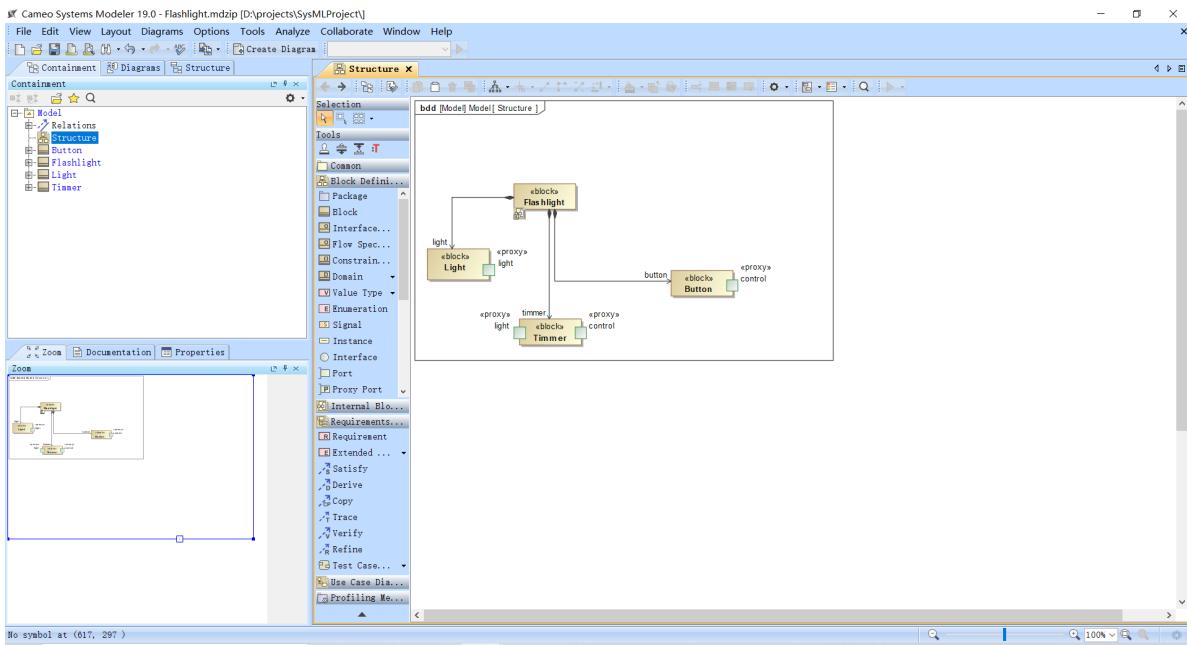


Flashlight

Step

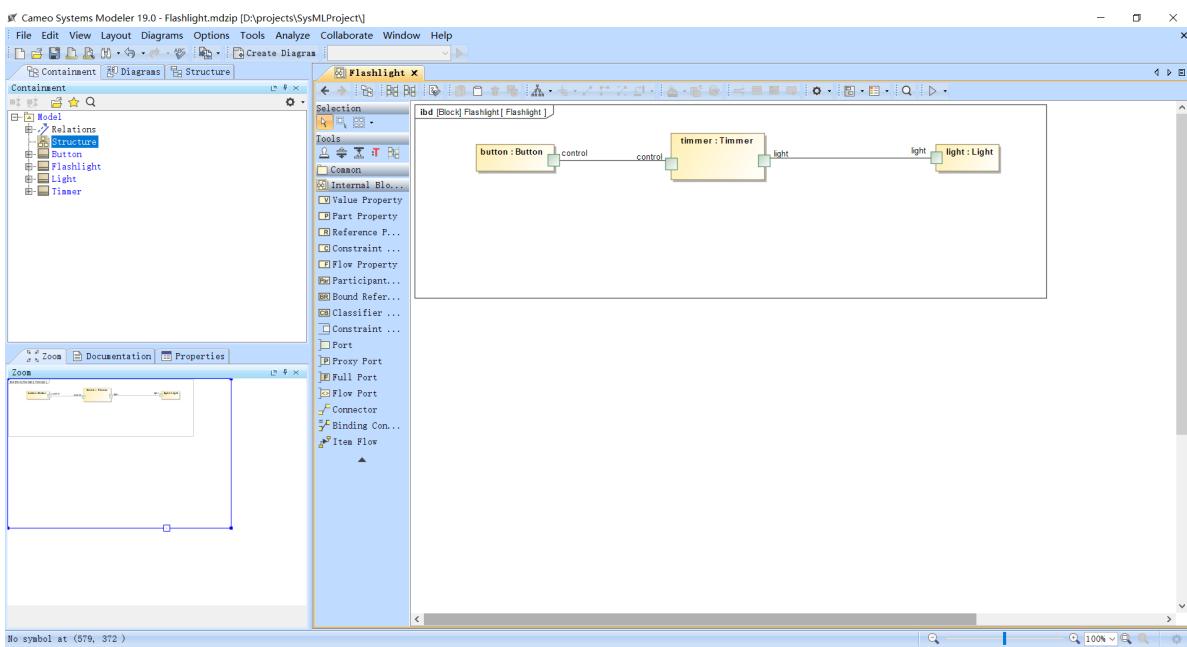
创建bdd

在根目录Model下创建bdd "Structure"， 创建Flashlight及三个子模块（带有代理端口）。



创建ibd

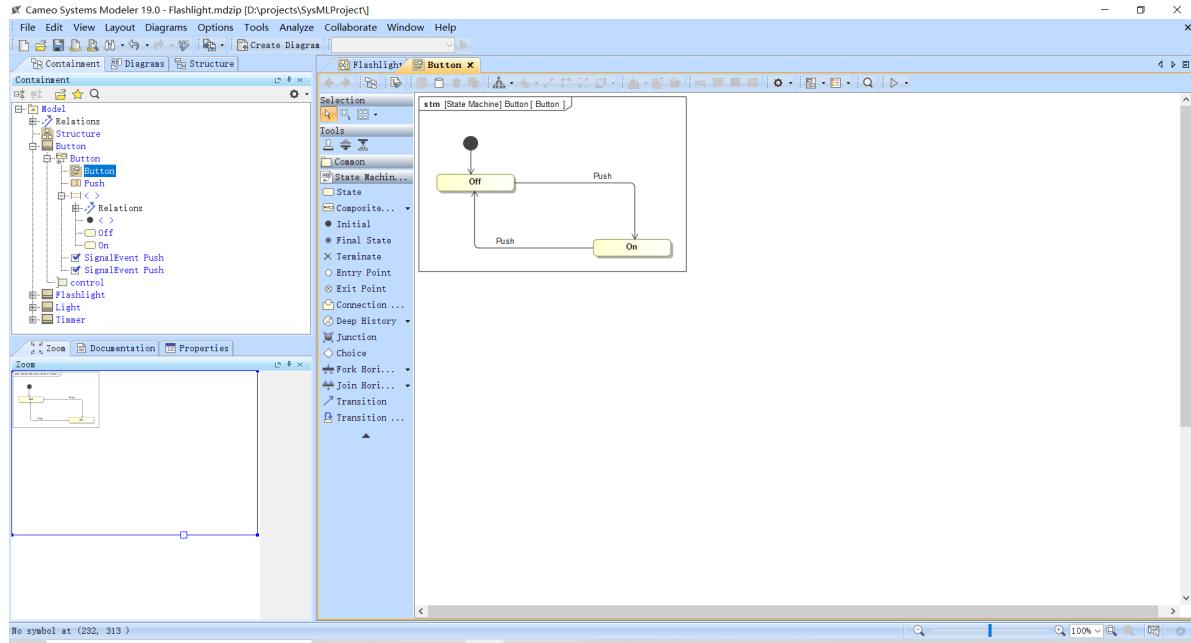
单击bdd中的Flashlight模块，在出现的工具栏中选择SysML Internal Block Diagram创建ibd。



创建状态机图

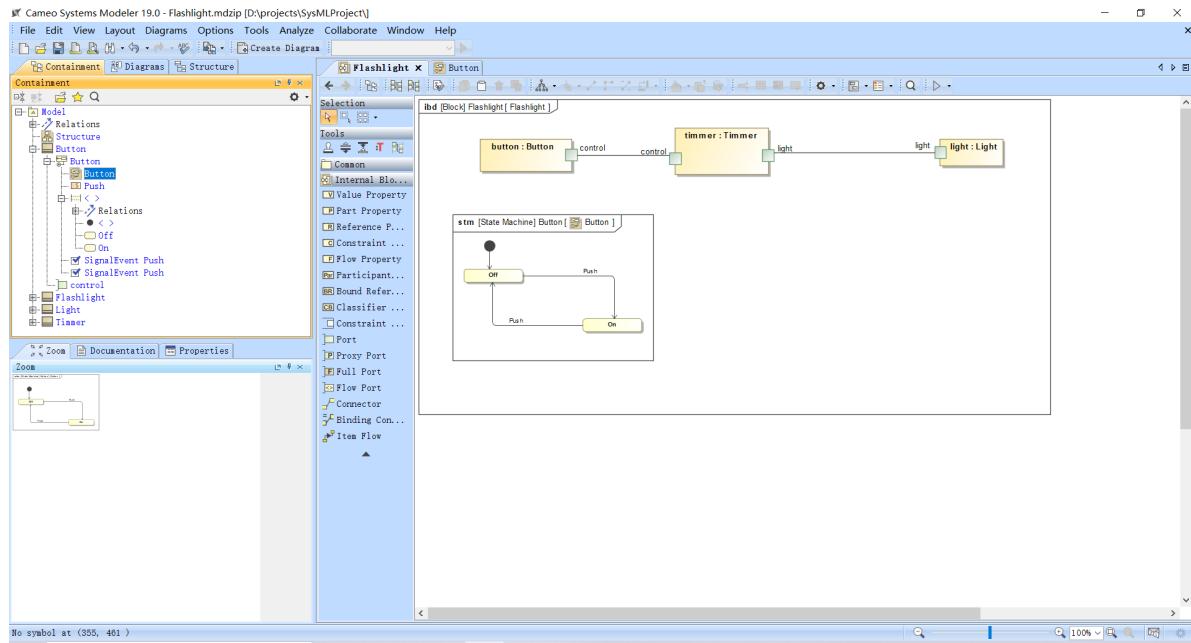
create state machine to describe the logic.

右击Containment中的Button模块元素 (创建stm方式1) -->create Diagram-->SysML State Machine Diagram。

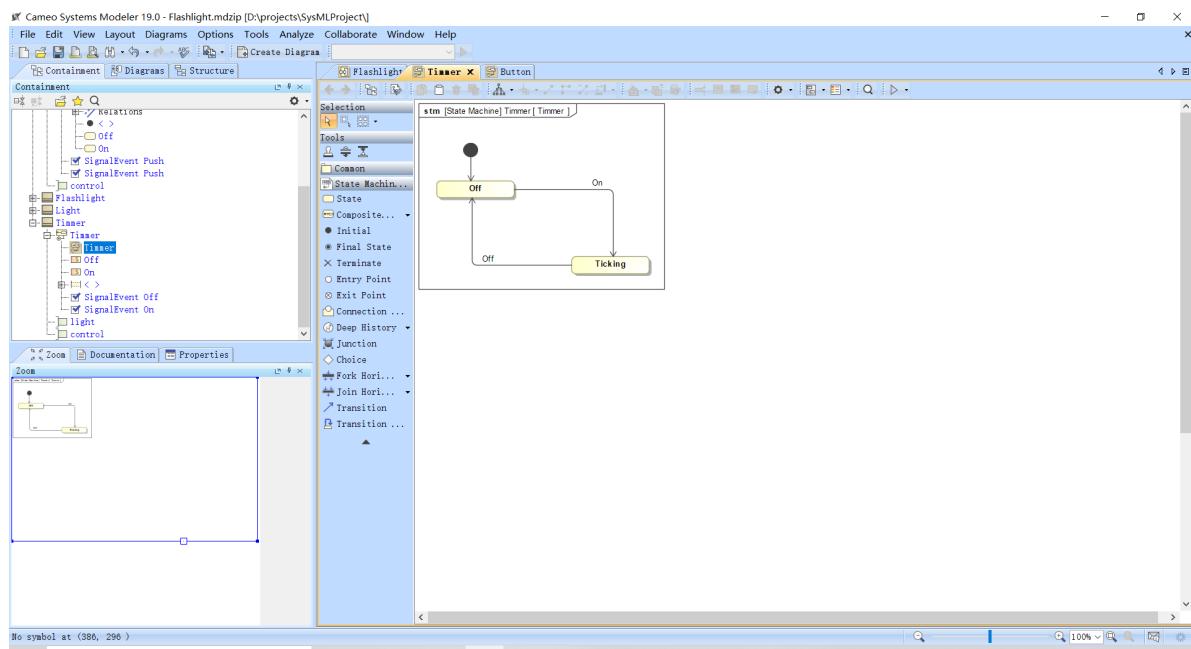


注意：创建图中的状态和转换时，会在Containment中同步创建相应元素，可把信号元素直接拖到一个转换上。

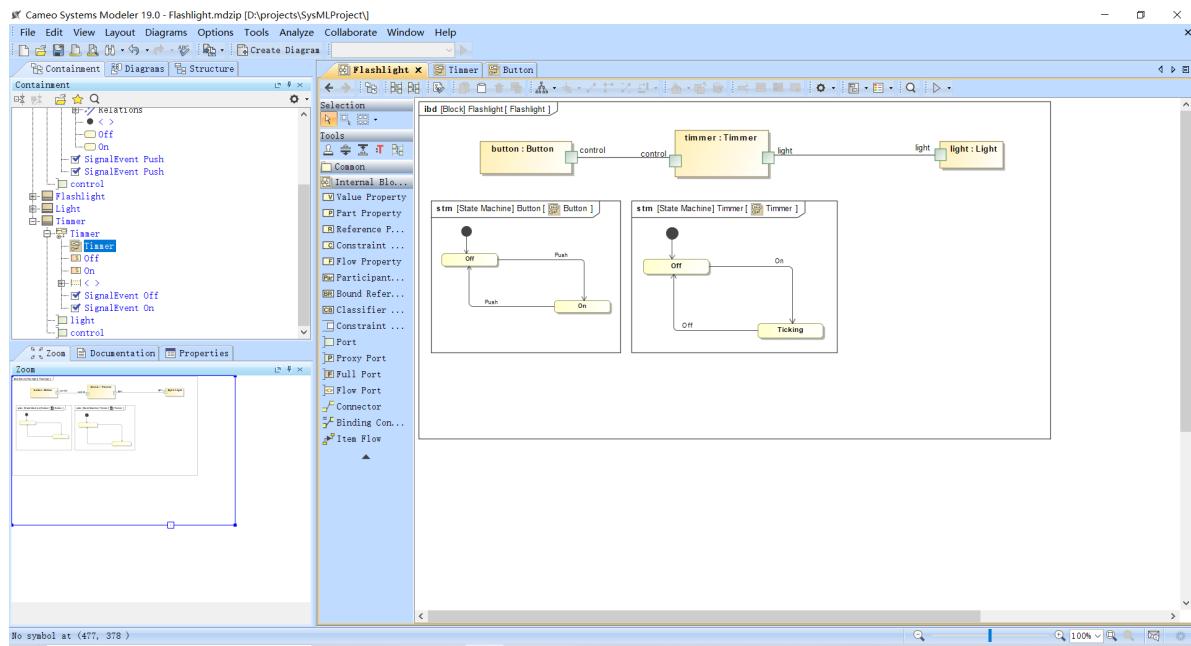
可以将Button的状态机图直接拖到Flashlight的ibd中，使表达更清晰，这不影响模型。



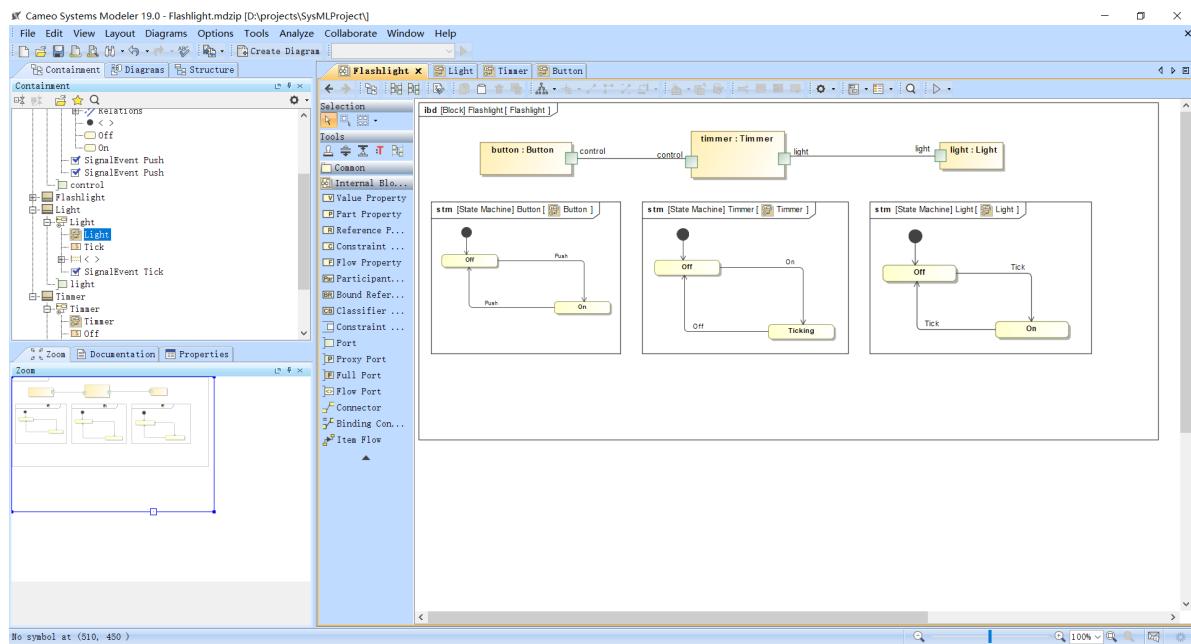
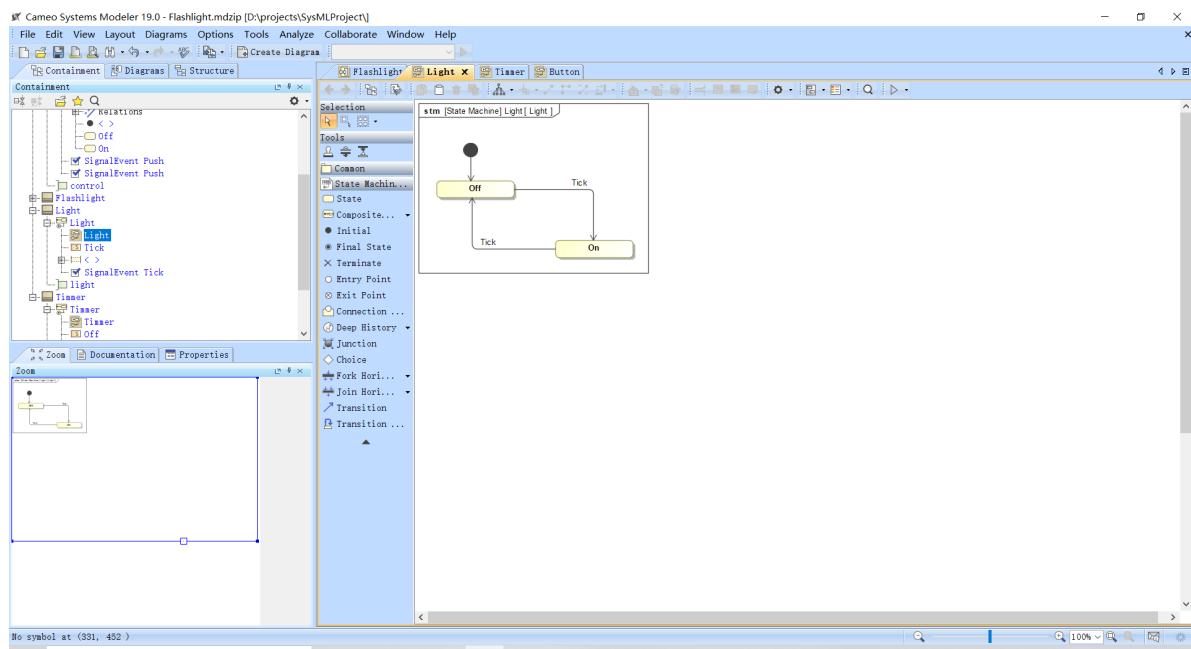
右击Flashlight的ibd中的timmer创建状态机图 (方式2) -->create Diagram-->SysML State Machine Diagram。



将Timmer的状态机图直接拖到Flashlight的ibd中。



类似地，创建Light的状态机图，并将其拖到Flashlight的ibd中。



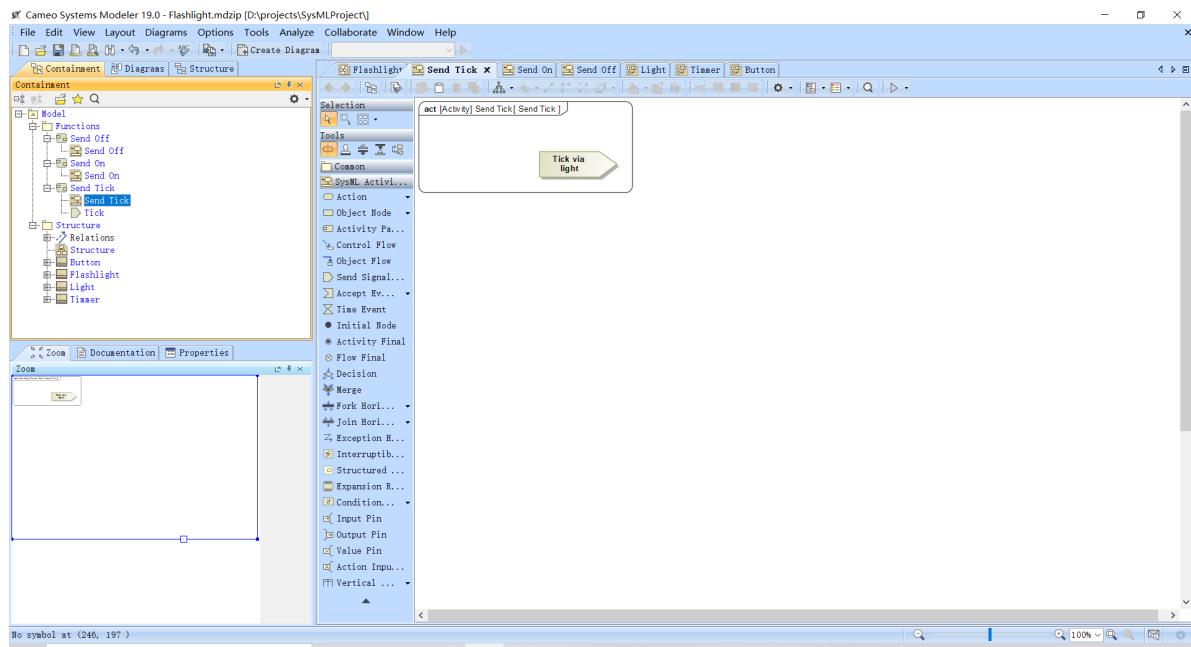
在Timmer状态机图中给Ticking加个自转换"after(2s)"，图略。

在根目录Model下-->Create Element-->创建"Structure"包，并将"Stucture"bdd以及其中的四个模块元素(Button、Flashlight、Light、Timmer)移入"Structure"包中，图略。

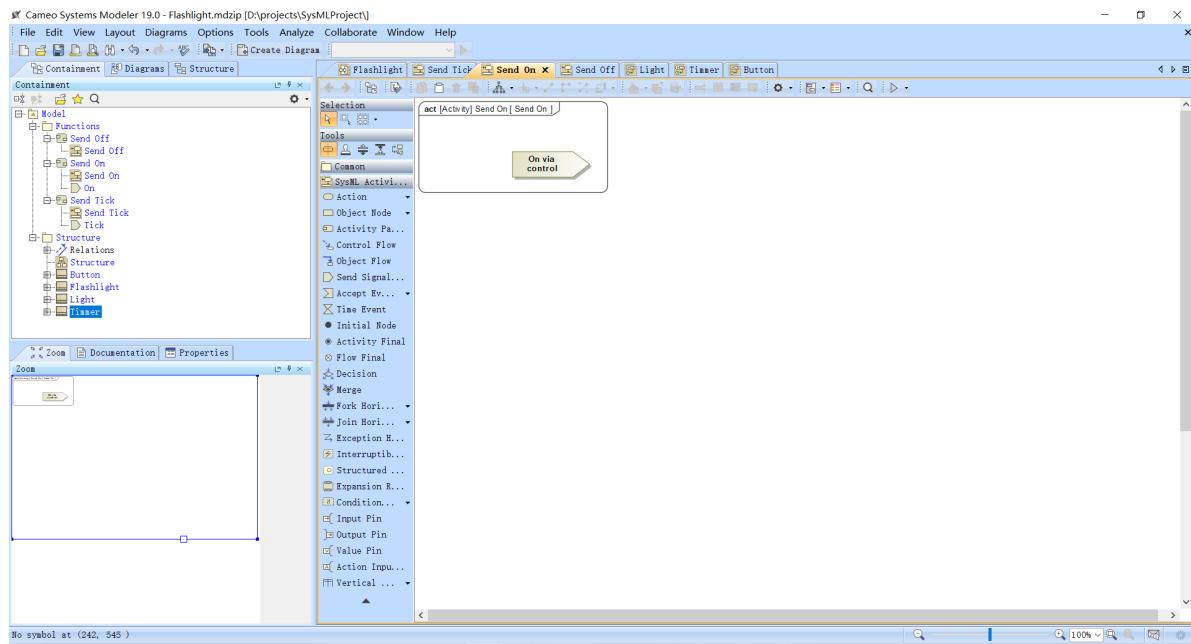
创建活动图

在根目录Model下-->Create Element-->创建"Functions"包，在该包下创建几个活动图 to send those signals(上一步在状态机图创建的各种信号，如Tick)，创建活动图的过程中时不时参考Flashlight的ibd。

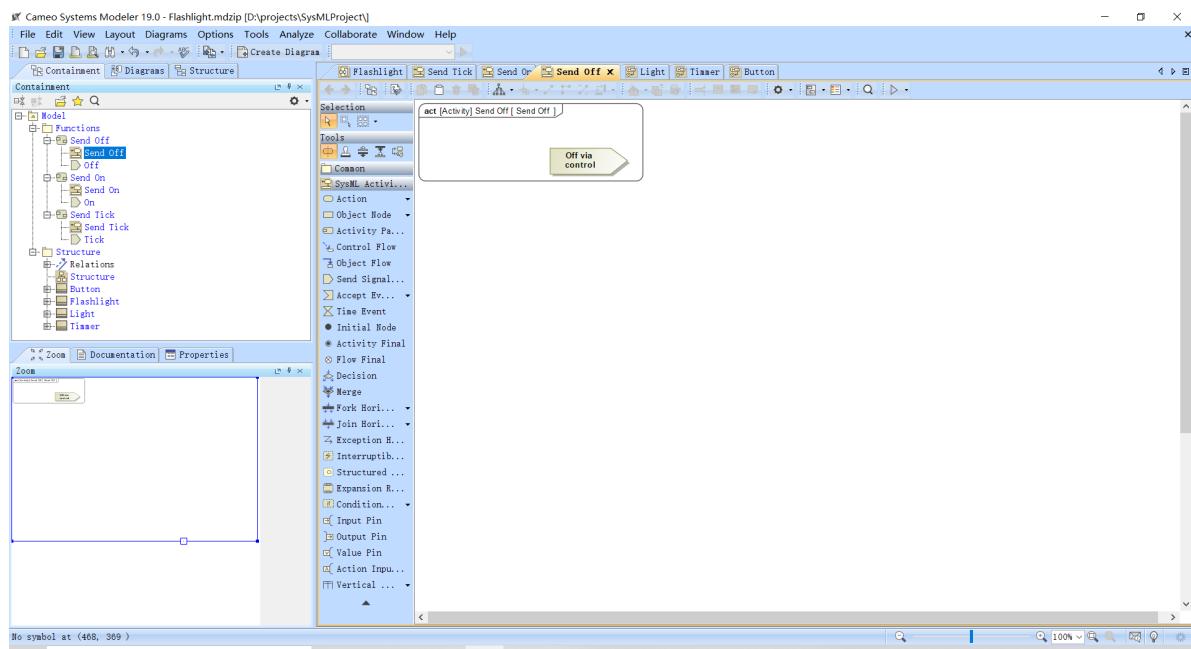
创建"Send Tick"活动图：1.将"Structure"包下"Light"元素的状态机中的"Tick"信号拖到该活动图中；2.双击图中的"Tick"信号(软件将其称为"发送信号动作") -->在"Specification of Send Signal Action<>框"中点开"On Port栏"的选择框（找不到On port，记得将"Standard"改为"Expert"）-->在"Select Port框"中选择Structure包下Timmer元素的"light"端口（在ibd中，Timmer给Light元素发送信号）。选择后信号名称变为"Tick via light"。



创建"Send On"活动图：1.将"Structure"包下"Timmer"元素的状态机中的"On"信号拖到该活动图中；2.双击图中的"On"信号(软件将其称为"发送信号动作") -->在"Specification of Send Signal Action<>框"中点开"On Port栏"的选择框-->在"Select Port框"中选择Structure包下Button元素的"control"端口。选择后信号名称变为"On via control"。

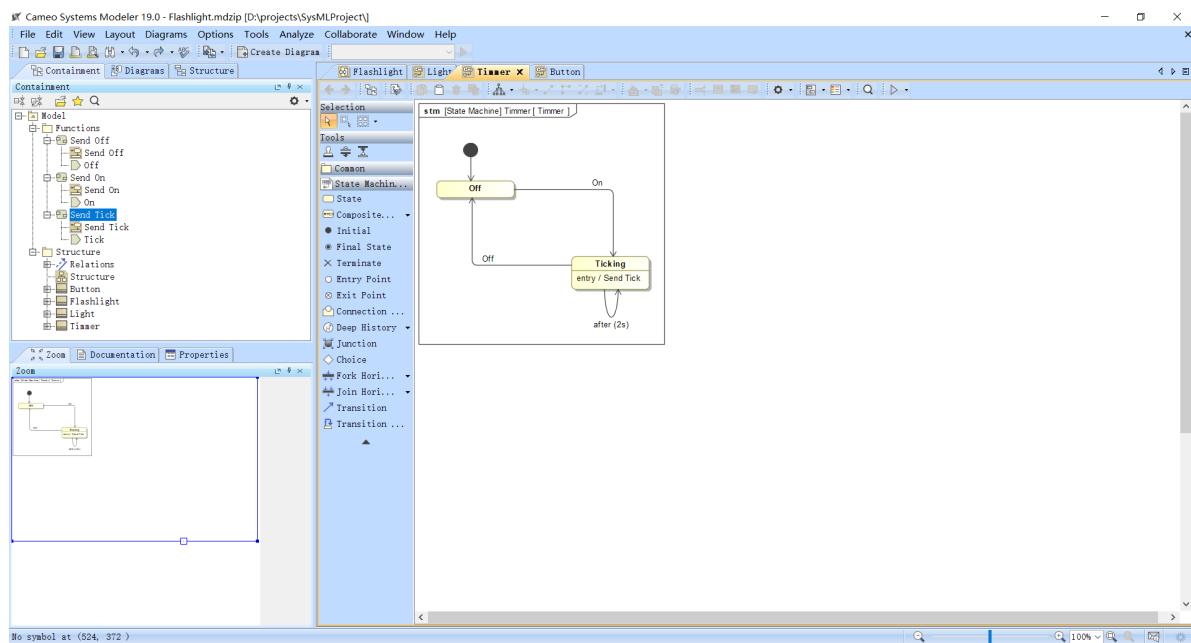


创建"Send Off"活动图：1.将"Structure"包下"Timmer"元素的状态机中的"Off"信号拖到该活动图中；2.双击图中的"Off"信号(软件将其称为"发送信号动作") -->在"Specification of Send Signal Action<>框"中点开"On Port栏"的选择框-->在"Select Port框"中选择Structure包下Button元素的"control"端口。选择后信号名称变为"Off via control"。



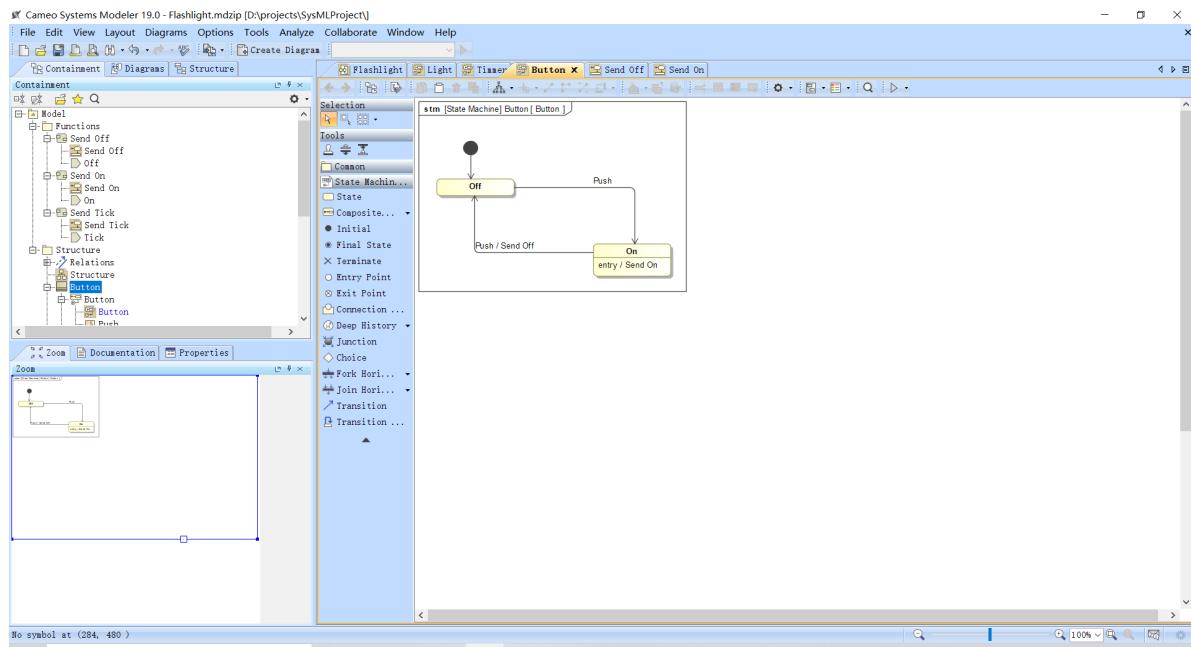
将这些活动（注意，不是活动图，例如外层的"Send Off"表示活动，它包括内层的"Send Off"活动图和"On"元素）分配给对应的状态。

将"Send Tick"活动拖到Timmer状态机图中的"Tick"状态，选择"Entry"，表示该状态的入口行为是Send Tick。



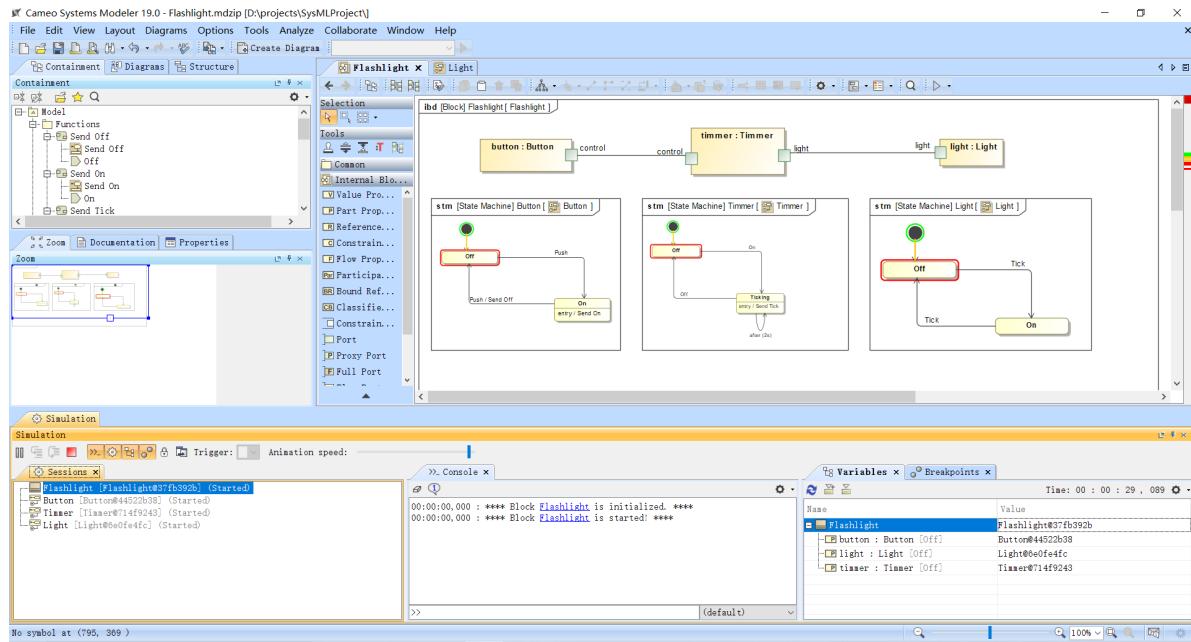
将"Send On"活动拖到Button状态机图中的"On"状态，选择"Entry"，表示该状态的入口行为是Send Tick。

将"Send Off"活动拖到Button状态机图中的"On"状态到"Off"状态的转换，表示该转换的效果行为是"Send Off"(不要把"Send Off"活动分配给Button状态机图中的"On"状态的"Entry"，因为默认情况下就会进入"On"状态；也可以把这个活动分配给"On"状态的"Exit"行为）。

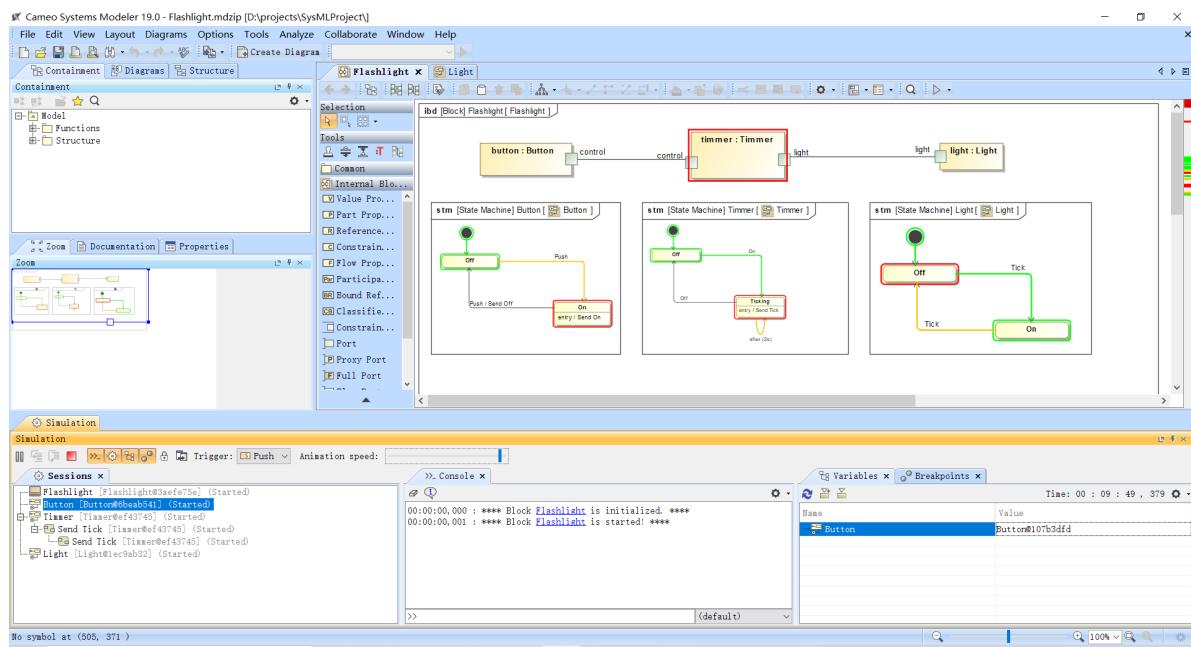


状态机图仿真

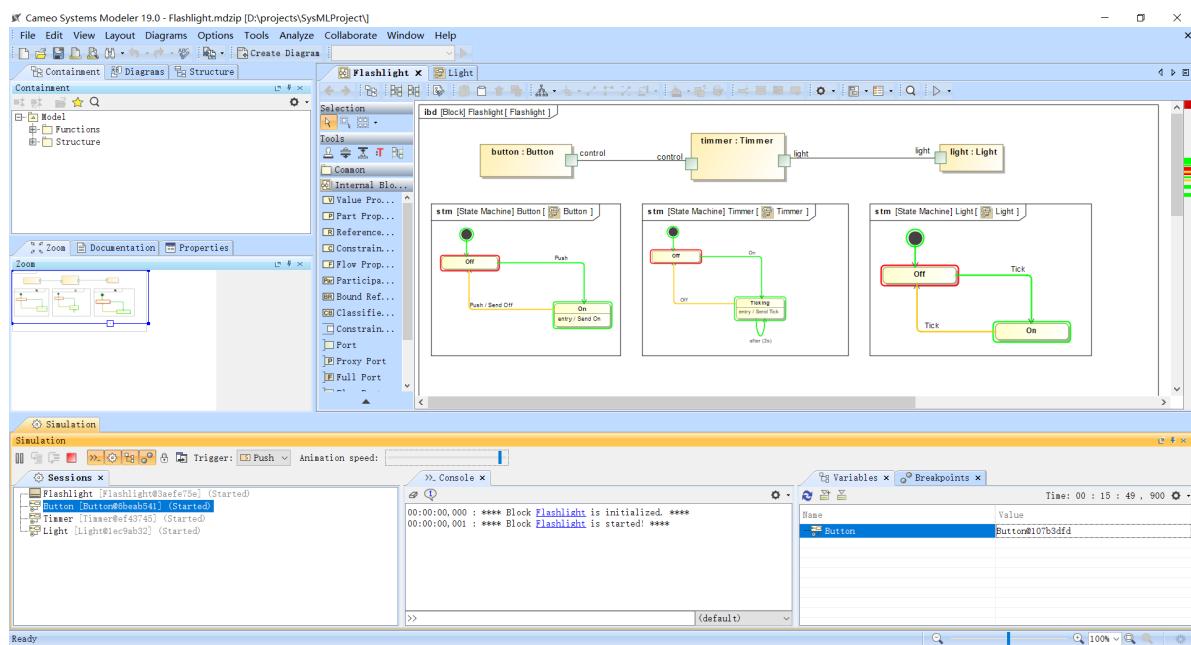
此时Flashlight有了一套行为逻辑，可以在其ibd中把这些行为跑起来：点击ibd上方工具栏中的三角形(Run)，弹出仿真(Simulation)窗口，初始化Flashlight模块。-->点击仿真窗口中的绿色三角形(Start)，以启动行为。



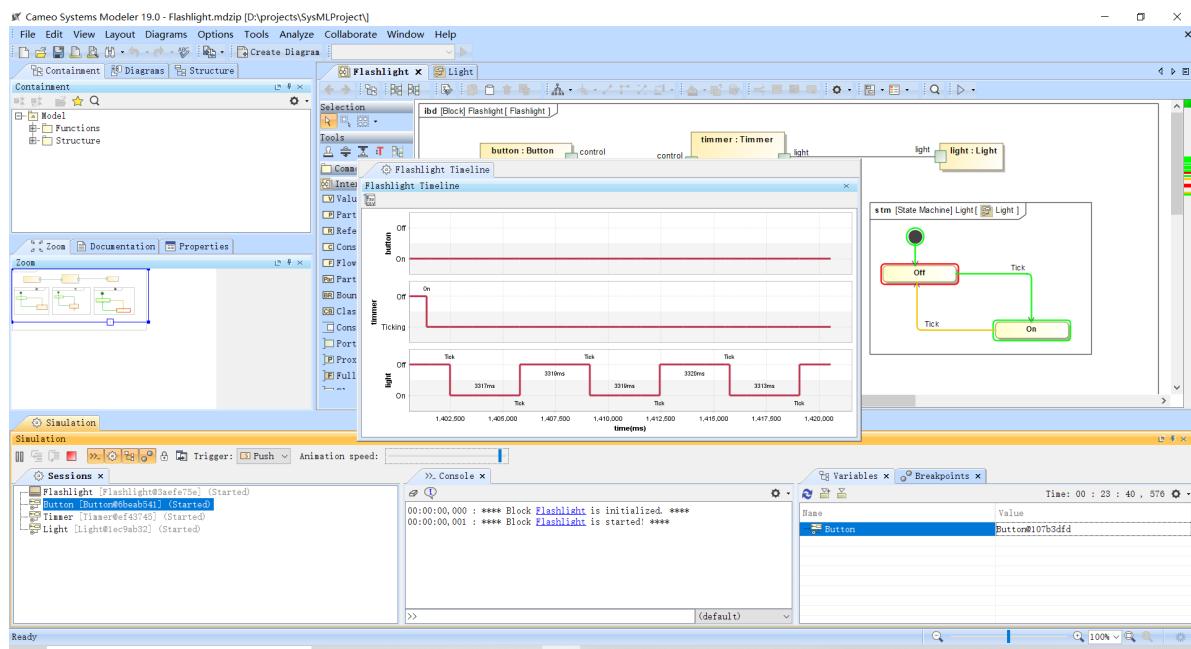
点击Sessions窗口中的Button，此时Simulation窗口的Trigger中会出现"Push"信号，点击Trigger下拉框再点击"Push"，从而执行发送"push"信号操作。然后ibd上呈现信号发送、状态转移的动画。



再发送一次"Push"信号，Button和Timmer的状态回到Off，Light的状态既可能是Off也可能是On。



右击Variables窗格中的Flashlight-->Show in Timeline chart-->state，可以看到Flashlight各子模块的状态随时间变化（通过发信号）的情况。

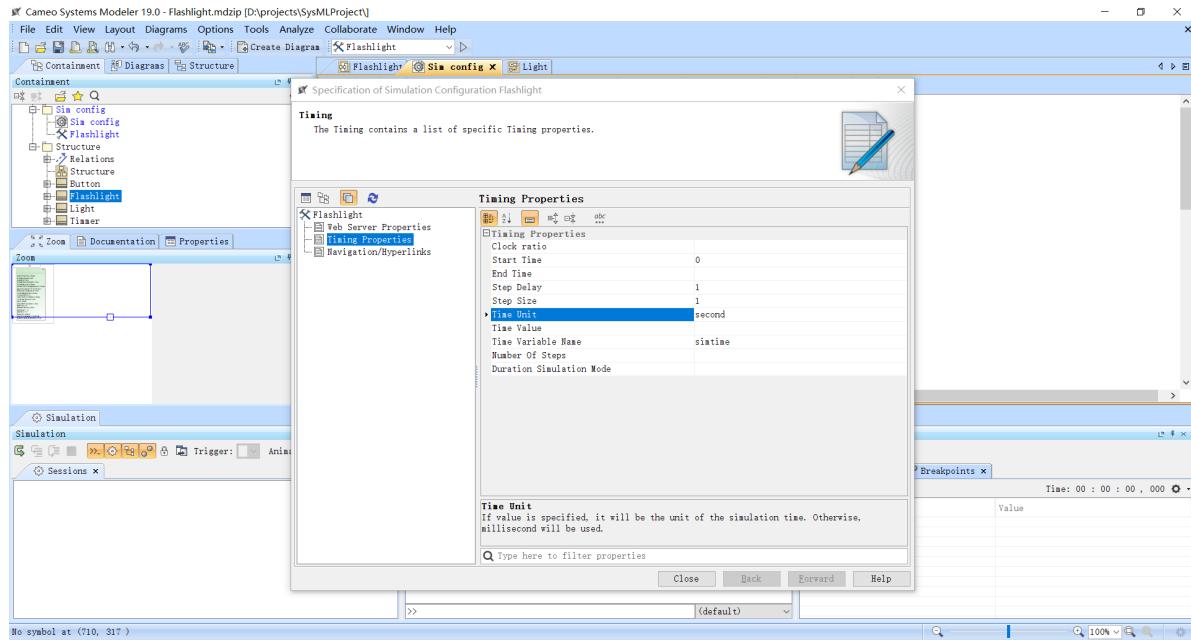


创建仿真配置图

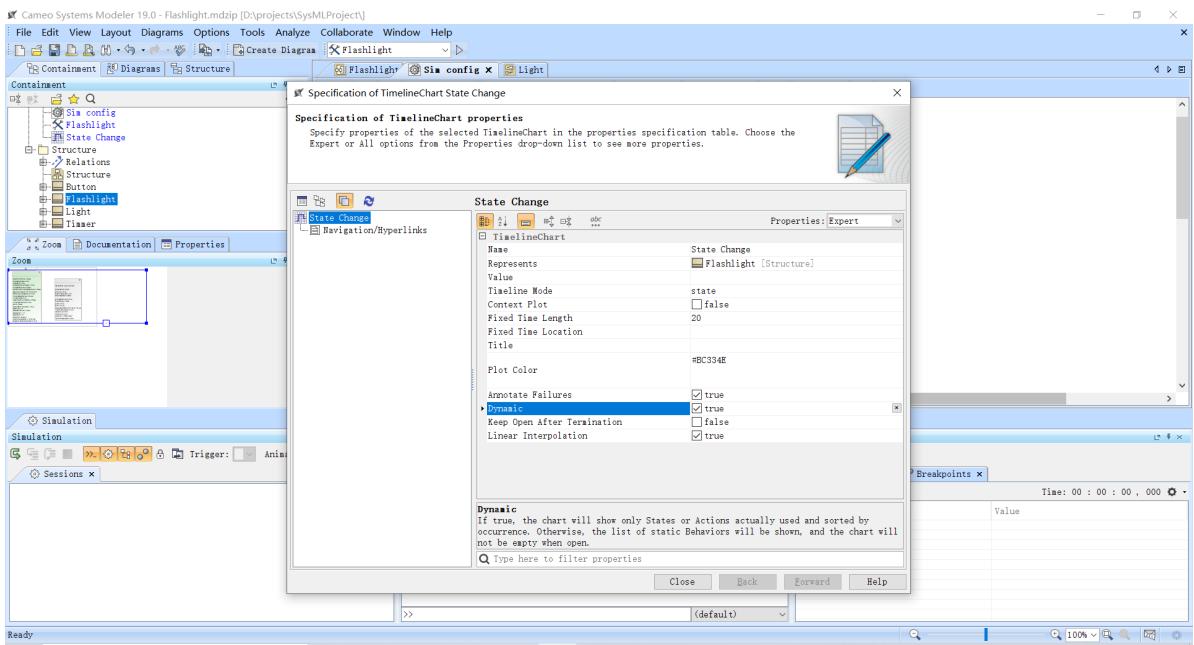
在根目录Model下-->Create Element-->创建"Sim config"包-->在该包中创建Simulation Configuration Diagram, 修改图中《SimulationConfig》框的名字为Flashlight。

将Structure包中的Flashlight模块拖到《SimulationConfig》框中, 这样能将executionTarget设置为Flashlight模块。

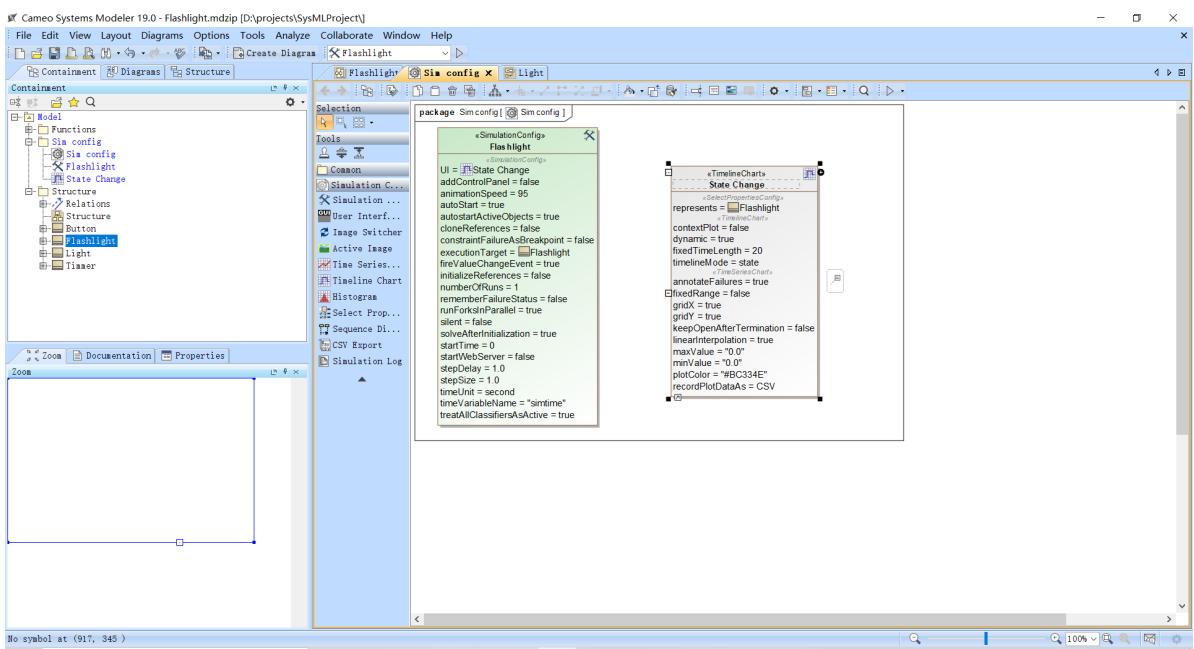
双击《SimulationConfig》框-->Timing Properties-->设置Step size等属性,如下:



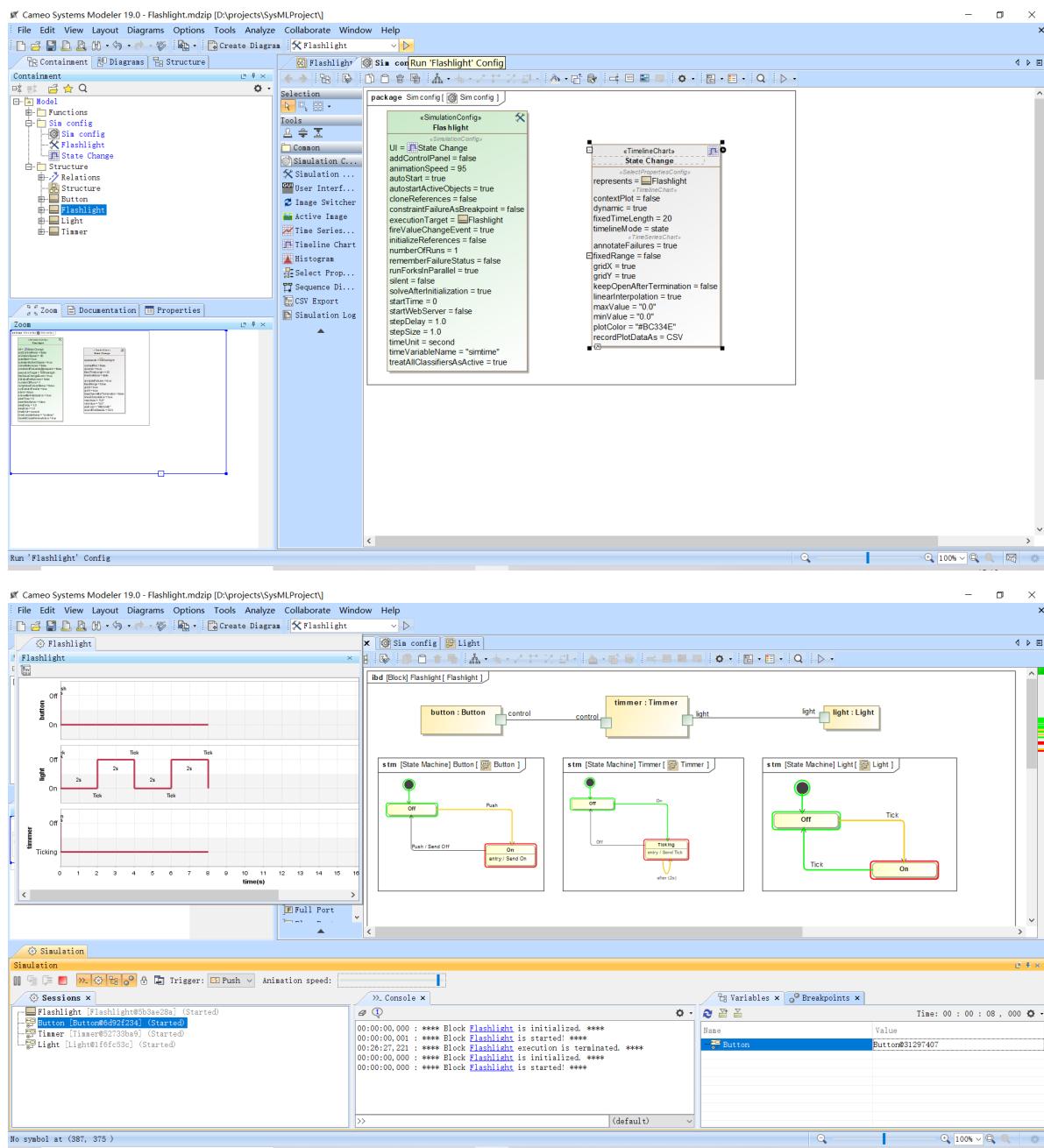
从"Sim Config"图的侧方工具栏中将Timeline Chart拖入图中, 命名"State Change"-->类似地, 将Structure包中的Flashlight模块拖到《Timeline Chart》框, 会多出一行属性"represents=Flashlight"-->双击《Timeline Chart》框设置属性,如下:



将《Timeline Chart》框拖入《SimulationConfig》框，从而设置后者的UI属性。



执行Simulation Configuration



可视化按钮和灯光

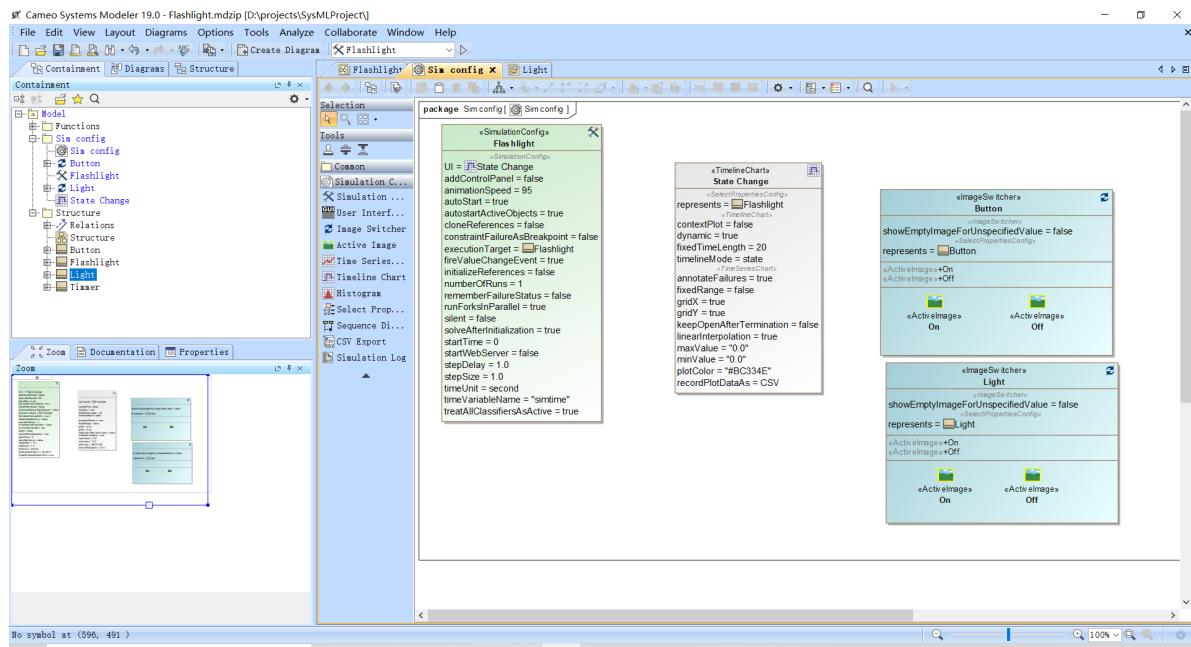
回到"Sim config"图，可以使用User Interface Configuration或Image Swicher来可视化，在此选用后者。将两个Image Swicher拖到"Sim config"图中，分别命名为"Button"和"Light"。

将两个Active Image拖入Image Swicher "Button"，分别命名为"On"和"Off"。

将两个Active Image拖入Image Swicher "Light"，分别命名为"On"和"Off"。

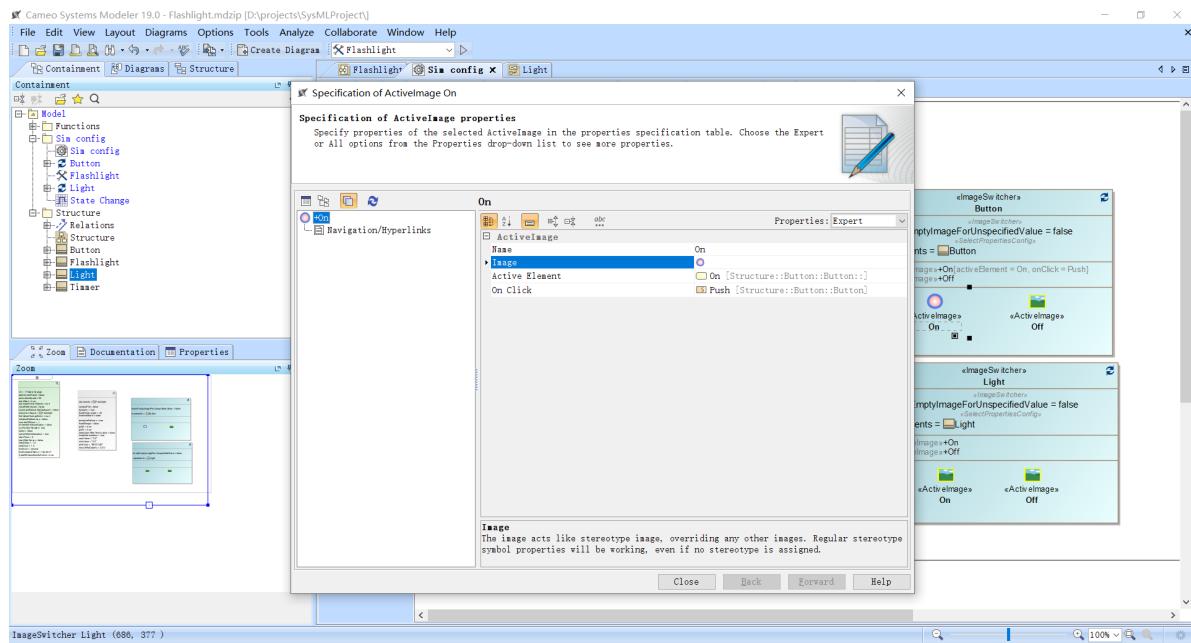
将Structure包中的Button模块拖到Image Swicher "Button"，会多出一行属性"represents=Button"，效果就是该Image Swicher代表Button模块。

将Structure包中的Light模块拖到Image Swicher "Light"，会多出一行属性"represents=Light"。

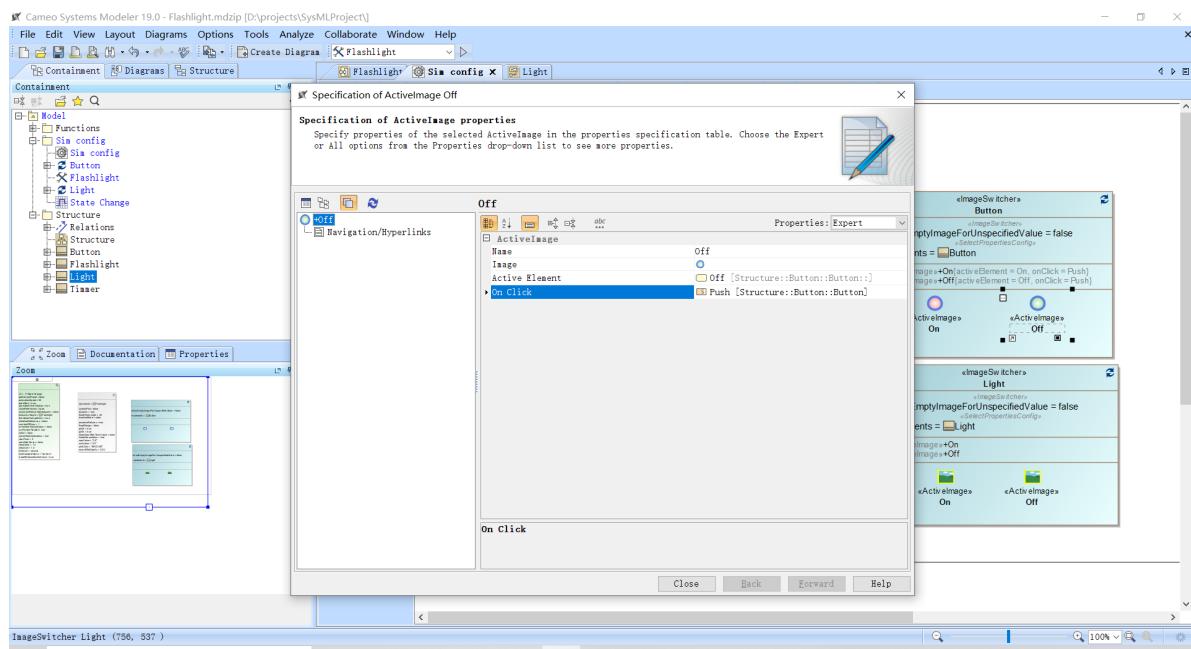


接下来要指定每个Active Image对应的状态。

双击"Button"中的Active Image "On"，设置Active Element属性为Button模块的On状态，设置On Click属性为Push (i.e, Send Signal "Push")，设置Image属性随便选一个就行 (此处选粉红圆圈)。



设置"Button"中的Active Image "Off"的Active Element属性为Button模块的Off状态，设置On Click属性为Push (i.e, Send Signal "Push")，设置Image属性随便选一个就行 (此处选绿色圆圈)。

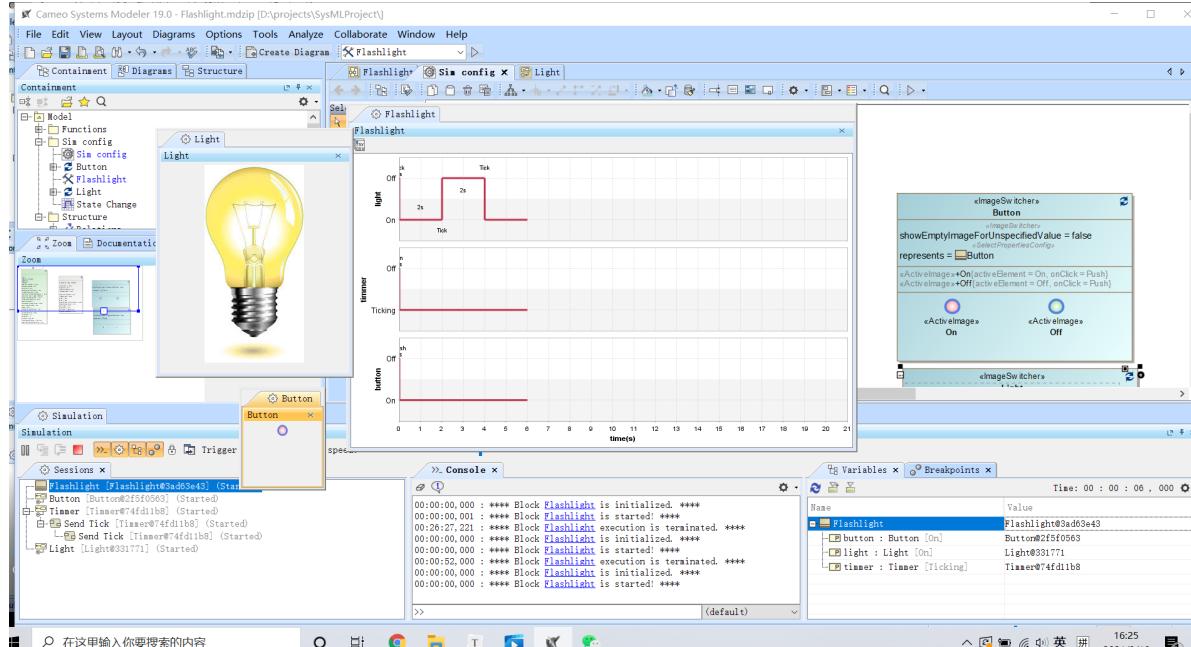


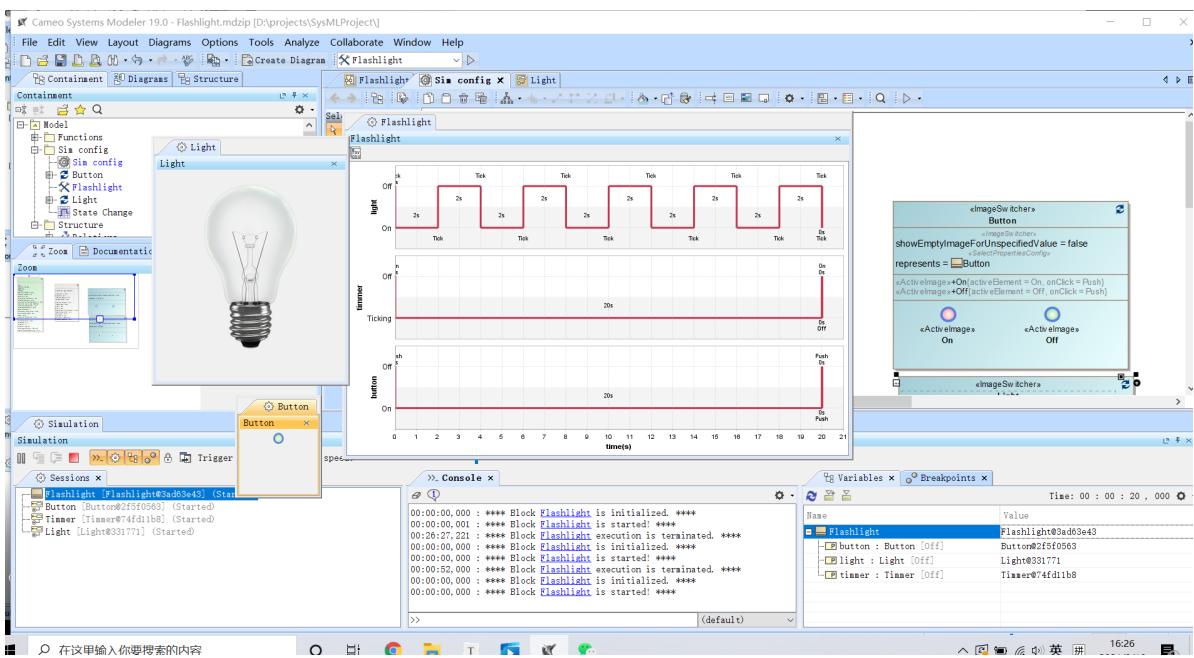
从网上随便搜索light on和light off的图，拖到"Light"中相应的Active Image图标上，便捷地设置Image属性。

设置"Light"中的Active Image "On"的Active Element属性为Light模块的On状态，Active Image "Off"的Active Element属性为Light模块的Off状态。"Light"不需要设置On click属性。

将两个Image Swicher拖到《SimulationConfig》框中，添加UI属性的值。

此时执行仿真配置文件Flashlight，得到下图效果：





注意：若双击《SimulationConfig》框，将silent属性设置为true，则仿真过程中没有动画效果。

如果想看到序列图，回到"Sim config"图，将Sequence Diagram Generator拖到图中，命名为"Sequence"，再将该Generator拖到《SimulationConfig》框。此时执行仿真配置文件Flashlight，得到下图效果（此处将Timmer的Ticking自转移改为1s,即after(1s)）：

