**Alterações no Main.py**

Após o algoritmo verificar se existe gateway foi adicionado ao bloco lógico responsável por realizar a conexão com o banco uma condicional que verifica se e software foi recentemente atualizado, caso tenha sido faz a chamada do método **self.gate.carregaEvento** para que o banco de dados offline seja preenchido com os eventos e propriedades correspondentes a uma unidade.

**Alterações no atualizador.py**

Anteriormente a essa versão do **atualizador.py** o sistema só era parametrizado para reinicialização quando o **MAIN.py** e ou **classes.py** fossem atualizados, agora foi adicionado que se o banco de dados for atualizado o sistema também deverá ser reinicializado.

**Alterações no classes.py**

**202003161400**

Para solucionar o problema que ocorria quando uma atualização era feita no CLP.py e ocorria um erro na execução a versão de CLP.py não era atualizada no banco de dados online, então foi adicionado logo após o classes.py pegar um CLP.py do backup ou backup-1 ele executa o **getVersao** e **setVersao.**

**202003261418**

Com o objetivo de melhorar a velocidade em que o gateway realiza alterações nas entradas e saídas e o feedback ao usuário com os sinalizadores e valores de forçado e escrita. Todas as rotinas que são responsáveis por algum tipo de propriedade que o usuário pode alterar sofreram alterações no ponto em que o código começa verificando se deve forcar ou escrever algo para determinada propriedade, o trecho alterado ficou desta maneira:

if conjVar.escreverNovoValor == True or conjVar.forcar == True:

conjVar.valor = conjVar.novoValor

if conjVar.forcar:conjVar.forcado = True

else:conjVar.forcado = False

if conjVar.escreverNovoValor:

conjVar.escreverNovoValor = False

else:

conjVar.forcado = False

conjVar.valor = (1 if valBit > 0 else 0)

conjVar.timeStamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")

Alterações também foram feitas no método **escrever valores,** agora quando ocorrer o bit 4 serão enviadas aos bancos de dados só as propriedades que de fato sofrerem atualizações.

**202003301430**

Foi observado que o gateway não mostrava valores no site ou banco menores que 1,

Então foi necessário aplicar essa alteração no classes.py.

Em getValVariaveis anterior a esta versão os blocos de if elif e else realizavam um try execpt para definir os valores que chegavam como strings, em inteiros ou floats.

Agora o algoritmo irá olhar para o tipo de valor e então transformar a string para o tipo específico definido no banco de dados offline.

**202003301524**

As temperaturas não estavam sendo forçadas então foi atualizado em Sensor Inteligente no classes.py, então quando houver um force o valor recebe o novo valor adicionado pelo usuário.

**Atualização em BCK.sql**

**202003301524**

Para que o site fosse capaz de adicionar o número de casas decimais em cada valor de acordo com o tipo de valor e tipo de propriedade foi adicionado uma nova coluna no banco de dados online o offline.

A coluna Casas\_Decimais em Nuvem Propriedades\_Máquinas.

**Criação de um novo arquivo para o sistema.**

Até a data de hoje 23-01-2020 o software está vulnerável a erros que podem ocorrer com os arquivos **supervisores** que são responsáveis pela inicialização do sistema, em decorrência de erros de corrupção de arquivos **supervisores** ou a falta de existência deles está sendo implementado o arquivo **restaurasupervisores.py** após baixado no gateway esse arquivo será um **executável** que será chamado uma única vez pelo autostart do Linux do gateway, seu papel será verificar a existência dos arquivos **supervisores** caso algum deles ou ambos não estejam no diretório o **novo arquivo** renomeará os arquivos **supervisores** correspondentes que deverão estar presentes no diretório com o acréscimo ‘**.old**’ no final de seu nome.

**Algoritmo empregado no arquivo restaurasupervisores.py.**

* Baixa as bibliotecas necessárias para utilização de ferramentas durante execução do algoritmo.
* Define o método **getPythonVer().**
  + Criar a lista **pastas** contendo o endereço de todos os diretórios contidos dentro de **/usr/lib/.**
  + Percorre a lista **pastas** procurando pelo diretório **python3.versão.**
  + E então retorna todo o caminho do diretório que possui como nome a versão do PYTHON que está sendo utilizada.
  + Configura o modo **GPIO.BOARD,** cancela o enviou de avisos de erros referentes aos GPIO e em seguida define os pinos **29** e **31** como saída.
  + Define uma lista vazia chamada **supervisores** para utilização durante execução do algoritmo.
  + Utilizando ferramentas internas da linguagem PYTHON tenta abrir um por vez ambos **supervisores** que são necessários para funcionamento do sistema, caso qualquer deles não seja encontrado é adicionado à lista recém-criada o nome dos arquivos supervisores reserva que são: **supervisorBits.old** e **supervisor.**
  + Um bloco condicional então faz uma contagem para verificar se a lista tem algum nome, indicando assim a necessidade de restauração de algum supervisor que não foi encontrado no diretório.
  + Coloca em nível alto os pinos definidos como saída **31** e **29.**
  + Percorre a lista e executa o comando ‘**sudo cp /caminho/arquivo.old /mesmoCaminho/arquivo**’ para todos os arquivos de supervisores que não foram encontrados.
  + Reinicia a CPU para tentativa de sucesso na inicialização do sistema.