamp (Minutes) | uint8 |  |
| TimeStamp (Seconds) | uint8 |  |
| TimeStamp (CentiSeconds) | uint8 | Multiplicado por 100. |
| Status | uint8 |  |
| - | 1 byte de alinhamento |  |
| Heading | uint16 | Multiplicado por 100. |
| Latitude | uint32 | Multiplicado por 106, possibilita manutenção de seis casas decimais do formato graus decimais. |
| Longitude | uint32 |
| Speed | uint32 | Multiplicado por 1000. |
| Data Block Trailer | CheckSum | uint32 |  |
| EOF | *string* “UUUU”, a qual possui 4 caracteres, sendo representada por 1 byte por caractere) | “UUUU” |

**DataType5: Dados de espectro para monitorações fixas.**

|  |  |  |  |
| --- | --- | --- | --- |
| Data Block Header | ThreadID | uint16 |  |
| BytesBlock | uint32 |  |
| DataType | uint16 | 5 |
| Data Block Body | TimeStamp (Day) | uint8 | Horário local |
| TimeStamp (Month) | uint8 |  |
| TimeStamp (Year) | uint8 | Subtraído 2000. |
| - | 1 byte de alinhamento |  |
| TimeStamp (Hours) | uint8 |  |
| TimeStamp (Minutes) | uint8 |  |
| TimeStamp (Seconds) | uint8 |  |
| TimeStamp (CentiSeconds) | uint8 | Multiplicado por 100. |
| OFFSET | uint8 |  |
| Array of levels | Cada bin representado por um 1 byte (uint8) | 2\*(specData - OFFSET) + 255 |
| Array of zeros (padding) | Entre 1 e 3 zeros, cada um representado por 1 byte. |  |
| Data Block Trailer | CheckSum | uint32 |  |
| EOF | *string* “UUUU”, a qual possui 4 caracteres, sendo representada por 1 byte por caractere) | “UUUU” |

.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metadado** | | **Formato de arquivo** | | |
| **CRFS BIN** | **SM1809** | **CSV** |
| Node | | Extraída do campo “Hostname”, parte do bloco DataType 21. | Inexistente. Caso gerado a partir do appAnálise poderá ser extraída do campo opcional "Node". | Extraída do campo "Meas. Unit", presente na última coluna do CSV |
| ThreadID | | Extraída do campo “Data thread identifier”, parte do cabeçalho de todos os tipos de blocos (DataType 21, 40, 60, 61 etc). | Inexistente. Caso gerado a partir do appAnálise poderá ser extraída do campo opcional "ThreadID". | Inexistente. Inserido valor igual a "1". |
| MetaData | DataType | Extraída do campo “Data type”, parte do cabeçalho de todos os tipos de blocos espectrais (DataType 60, 61, 62, 63, 64, 65, 67, 68 e 69). | Inexistente. Adotado “1809”. | Inexistente. Adotado 167 quando no campo "Measurement Result Type" estiver indicado “Measurement Result”, e 168 quando no supracitado campo estiver indicado “Compressed Measurement Result”. |
| F0 | Na v.3 do CRFS BIN,   * Extraída dos campos “START” e “FREQEXP”.   A partir da v.4,   * Extraída dos campos “STARTMEGA” e “STARTMILLI”. | Extraída do campo "FreqStart". | Trata-se do campo “Minimum Frequency”, presente na última coluna do CSV. |
| F1 | Na v.3 do CRFS BIN,   * Extraída dos campos “STOP” e “FREQEXP”.   A partir da v.4,   * Extraída dos campos “STOPMEGA” e “STOPMILLI”. | Extraída do campo "FreqStop". | Extraída do campo “Maximum Frequency”, presente na última coluna do CSV. |
| DTYPE | 0: dBm, 1: dBuV/m. | Extraída do campo "LevelUnits". | Extraída da primeira linha do CSV. |
| Resolution | Apenas na v.5 do CRFS BIN. | Extraída do campo "FilterBandwidth" | Extraída do campo “IF Bandwidth”, presente na última coluna do CSV. |
| Duration |  | Extraída do campo "ScanTime". | Pode ser extraída do campo “Meas. Time”, caso não esteja no modo “Default”.  Caso esteja no modo “Default”, adota-se:   * DataType 167: extraída do campo “Compress Time Interval”. * DataType 168: aferida média da diferença entre *TimeStamps* consecutivos. |
| Threshold | Existente apenas nos fluxos de OCC (62, 65 e 69) e de espectrais comprimidos (61, 64 e 68). | - | - |
| TraceMode |  | Não se trata de campo obrigatório, mas se tiver sido gerado a partir do appAnálise poderá ser identificada a informação no campo opcional "TraceMode". | Inexistente. Adota-se:   * DataType 167: “Average” * DataType 168: “ClearWrite” |
| Detector | Não se trata de campo obrigatório, mas se tiver sido gerado a partir do appAnálise ou appColeta poderá ser identificado. | Extraída do campo "Detector". | Extraída do campo “Detector”, com o seguinte mapeamento:   * "Fast" é escrito como "Sample". |
| AntennaID |  | Extraída do campo "AntennaType". |  |
| FileVersion | | "CRFS BIN v.3"  "CRFS BIN v.4"  "CRFS BIN v.5" | "SM1809 v.2" | “Argus Measurement Result”  “Argus Compressed Measurement Result” |
| ScriptName | | Informação "method" inserido no script Logger. Ou informação textual, caso gerado por sensor diferente do Rfeye. | Não se trata de campo obrigatório, mas se tiver sido gerado a partir do appAnálise poderá ser identificada a informação no campo opcional "ScriptName". |  |
| Description | | Informação “description” inserida no script, em cada um dos fluxos de dados, e extraída do DataType 24 (até a v.4 do CRFS BIN) ou dos próprios blocos de espectro (v.5 do CRFS BIN). | Não se trata de campo obrigatório, mas se tiver sido gerado a partir do appAnálise poderá ser identificada a informação no campo opcional "Description". Não recomendado uso do campo opcional "Note" porque outros decodificadores não individualizarão a descrição de cada um dos fluxos, caso se trate de um registro multiscan. |  |
| gps | Status |  |  |  |
| Satellites |  |  |  |
| Heading |  |  |  |
| Latitude |  |  |  |
| Longitude |  |  |  |
| Speed |  |  |  |
| Altitude |  |  |  |

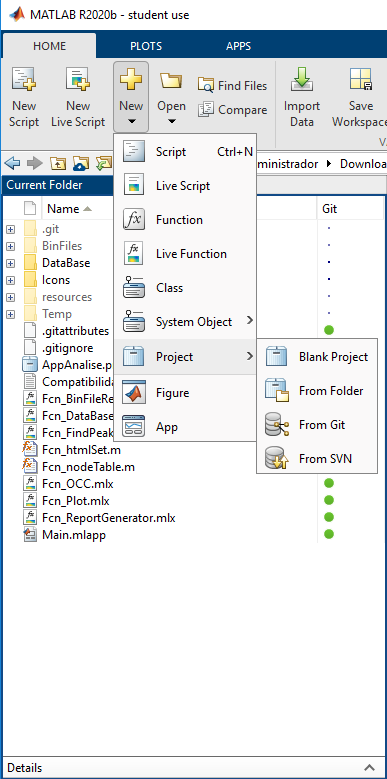


A depender da quantidade de arquivos inseridos no appAnálise e do número de traços e pontos de varredura, pode ser dado erro “Out of memory”. Esse estouro de memório é mais comum quando da leitura de arquivos comprimidos do CRFS BIN, os quais podem ter, por exemplo, 150 mil traços, cada um com 15 mil pontos por varredura.

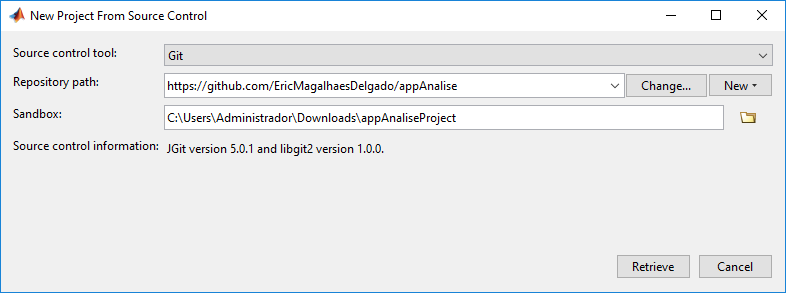
FIM DA SEÇÃO DE COISAS ALEATÓRIAS QUE AINDA PRECISAM SER ORGANIZADAS.

## Item 4: Da criação do projeto a partir do GitHub.

**Passo 1**: Cria um projeto a partir do GitHub.



**Passo 2**: Cria uma pasta que funcionará como repositório local, além de indicar o link pro projeto do GitHub - <https://github.com/EricMagalhaesDelgado/appAnalise>



**Passo 3**: Executar o arquivo Main.mlapp, acessando a opção Run com o botão direito do mouse.

