

カスタム イベント プロバイダのサンプル

*このサンプルは、2016 年 3 月の Xbox One XDK 以降と互換性があります。*

# 説明 このサンプルでは、Xbox One でカスタム ETW イベントプロバイダを使用する方法を示します。 サンプルの使用

サンプルでは、以下のコントロールを使用しています。

|  |  |
| --- | --- |
| 作用 | ゲームパッド |
| サンプルを終了する。 | 左トリガー + 右トリガー + Rボタン（rb） |

実装上の注意  
このサンプルは従来の Windows ETW プロバイダと同じ構造に従いますが、Exclusive パーティションで実行されているタイトルは自身のイベントプロバイダをレジストリに追加できないため、生成されたイベント データを正しく解決するためにはホスト PC上で追加の手順が必要です。

イベント マニフェストの **etwprovider.man** で、手動で編集することも、Windows SDK の一部として提供されている ecmangen.exe ツールを使用して編集することもできます。Visual Studio　メッセージ　コンパイラ　(mc.exe)　によってリソース ファイル (**etwproviderGenerated.rc**) とヘッダ （**etwproviderGenerated.h**） にコンパイルされます。リソース ファイルとヘッダーの両方がタイトル プロジェクトに含まれています。

タイトルの初期化中に、イベント プロバイダは**EventRegisterCEP\_Main**の呼び出し経由で登録され、 そして続けてシャットダウン処理中に**EventUnregisterCEP\_Main** の呼び出し経由で登録解除されます。「Mark」イベントは、単一の Unicode 文字列をパラメータとして受け取る **EventWriteMark** への呼び出しを介して発行されます。

サンプルは、他のサンプルと同様に構築、配置およびアクティブ化できます。実行したら、tracelog を使用してイベントをキャプチャします (xbperf にはカスタム イベント プロバイダを指定する機能がありません)。タイトルのイベント プロバイダはレジストリに追加されていないため、名前ではなく GUID で識別する必要があります (GUID はイベント マニフェストで指定されたプロバイダの GUID と一致する必要があります)。

C:\temp>xbrun /x/title /O tracelog -start CustomSession -f d:\custom03.etl -eflag PROC\_THREAD+LOADER+DPC+INTERRUPT+CSWITCH+PROFILE -guid #{A4A76336-4BA7-4CD9-85C3-B9C236D3041C} -stackwalk PROFILE+CSWITCH

必要なデータが取得されたら、通常の方法でセッションを停止できます ;

C:\temp>xbrun /x/title /O tracelog -stop CustomSession

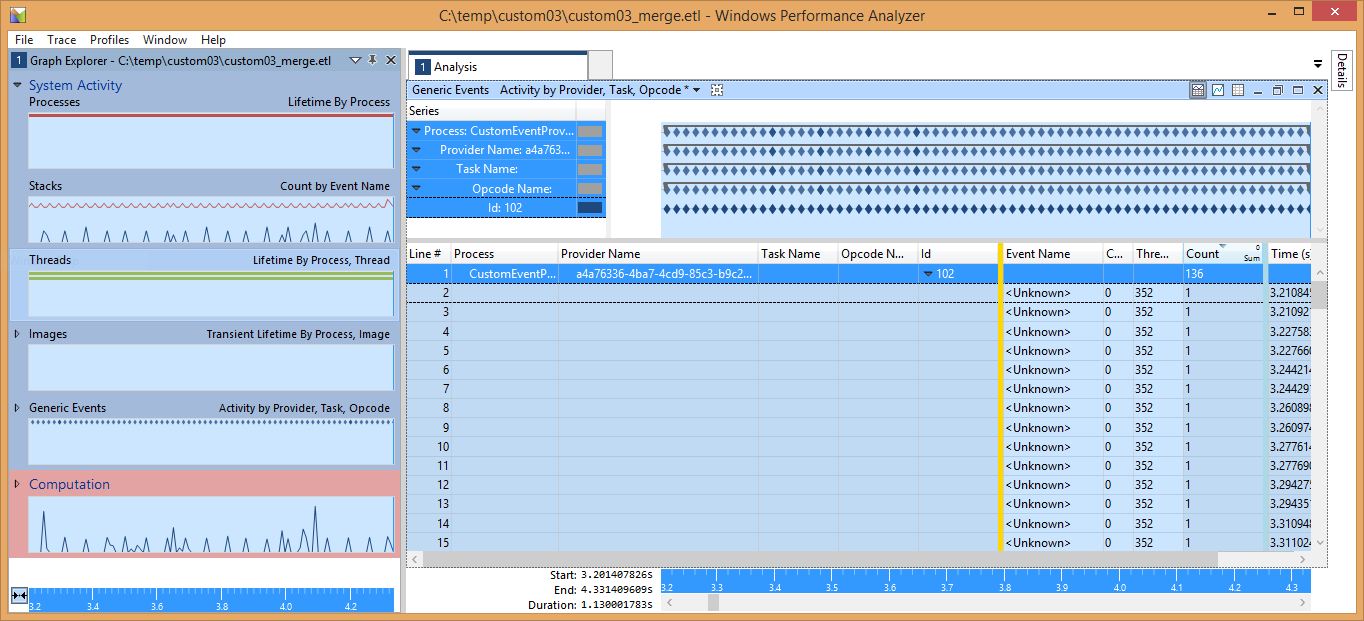
開発キットの ETL ファイルをマージしてシステム イベント プロバイダを解決します。カスタム イベント プロバイダ自体は解決*しません*:

C:\temp>xbrun /x/title /O tracelog -merge d:\custom03.etl d:\custom03\_merge.etl

これで、マージしたファイルをホスト PC にコピーバックすることができます:

C:\temp\custom03>xbcp /x/title xd:\custom03\_merge.etl .

このファイルは　WPA　にロードすることができ、カスタム　イベントは「システム　アクティビティ」グループ内の「Generic Events」グラフに表示されます。ただし、この時点ではイベントは GUID によってのみ識別され、タスク名やオペコード名などの情報は表示されません。さらに重要なことに、それぞれのイベントに提供したカスタム データ (Unicode 文字列) も表示されません。



それぞれのカスタム イベントの完全な情報を表示するためには、一工夫します。つまり、*開発キットではなくホスト PC にイベント プロバイダを登録し、そこでイベントを解決します。*

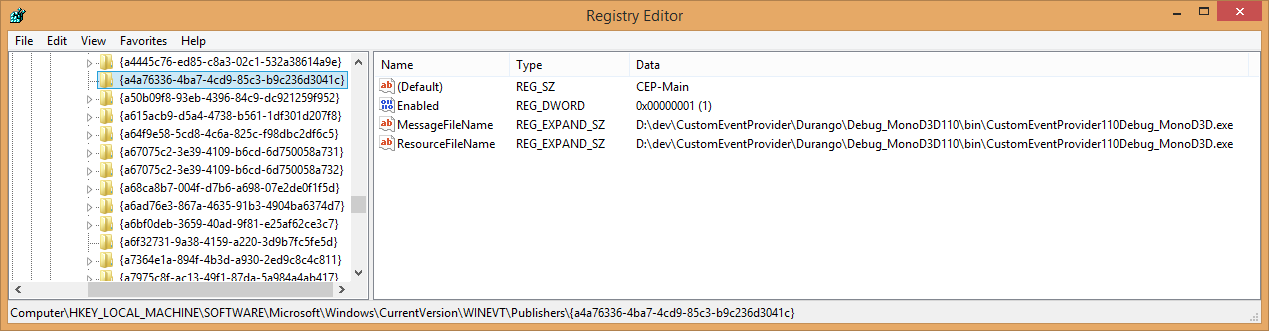
まず、イベント マニフェストのプロバイダ ノード (evtprovider.man) を編集し、**resourceFileName** および **messageFileName** の属性が、開発用 PC 上の Xbox One 実行可能ファイルが構築されている場所を指すようにします：

<provider name="CEP-Main" guid="{A4A76336-4BA7-4CD9-85C3-B9C236D3041C}"   
symbol="CEP\_MAIN"   
resourceFileName="D:\dev\CustomEventProvider\Durango\Debug\_MonoD3D110\bin\CustomEventProvider110Debug\_MonoD3D.exe"   
messageFileName="D:\dev\CustomEventProvider\Durango\Debug\_MonoD3D110\bin\CustomEventProvider110Debug\_MonoD3D.exe">

次に、権限昇格したコマンド プロンプトから wevtutil.exe ツールを実行して、ホスト PC にイベント プロバイダを登録します。

D:\dev\CustomEventProvider>wevtutil im etwprovider.man

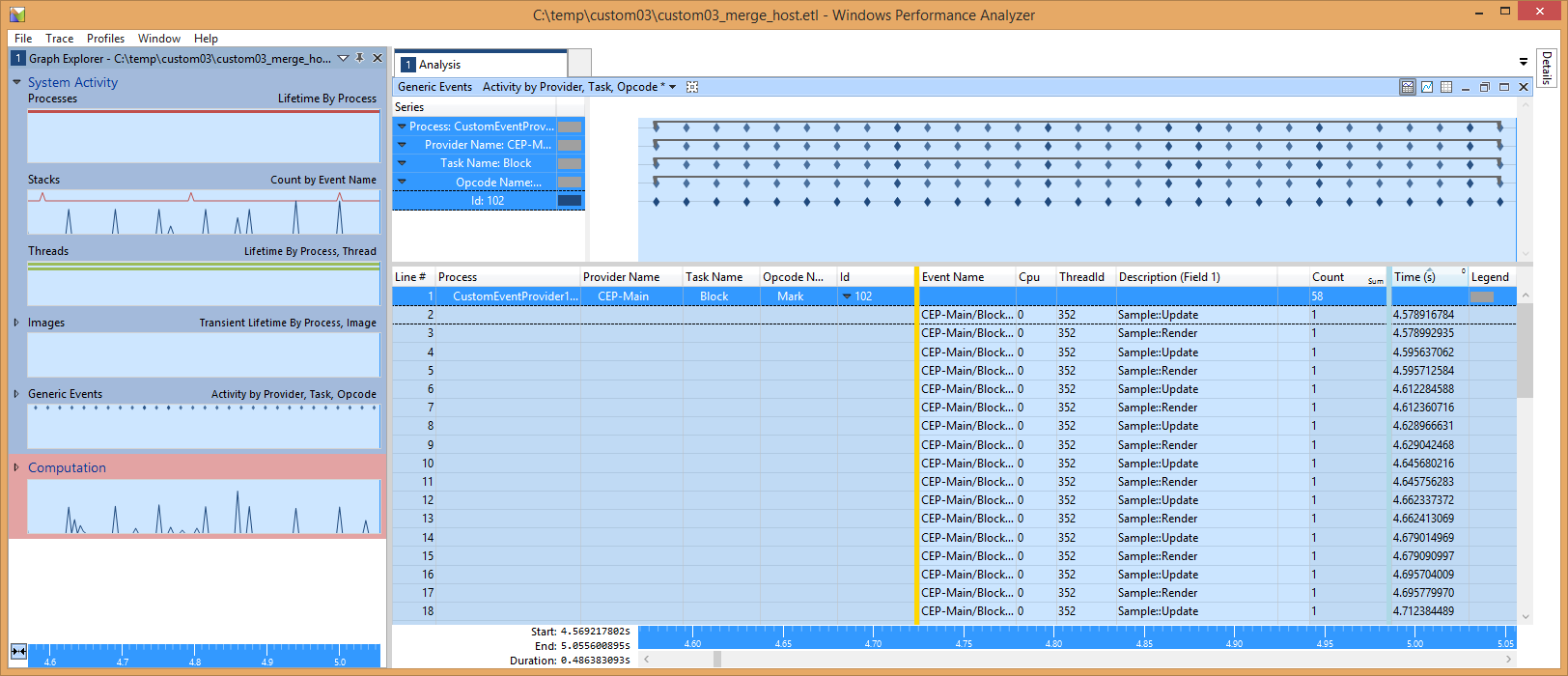
ホスト PC のレジストリにチェックインすると、HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WINEVT\Publishers の下にプロバイダが表示されます。



最後に、xperf を使用してホスト PC 上の ETL ファイルを解決します:

C:\temp\custom03>xperf -merge custom03\_merge.etl custom03\_merge\_host.etl

ホスト マージ ETL ファイルが WPA にロードされている場合は、イベントが正しく解決されているはずです。



Description (Field 1) 列に、イベントとともに記録された文字列が表示されるようになりました。タスク名とオペコード名も確認できます。

パフォーマンス分析セッションが終了したら、ホスト PC からプロバイダを削除できます:

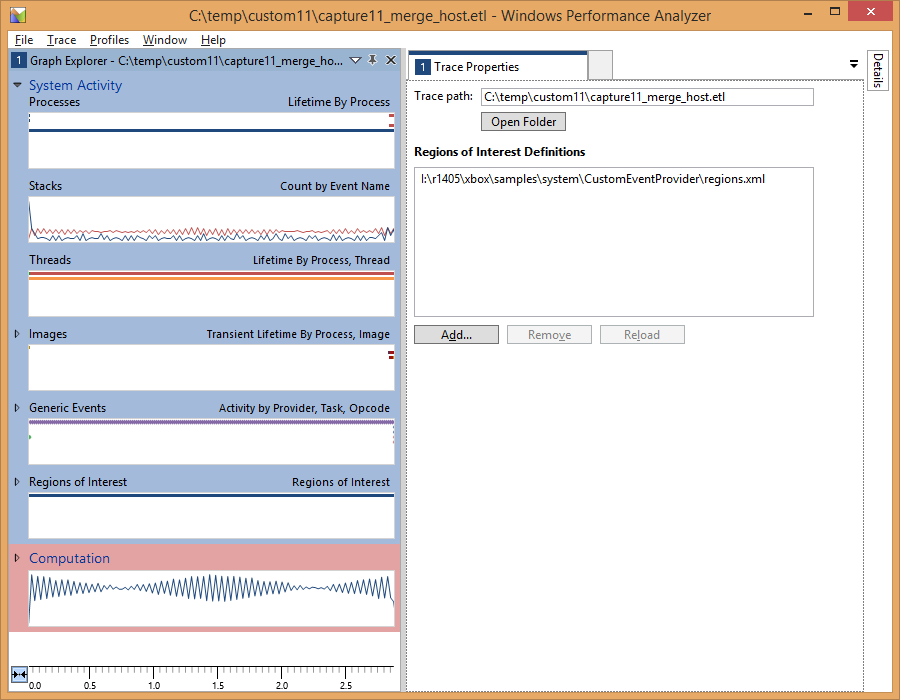
D:\dev\CustomEventProvider>wevtutil um etwprovider.man

BlockCulled イベントは Mark イベントと似ていますが、文字列ペイロードではなく単一の UInt32 ペイロードを持つという点では異なります。残念ながら、現在のところ WPA 内でカスタム イベントの数値フィールドをグラフ化することはできません。

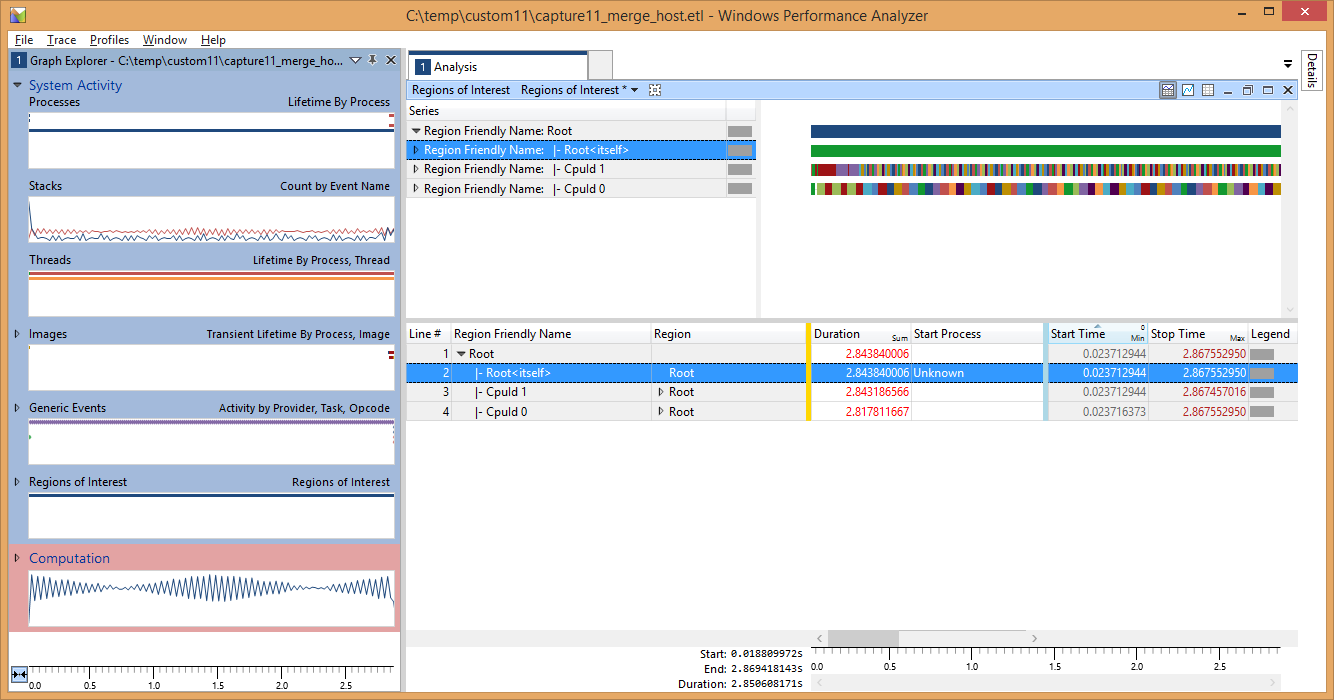
**Regions of Interest**

2013 年 10 月以来、WPA は[Regions of Interest](http://msdn.microsoft.com/ja-jp/library/windows/hardware/dn450838.aspx): の概念をサポートしてきました。つまり、キャプチャ内の一時的な範囲を示し、ラベルを付ける機能です。**EtwScopedEvent** クラスと **ETWScopedEvent()** マクロは、適切なペイロードにより、Regions of Interest (ROI) を使用して　**PIXBeginEvent()**　と　**PIXEndEvent()**　に類似するブラケット機能を提供する方法を示します。

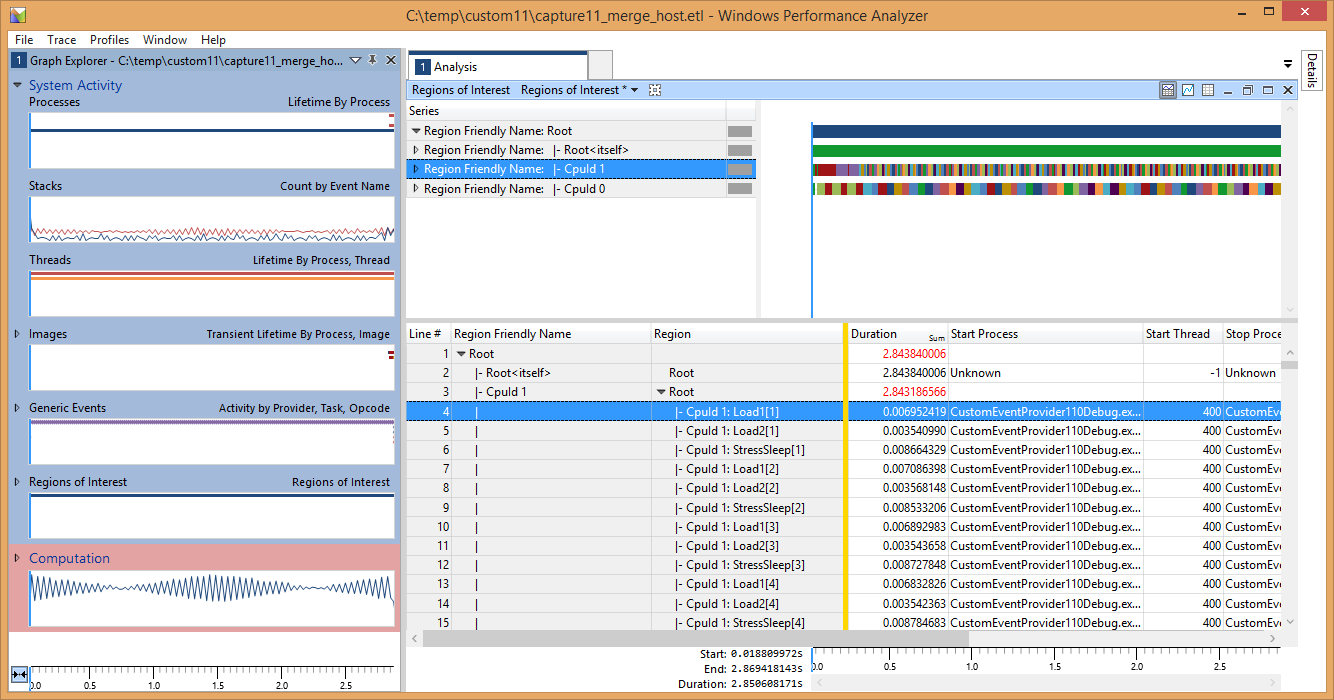
ROI を表示するには、まず地域定義ファイルを読み込む必要があります。[トレース] メニューから [トレースのプロパティ] を選択して、サンプルに付属の regional.xml 定義をロードします。



これで、Generic Events グラフの下に Region of Interest グラフが表示されるはずです。ROI グラフを分析領域までドラッグして拡大します。ツールバーのデフォルトのビュー プリセットは「Regions of Interest」になります。Region 列をテーブルに追加し (各領域が一意の色になるように)、Root ノードを展開します。次と同じように表示されるはずです。

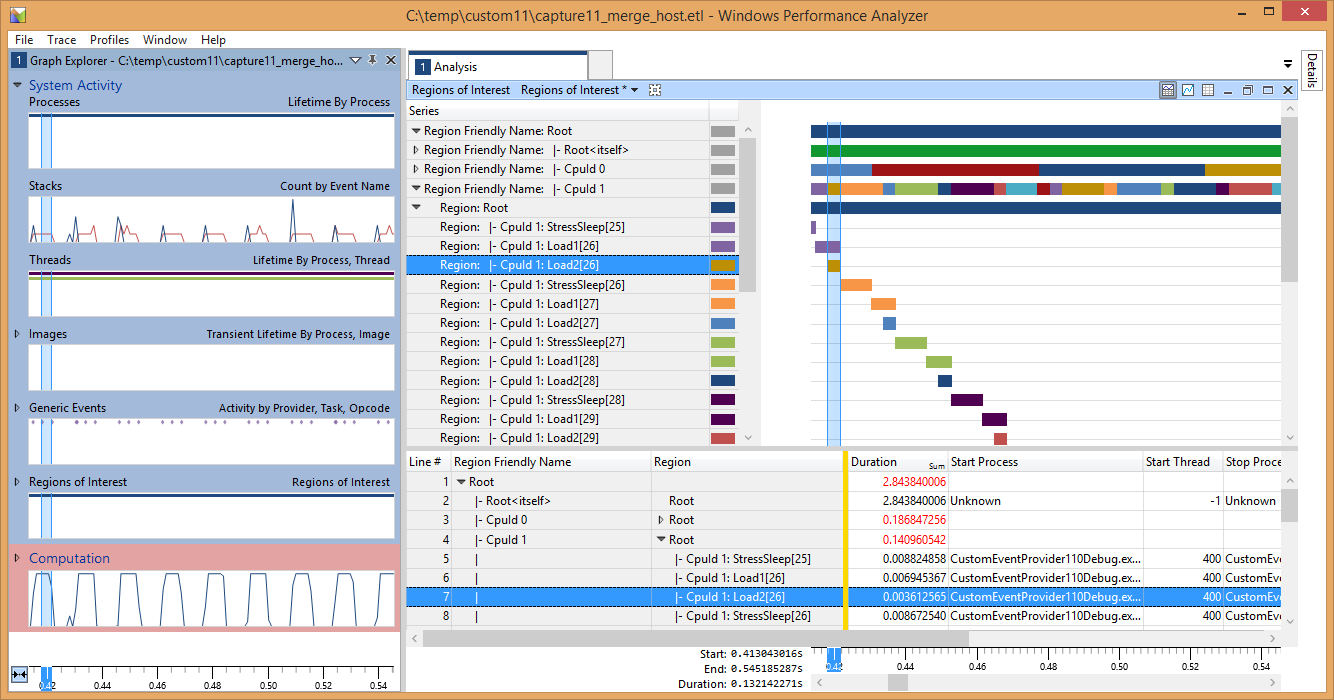


テーブルの Region ノードを展開すると、それぞれの括弧に関する情報が得られます。

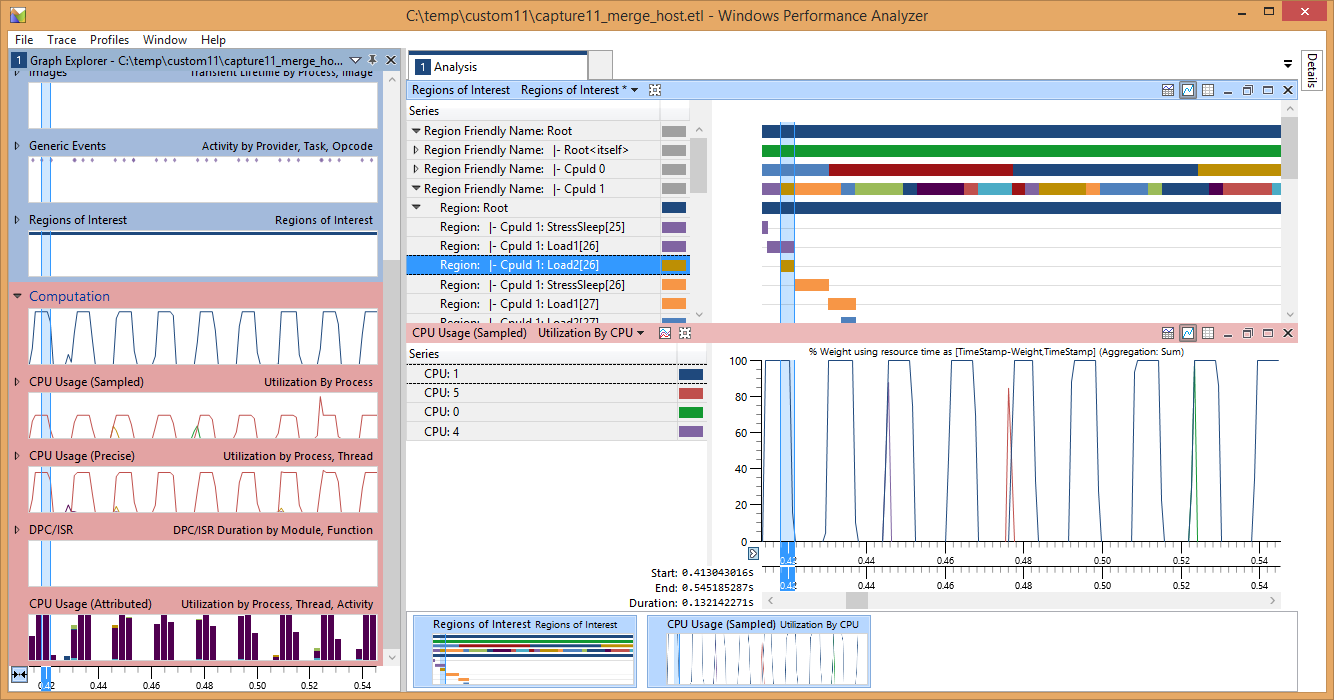


ご覧のとおり、**ETWScopedEvent()** の呼び出しで提供されたラベルがここに表示されます (番号は特定のラベルのインスタンスです)。

グラフを拡大すると、各地域のタイムラインが別々に表示されます:



さらなる利点として、これを有効にしていれば、リージョンとサンプリングキャプチャからのデータの関連性を比較できます:



# 既知の問題点

ホスト　PC　でイベントプロバイダの　GUID　を解決するので、その　ID　を持つ　ETW　プロバイダが既に登録されていない必要があります。サンプルのイベント　マニフェストから新しいイベント　マニフェストを作成する場合は、マニフェストの編集中に　ecmangen　を使用して　GUID　を再生成するか、新しい　GUID　を生成するために　guidgen.exe　(Visual Studioに付属)　を使用します。

ホスト PC で ETL ファイルが解決されると、[Generic Events] ビューにいくつかの追加のイベント プロバイダが表示されることがあります。これらは無視することができます。

# プライバシーステートメント

サンプルをコンパイルして実行すると、サンプルの実行可能ファイルの名前がMicrosoftに送信され、サンプルの使用状況の追跡に役立ちます。このデータ収集を無効にするには、Main.cppの「Sample Usage Telemetry」というラベルの付いたコードブロックを削除します。

マイクロソフトのプライバシーポリシー全般に関する詳細については、 [Microsoftのプライバシーステートメント](https://privacy.microsoft.com/ja-jp/privacystatement/)をご参照ください。