

# TOPPERS/TLVマニュアル

## (1.3rc 対応)

名古屋大学 大学院情報科学研究科  
附属組込みシステム研究センター

# はじめに

---

- ・ 本マニュアルは、TLVの基本的な使い方及び、TOPPERS/ASPカーネルまたは、TOPPERS/FMPカーネルでTLVで可視化するためのログ取得方法について解説している.
- ・ ログの標準形式への変換ルールに関しては、TLV\_convert\_rules.ppt を参照のこと.
- ・ ログの可視化のための可視化変換ルールについては、TLV\_visualize\_rules.ppt を参照のこと.

# 目次

---

- ・ ファイル一覧
- ・ 実行環境
- ・ TOPPERS/ASPカーネルとTOPPERS/FMPカーネルのトレースログの取得
- ・ 機能紹介

---

# ファイル一覧

# TOPPERS/TLVパッケージのファイル構成

フォルダ名	ファイル名	説明	フォルダ名	ファイル名	説明		
/	README.txt	TLVの簡単な紹介	sampleFiles/	asp/	asp_short.log	TOPPERS/ASPカーネルの トレースログのサンプル	
	TraceLogVisualizer.exe	TLVの本体			asp_short.res		
					asp_short.tlv		
					asp_long.log		
					asp_long.res		
					asp_long.tlv		
doc/	TLV.pdf	TLV本体のマニュアル		fmp/	fmp_short.log	TOPPERS/FMPカーネルの トレースログのサンプル	
	TLV_convert_rules.pdf	変換ルールのマニュアル			fmp_short.res		
	TLV_visualize_rules.pdf	可視化ルールのマニュアル			fmp_short.tlv		
convertRules/	asp.cnv	TOPPERS/ASPカーネルの変換ルール			fmp_long.log		
	fmp.cnv	TOPPERS/FMPカーネルの変換ルール	fmp_long.res				
	tecs.cnv	TECSの変換ルール	fmp_long.tlv				
resourceHeaders/	asp.res	TOPPERS/ASPカーネルのリソースファイル	tecs/	tecs.log	TECSのトレースログのサンプル		
	fmp.res	TOPPERS/FMPカーネルのリソースファイル		tecs.res			
	tecs.res	TECSのリソースファイル		tecs.tlv			
visualizeRules/	asp_rules.viz	TOPPERS/ASPカーネルの可視化ルール		logtrace/	asp/	kernel_fncode.h	TOPPERS/ASPカーネルの ログトレースモジュール
	asp_shapes.viz	TOPPERS/ASPカーネルの図形定義				tlv.tf	
	fmp_rules.viz	TOPPERS/FMPカーネルの可視化ルール				trace_config.c	
	fmp_shapes.viz	TOPPERS/FMPカーネルの図形定義	trace_config.h				
	tecs.viz	TECSの可視化ルール	fmp/		tlv.tf	TOPPERS/FMPカーネルの ログトレースモジュール	
	toppers_rules.viz	TOPPERS共通の可視化ルール					
	toppers_shapes.viz	TOPPERS共通の図形定義					

---

# 実行環境

# 実行環境

---

## TLV実行環境

- ・WindowsXP / Vista
- ・Microsoft .NET Framework 3.5 をインストールすること.

## ログ取得対象

- ・TOPPERS/ASPカーネル 1.3.2/1.3.1のトレースログ
- ・TOPPRES/FMPカーネル 0.B.0 のトレースログ
- ・TECSのログ

---

# TOPPERS/ASPカーネルと TOPPERS/FMPカーネルの トレースログの取得



# TOPPERS/ASPカーネルのトレースログの取得

---

- ・TLVパッケージのlogtrace/asp 以下のファイルをカーネルの asp/arch/logtrace に置く.
- ・asp/kernel/kernel.tf の最後に以下を追加.

```
最後の行 : $INCLUDE"arch/logtrace/tlv.tf"$
```

- ・対象プログラムのMakefileを編集してトレースログを有効にする.

```
92行目 : ENABLE_TRACE = true
```

# TOPPERS/ASPカーネルのトレースログの取得

- ・ asp/doc/user.txt の 11.5トレースログ記録のサンプルコードの使用方法を参照して、ログの取得と出力を行う。
- ・ トレースログ記録の使用法の一例として、システム起動時にトレースログの記録を開始し、システム終了時に記録したトレースログをダンプするためには、システムコンフィギュレーションファイル(.cfg)に次のような記述を追加する。

```
#include "logtrace/trace_config.h"
ATT_INI({ TA_NULL, TRACE_AUTOSTOP, trace_initialize });
ATT_TER({ TA_NULL, target_fput_log, trace_dump });
```

- ・ ここで、初期化ルーチン (trace\_initialize) への引数は、初期化直後のトレースログの動作モードを指定するものである。指定できる動作モードについては、arch/logtrace/trace\_config.h中のコメントに説明がある。
- ・ 終了処理ルーチン (trace\_dump) は、記録されたトレースログをターゲット依存の低レベル出力機能 (target\_fput\_log) を利用してダンプするためのものである。トレースログを別の方法で取り出す場合には、終了処理ルーチンを登録する必要はない。

# TOPPERS/ASPカーネルのトレースログの取得

---

trace\_config.h をログを取得する環境に合わせて変更する

- ・ バッファサイズ
  - ・ TCNT\_TRACE\_BUFFER
- ・ 時刻取得ルーチン
  - ・ ターゲット依存で時刻を取得したい場合は, TRACE\_GET\_TIM に定義する.
- ・ 取得するログトレース
  - ・ 取得したいログトレース以外はコメントアウトする.
  - ・ 例) loc\_cpuのログを取らない場合は, LOG\_LOC\_CPU\_ENTER() と LOG\_LOC\_CPU\_LEAVE(ercd) のマクロをコメントアウトする.  
//#define LOG\_LOC\_CPU\_ENTER()  
//#define LOG\_LOC\_CPU\_LEAVE(ercd)

# TOPPERS/ASPカーネルのトレースログの取得

---

## 各状態表示のための必須のログトレース

- ・ タスクの状態表示
  - ・ LOG\_TSKSTAT (p\_tcb)
- ・ ハンドラの実行表示
  - ・ LOG\_INH\_ENTER (inhno) / LOG\_INH\_LEAVE (inhno),  
LOG\_ISR\_ENTER (intno) / LOG\_ISR\_LEAVE (intno),  
LOG\_CYC\_ENTER (p\_cycpcb) / LOG\_CYC\_LEAVE (p\_cycpcb),  
LOG\_ALM\_ENTER (p\_almbcb) / LOG\_ALM\_LEAVE (p\_almbcb),  
LOG\_EXC\_ENTER (excno) / LOG\_EXC\_LEAVE (excno),  
LOG\_TEX\_ENTER (texptn) / LOG\_TEX\_LEAVE (texptn)
- ・ システムコール表示
  - ・ 有効にしたログトレースが表示される.

# TOPPERS/FMPカーネルのトレースログの取得

- ・TLVパッケージのlogtrace/fmp 以下のファイルをカーネルの fmp/arch/logtrace に置く.
- ・fmp/kernel/kernel.tf の最後に以下を追加.

最後の行 : `$INCLUDE"arch/logtrace/tlv.tf"$`

- ・対象プログラムのMakefileを編集してトレースログを有効にする.

•92行目 : `ENABLE_TRACE = true`

# TOPPERS/FMPカーネルのトレースログの取得

- ・ fmp/doc/user.txt の 10.3トレースログ記録のサンプルコードの使用方を参照して、ログの取得と出力を行う。
- ・ トレースログ記録の使用法の一例として、システム起動時にトレースログの記録を開始し、システム終了時に記録したトレースログをダンプするためには、システムコンフィギュレーションファイル(.cfg)に次のような記述を追加する。

```
#include "logtrace/trace_config.h"
ATT_INI({ TA_NULL, TRACE_AUTOSTOP, trace_initialize });
ATT_TER({ TA_NULL, target_fput_log, trace_dump });
```

- ・ ここで、初期化ルーチン (trace\_initialize) への引数は、初期化直後のトレースログの動作モードを指定するものである。指定できる動作モードについては、arch/logtrace/trace\_config.h中のコメントに説明がある。
- ・ 終了処理ルーチン (trace\_dump) は、記録されたトレースログをターゲット依存の低レベル出力機能 (target\_fput\_log) を利用してダンプするためのものである。トレースログを別の方法で取り出す場合には、終了処理ルーチンを登録する必要はない。

# TOPPERS/FMPカーネルのトレースログの取得

---

trace\_config.h をログを取得する環境に合わせて変更する.

- ・ バッファサイズ
  - ・ TCNT\_TRACE\_BUFFER
- ・ 時刻取得ルーチン
  - ・ ターゲット依存で時刻を取得したい場合は, TRACE\_GET\_TIM に定義する.
- ・ 取得するログトレース
  - ・ 取得したいログトレース以外はコメントアウトする.
  - ・ 例) loc\_cpuのログを取らない場合は, LOG\_LOC\_CPU\_ENTER() とLOG\_LOC\_CPU\_LEAVE(ercd)のマクロをコメントアウトする.  
//#define LOG\_LOC\_CPU\_ENTER()  
//#define LOG\_LOC\_CPU\_LEAVE(ercd)

# TOPPERS/FMPカーネルのトレースログの取得

---

## 各状態表示のための必須のログトレース

- ・ タスクの状態表示
  - ・ LOG\_TSKSTAT (p\_tcb)
- ・ ハンドラの実行表示
  - ・ LOG\_INH\_ENTER (inhno) / LOG\_INH\_LEAVE (inhno),  
LOG\_ISR\_ENTER (intno) / LOG\_ISR\_LEAVE (intno),  
LOG\_CYC\_ENTER (p\_cycpcb) / LOG\_CYC\_LEAVE (p\_cycpcb),  
LOG\_ALM\_ENTER (p\_almbcb) / LOG\_ALM\_LEAVE (p\_almbcb),  
LOG\_EXC\_ENTER (excno) / LOG\_EXC\_LEAVE (excno),  
LOG\_TEX\_ENTER (texptn) / LOG\_TEX\_LEAVE (texptn)
- ・ システムコール表示
  - ・ 有効にしたログトレースが表示される.



# TLVへの入力ファイル

---

- ・ TLVへの入力ファイルは、以下の二つであり、リソースファイルは、アプリケーションのビルド時に、トレースログファイルはアプリケーションの実行後に作成される。
  - ・ リソースファイル (kernel.res)
    - ・ アプリケーションをビルドすると、コンフィギュレーターにより、ビルドディレクトリに生成される。
  - ・ トレースログファイル (xxx.log)
    - ・ トレースログの出力をユーザーがファイルに保存する。

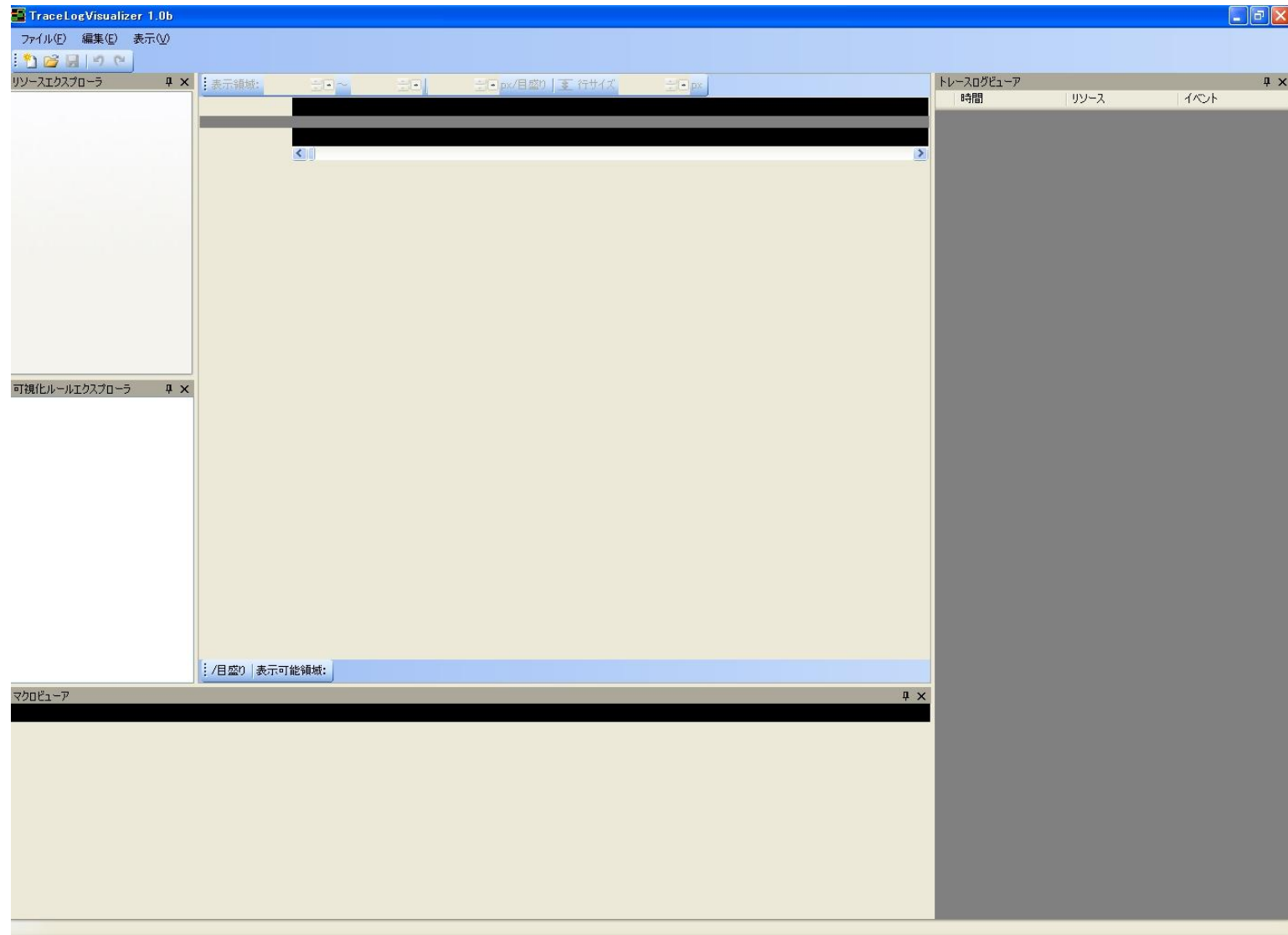
---

# 機能紹介

# 機能紹介

---

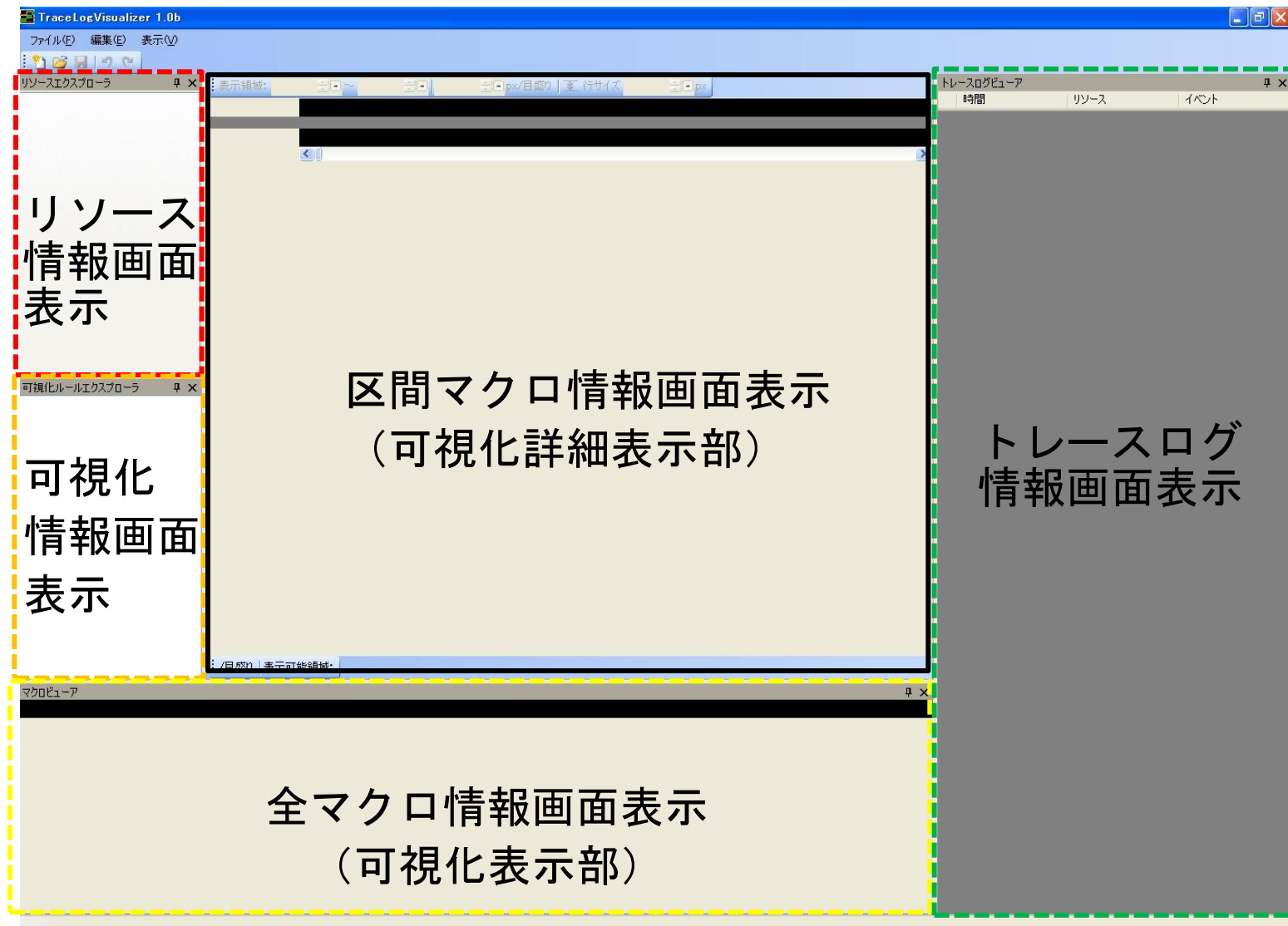
- ・ sampleフォルダに入っているTOPPERS/ASPカーネルのリソースファイルとログファイルを用いて、TLVの機能を紹介する.



# TLVメニュー一覧

---

- ・ ファイル
  - ・ 新規作成－新しいログファイルをオープン
  - ・ 開く－既存のログファイルをオープン(xxx.tlv)
- ・ 編集
- ・ 表示
  - ・ トレースログビューアー　　－トレースログ画面表示
  - ・ リソースエクスプローラ　　－リソース情報画面表示
  - ・ 可視化ルールエクスプローラ－可視化情報画面表示
  - ・ マクロビューアー(可視化表示部)－全マクロ画面表示
  - ・ 可視化詳細表示部　　－　区間マクロ情報画面表示



# ログファイルオープン

---

- ・ TLV1.0rc ログファイルオープンには、二つの方法がある.

## 1. 新規作成

kernel.res、xxx.log ファイルが必要.

「メニュー」→「ファイル」→「新規作成」をクリック

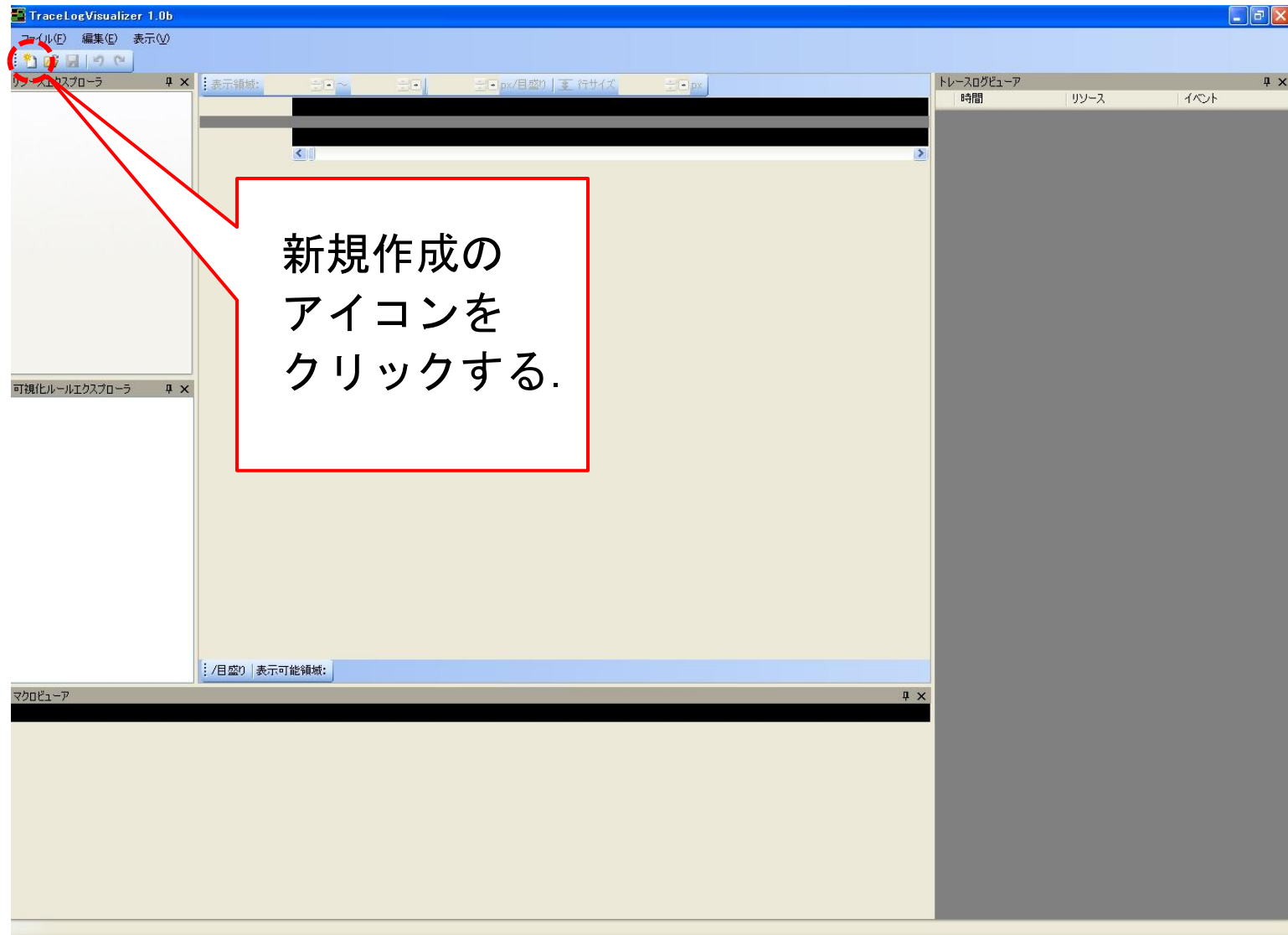
## 2. 開く（保存したものを開く）

xxx.tlv ファイルが必要.

「メニュー」→「ファイル」→「開く」をクリック

# 1. 新規作成一スタート

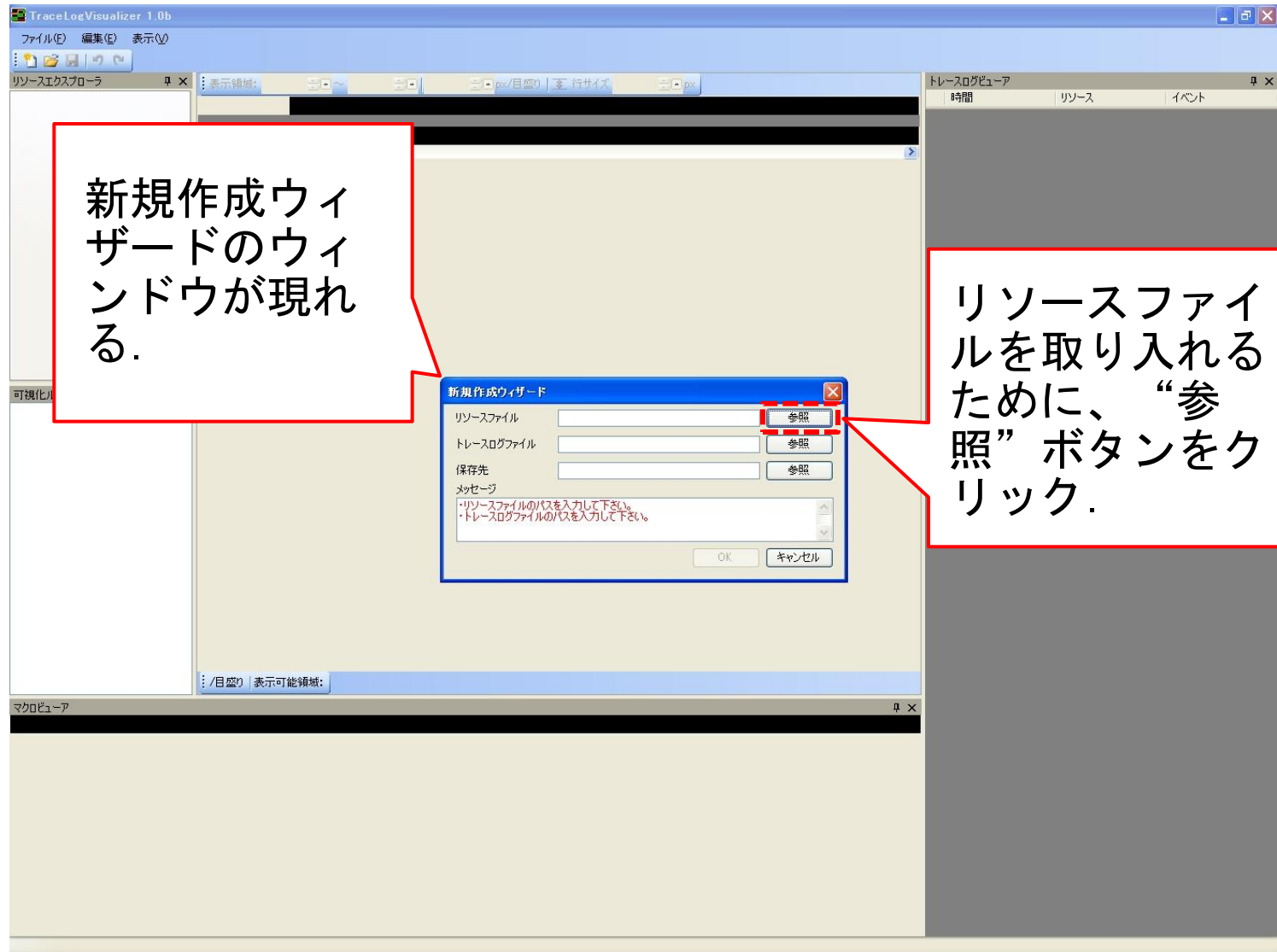
TLV初期画面



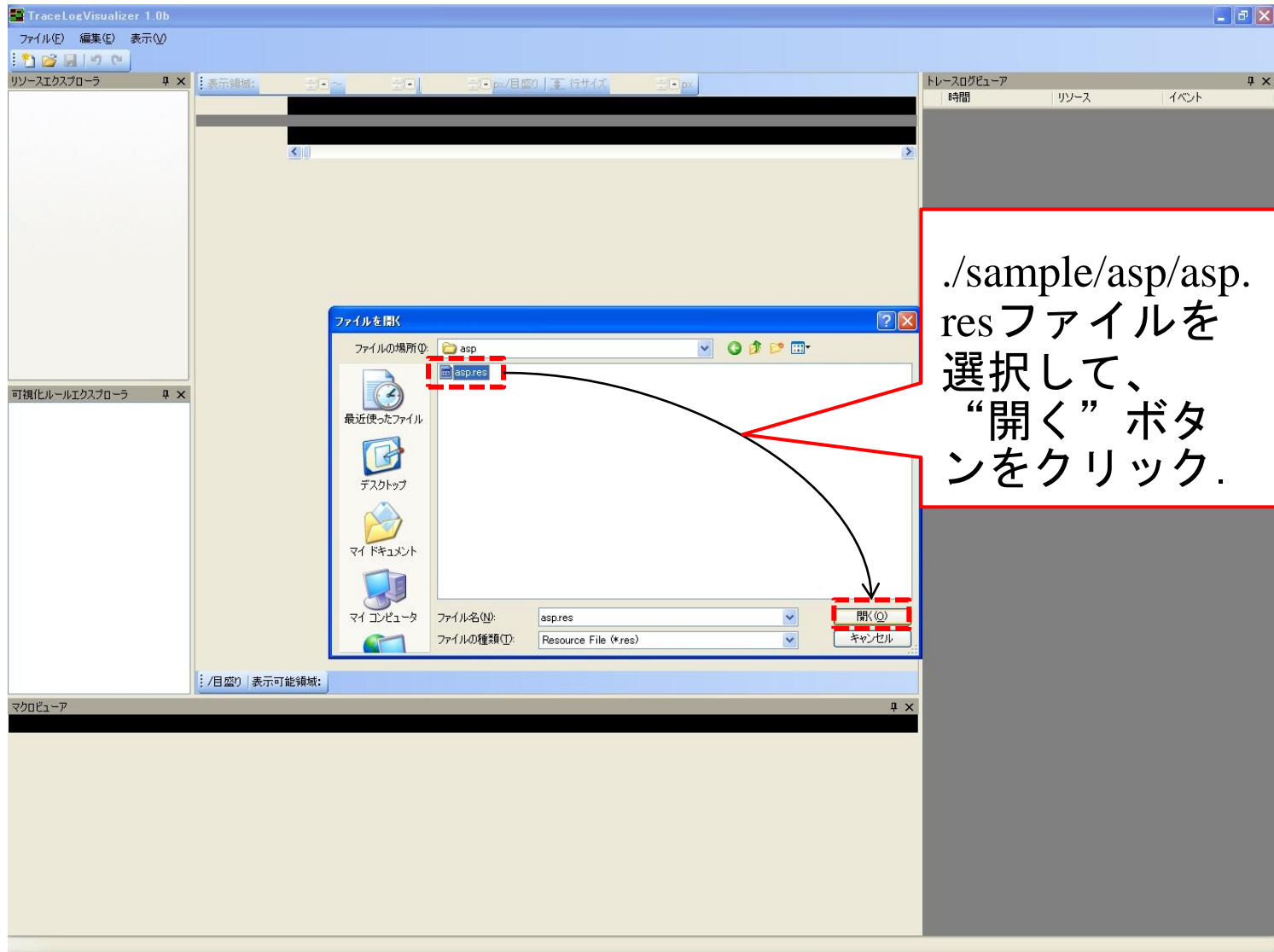
新規作成の  
アイコンを  
クリックする。



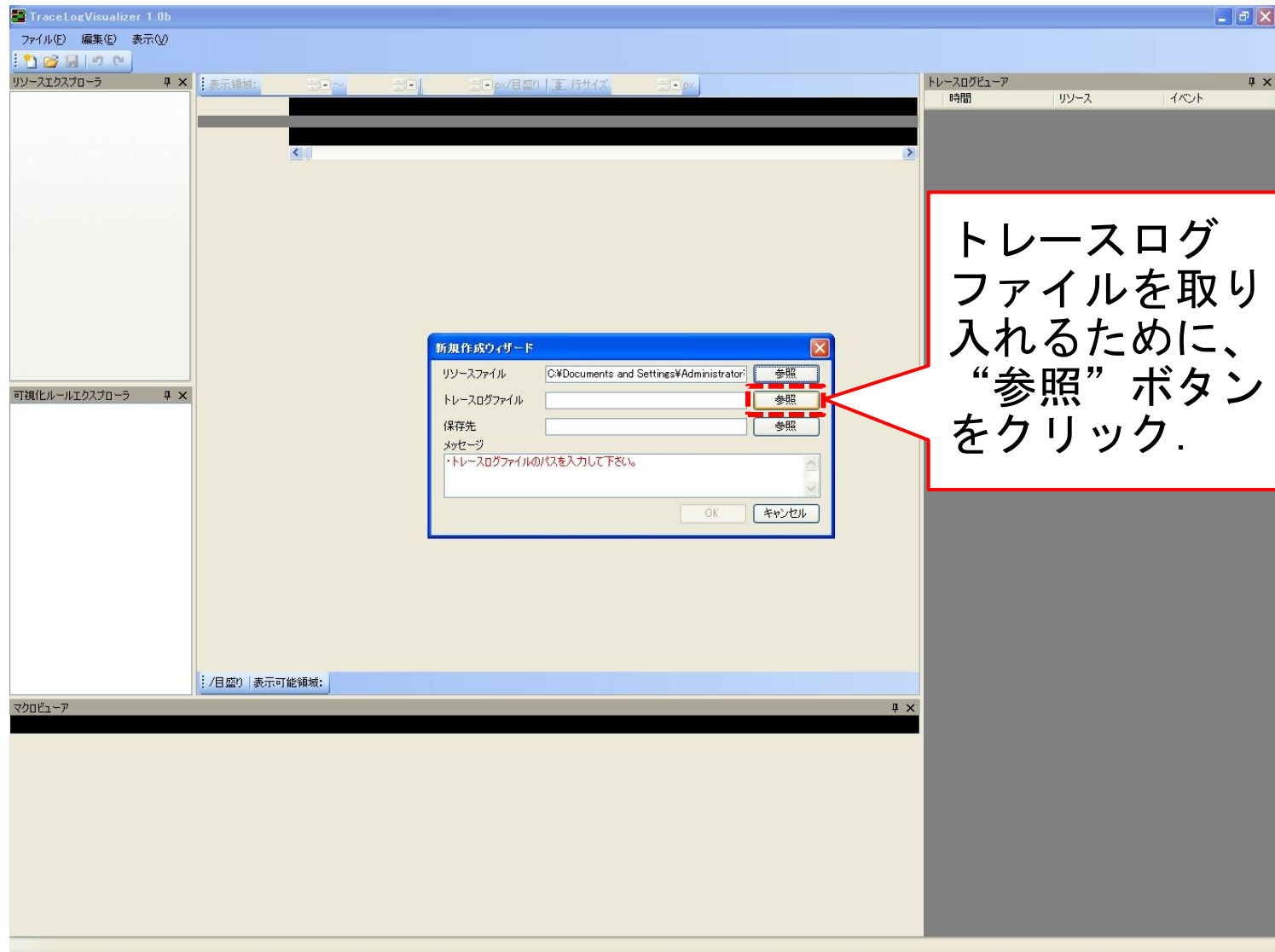
# 新規作成ーリソースファイルの参照



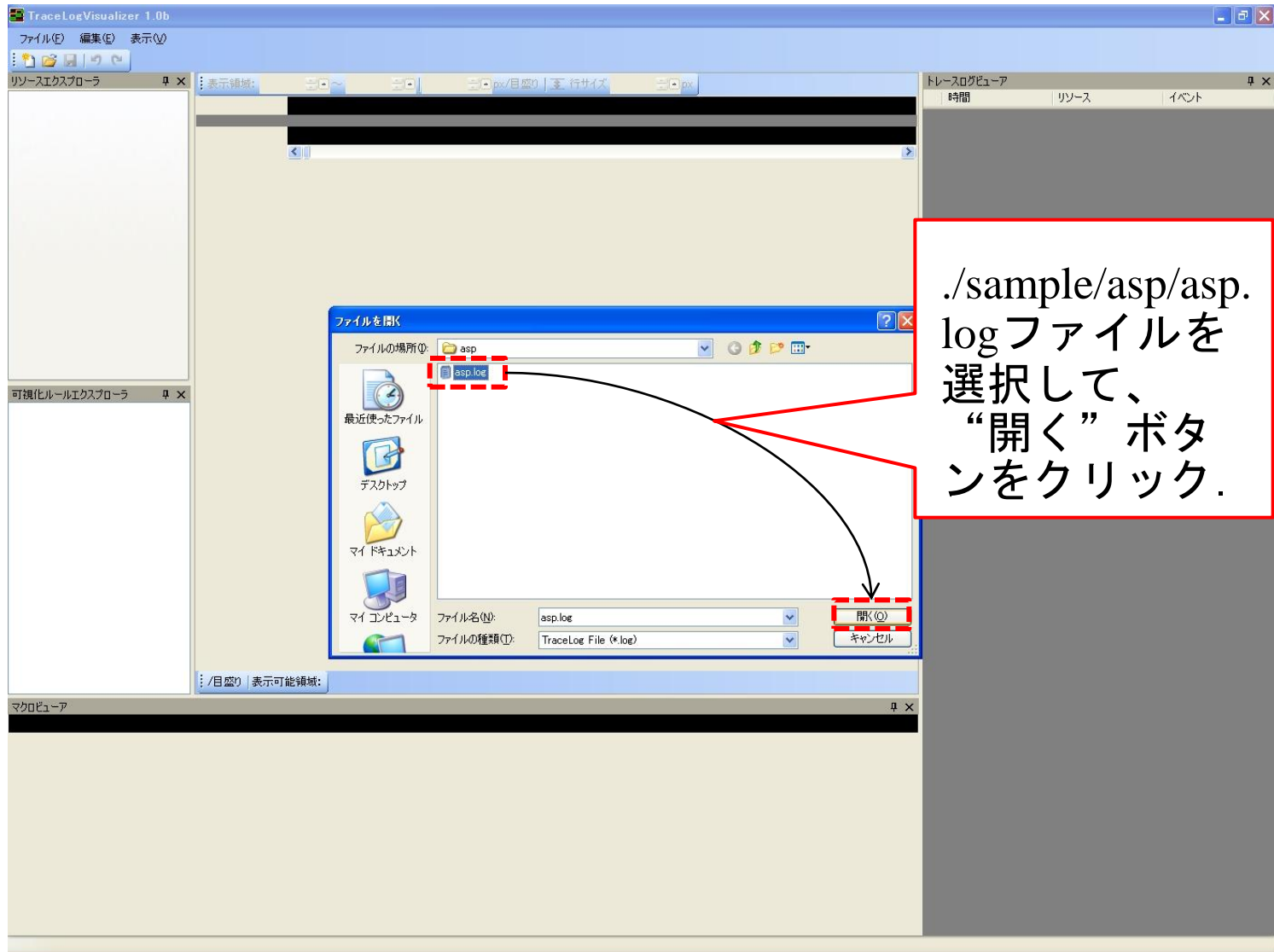
# 新規作成ーリソースファイルの参照



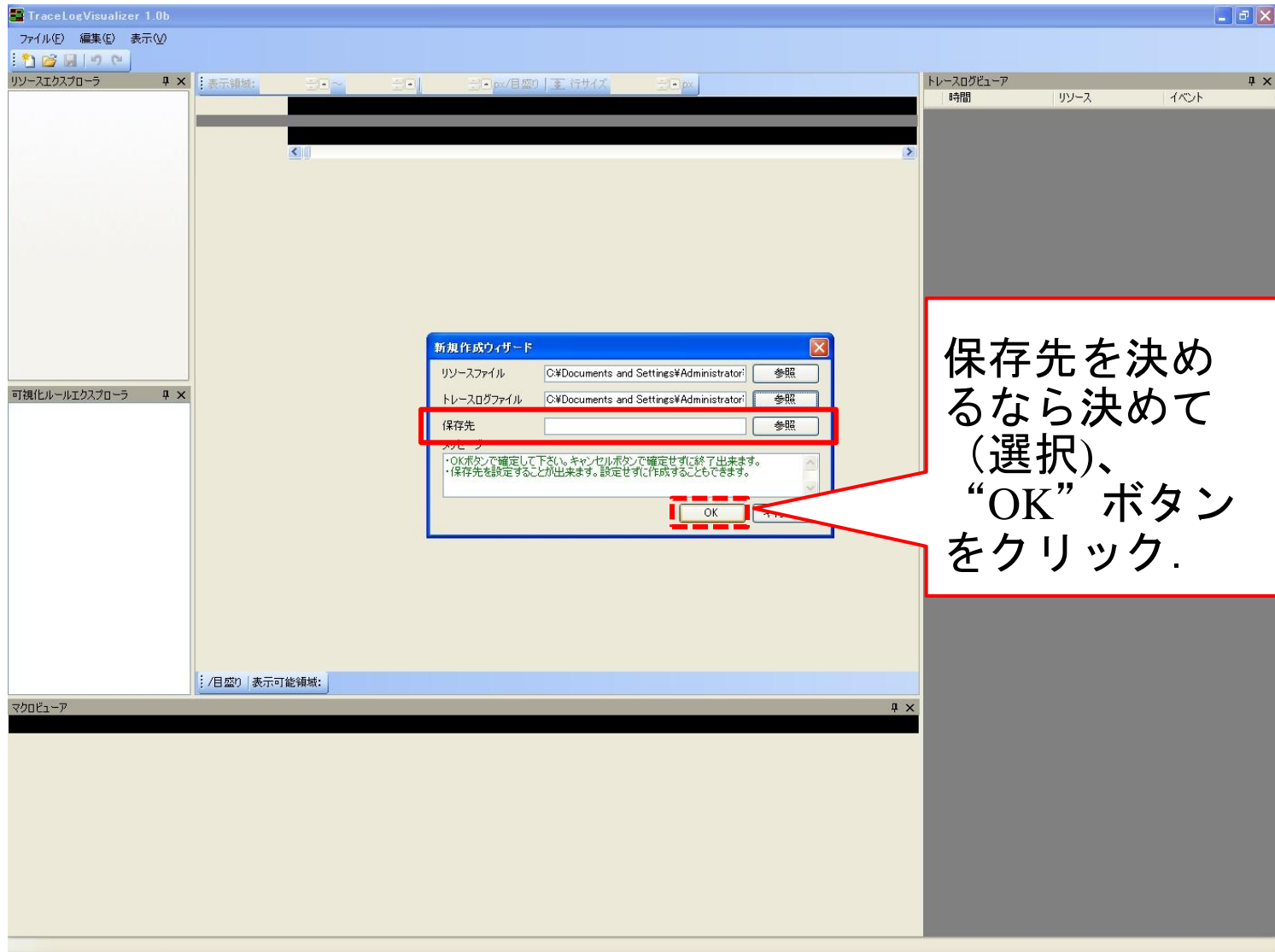
# 新規作成ートレースログファイルの参照



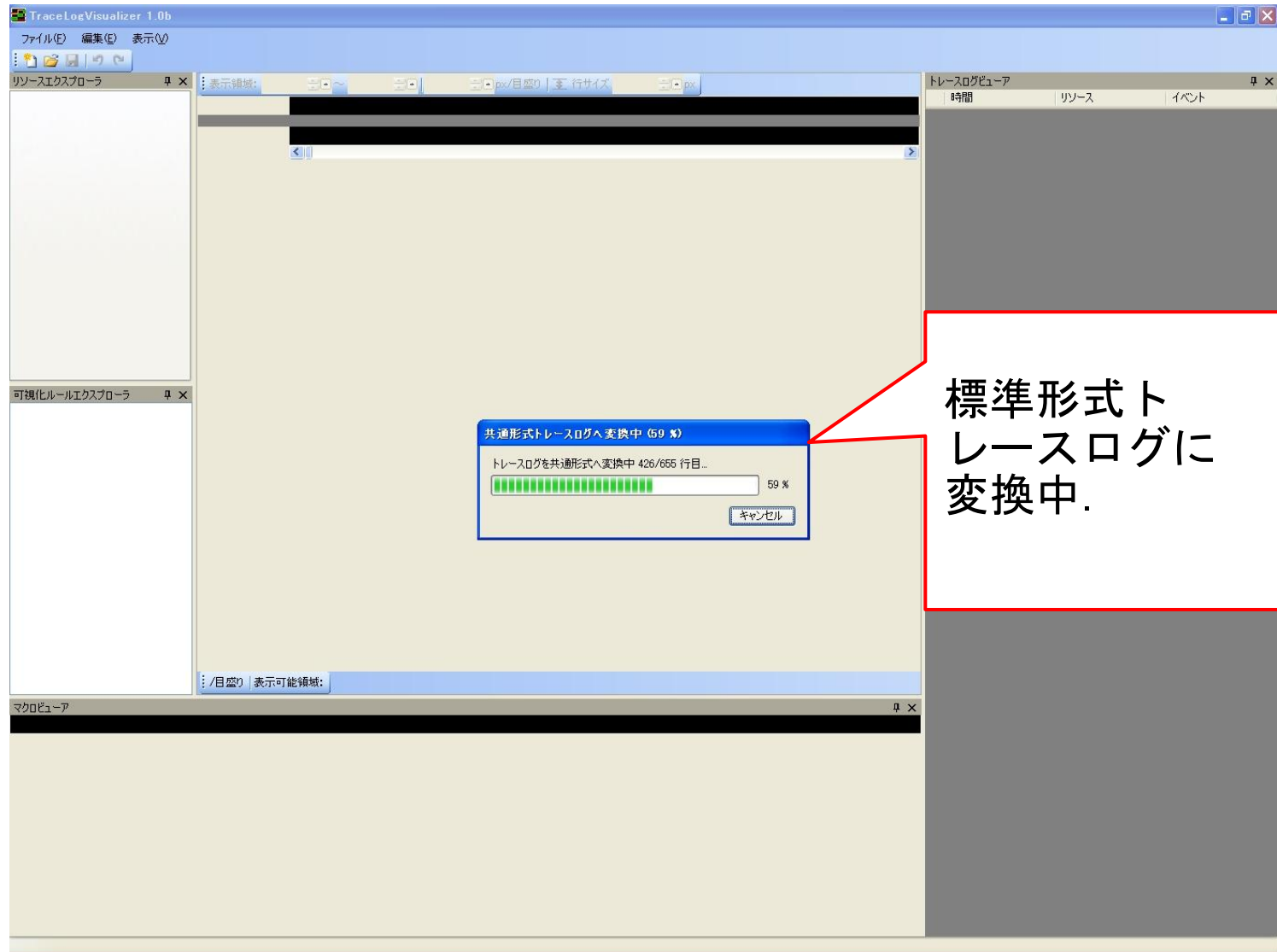
# 新規作成ートレースログファイルの参照



# 新規作成－保存先の設定



# 新規作成一標準形式トレースログに変換



標準形式  
トレースログに  
変換中.

# ログファイルオープン

---

- ・ TLV1.0ログファイルオープンには二つの方法がある.

## 1. 新規作成

kernel.res、xxx.log ファイルが必要.

「メニュー」→「ファイル」→「新規作成」をクリック

## 2. 開く（保存したものを開く）

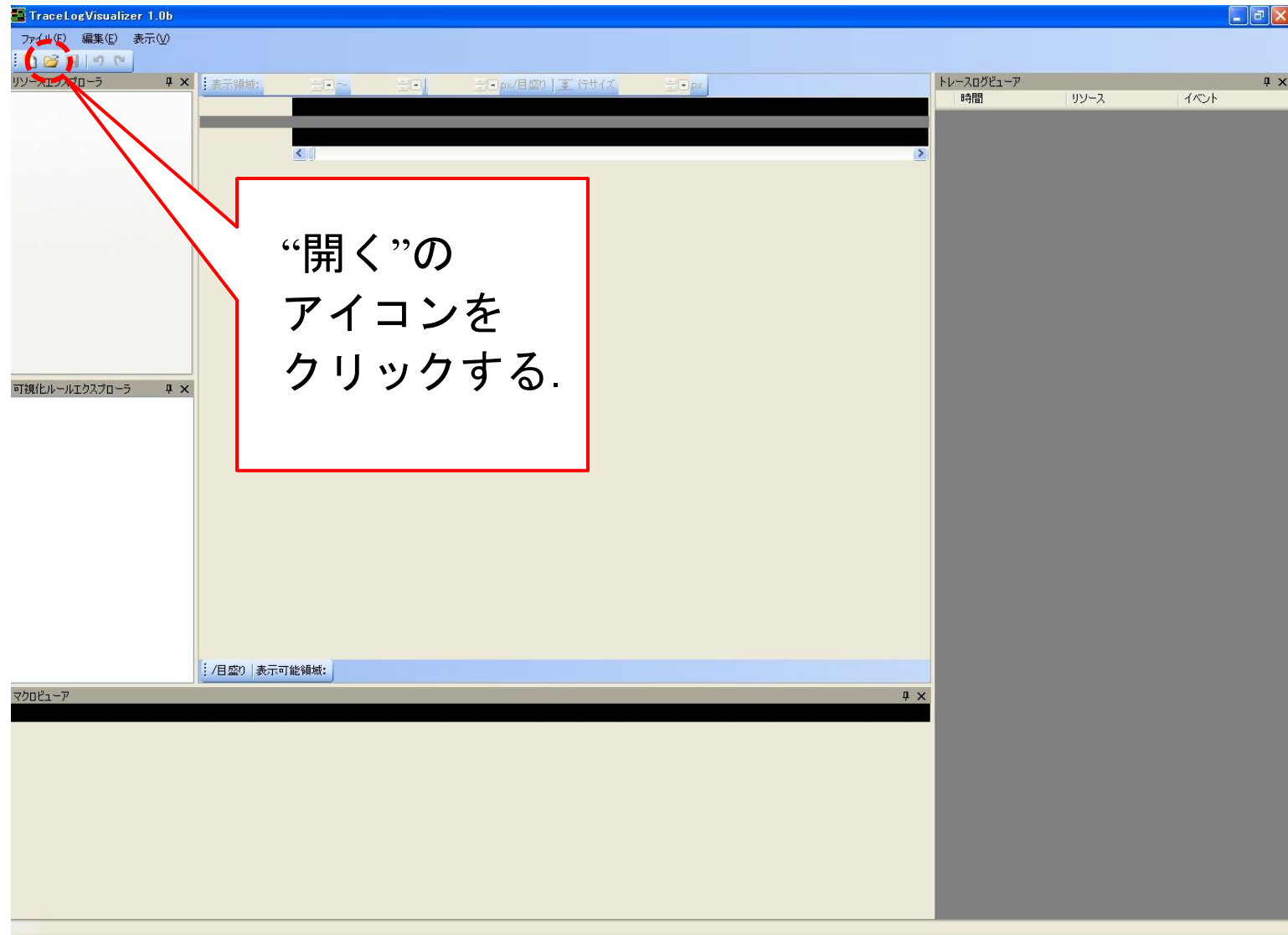
xxx.tlv ファイルが必要.

「メニュー」→「ファイル」→「開く」をクリック

- ・ 今回は、 sampleフォルダに入っているファイルを例に説明する.

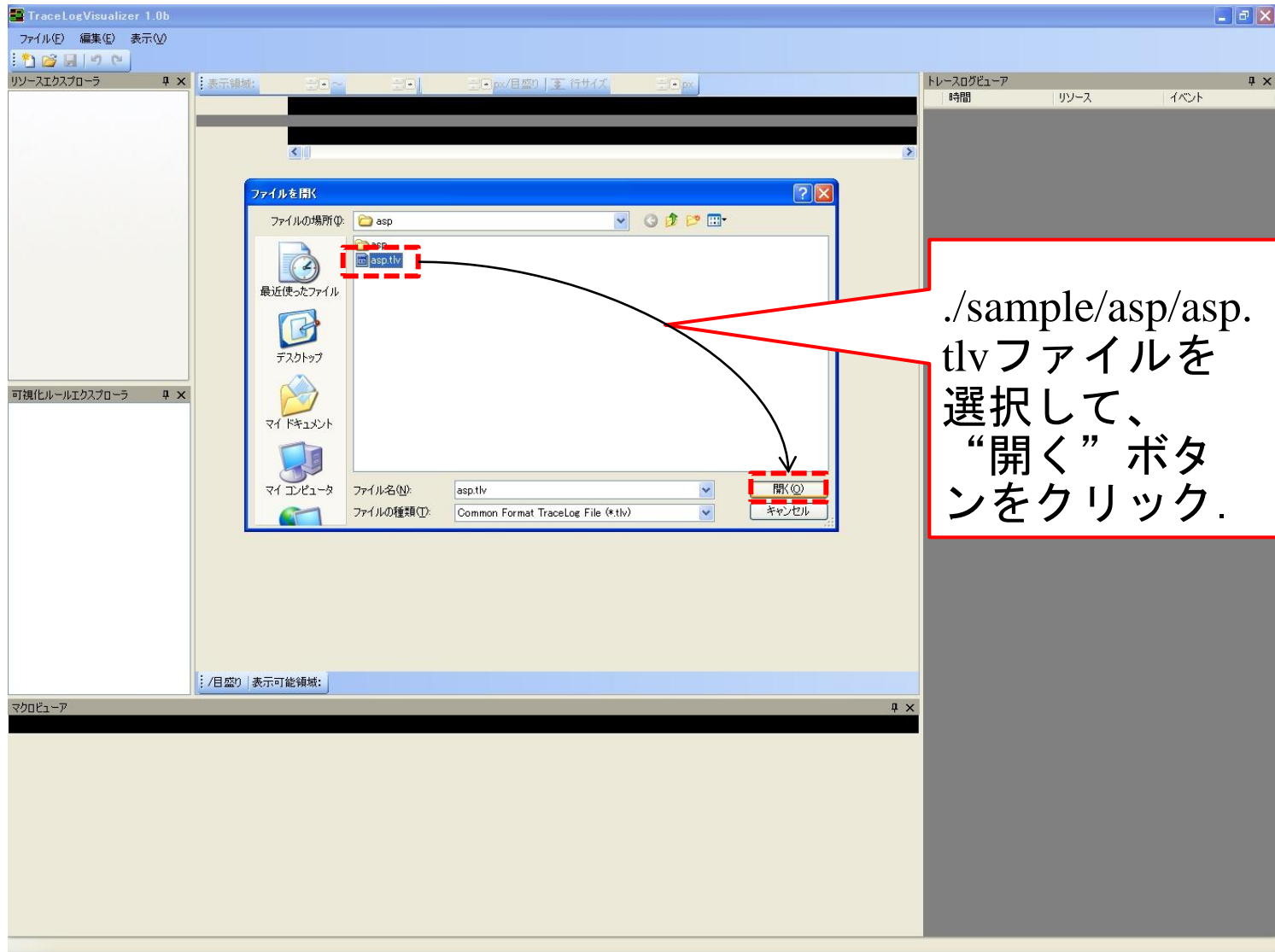
## 2. 開く(保存したものを開く)

TLV初期画面



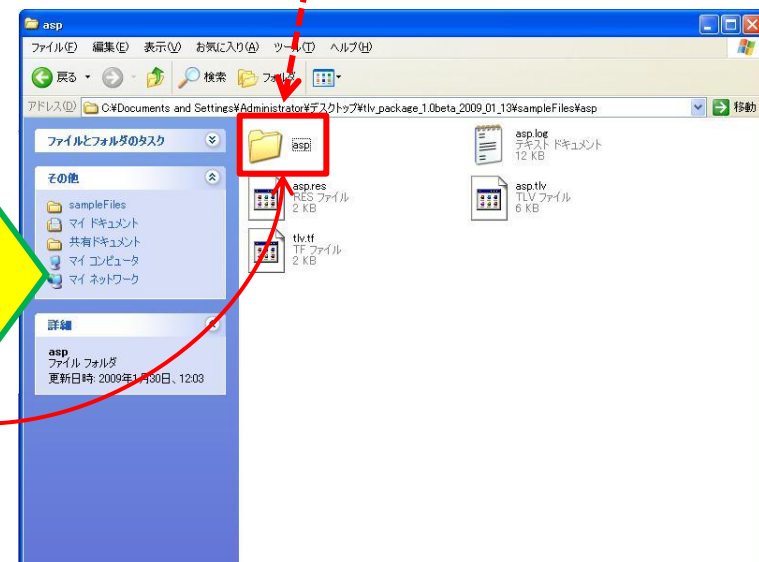
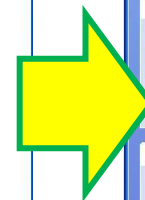
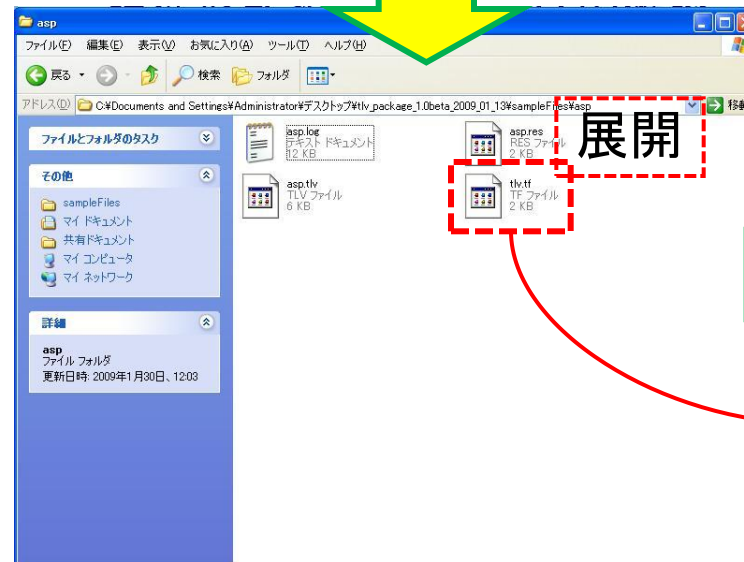
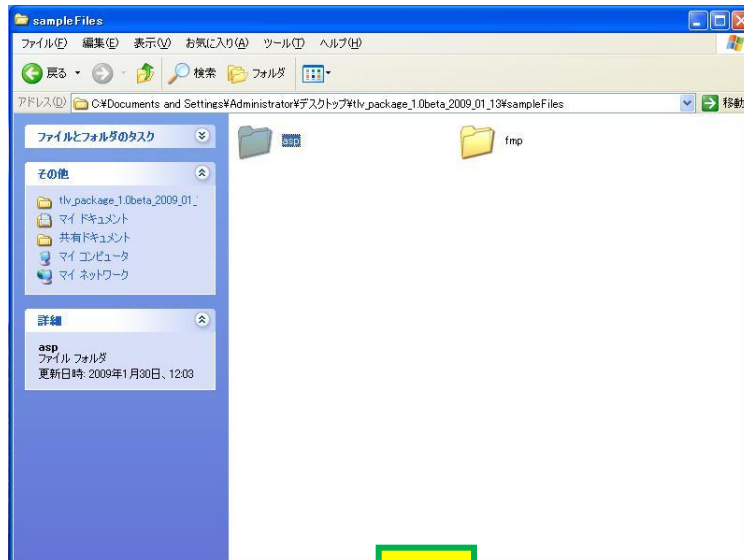


## 2. 開く(保存したものを開く)

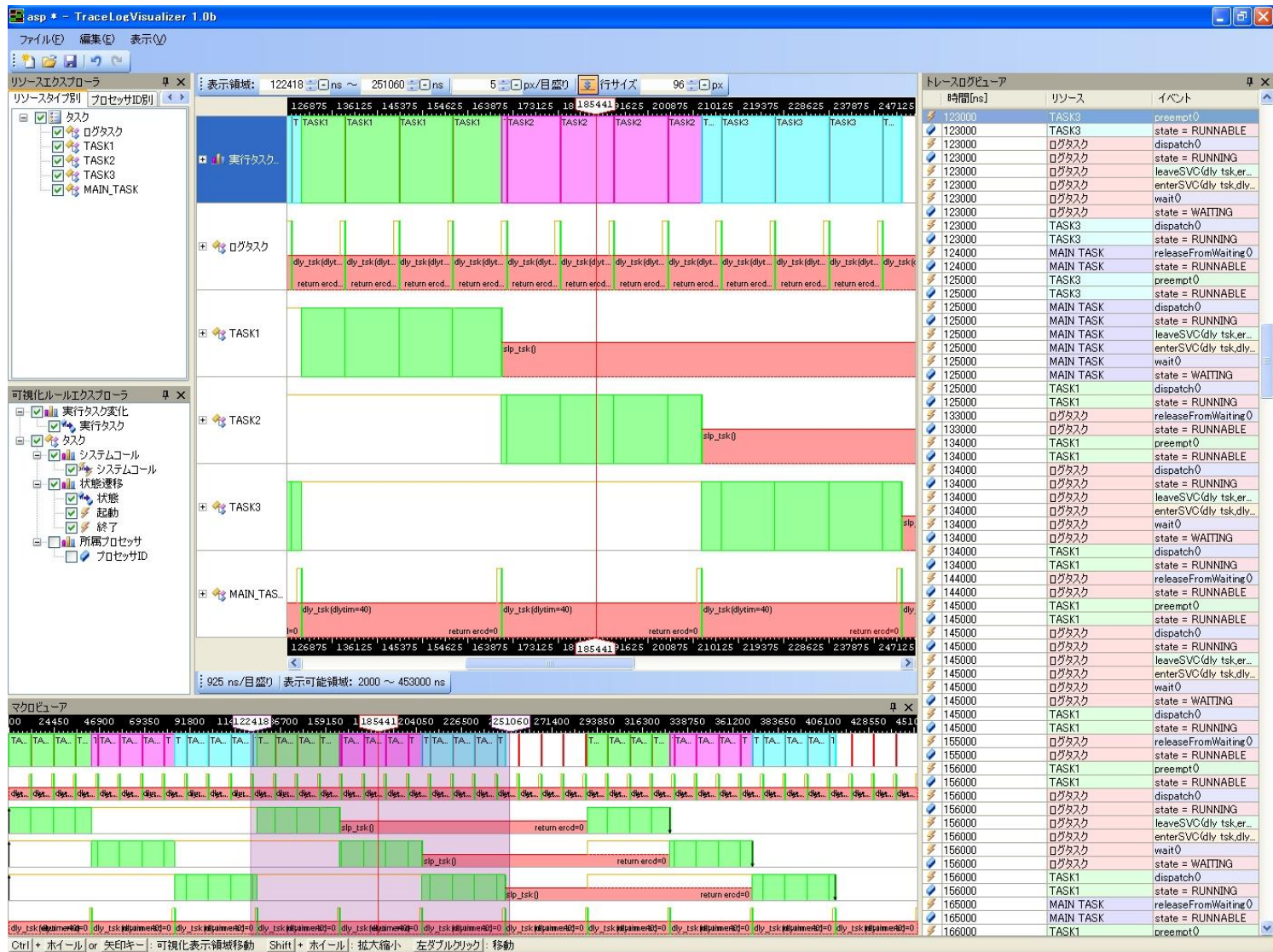


# 標準形式変換されたログの確認

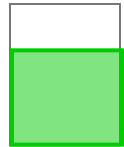
- 標準形式変換されたログの確認  
TLVを実行させ、現在の表示情報を保存すると、asp.tlvが指定したフォルダに新しく作られる。  
asp.tlvを解凍すると、asp.log、asp.res、asp.viz、asp.settingが出る。このように、解凍することで標準形式に変換されたasp.logを確認することができる。  
\* フォルダを作っておく。



# サンプルファイルを開いている画面



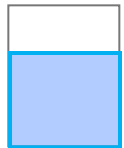
# 可視化表示部一波形表示



実行状態  
(タスクコンテキスト)



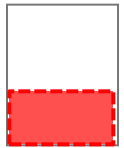
実行可能状態



非タスクコンテキスト  
(slp\_tsk, dly\_tsk以外)



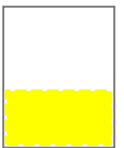
待ち状態



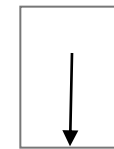
システムコール  
(slp\_tsk or dly\_tsk)



起動



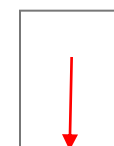
システムコール  
(slp\_tsk, dly\_tsk以外)



終了



休止状態



強制終了



# TLVの各情報表示画面



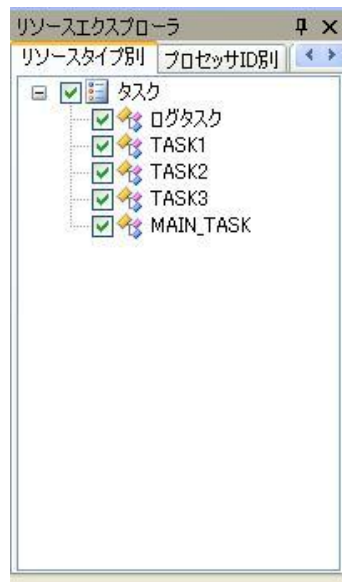
# TLVの各情報表示画面

---

1. リソースエクスプローラ : リソース情報画面表示
2. 可視化ルールエクスプローラ : 可視化情報画面表示
3. トレースログビューアー : トレースログ画面表示
4. マクロビューア: マクロ画面表示

# リソースエクスプローラ

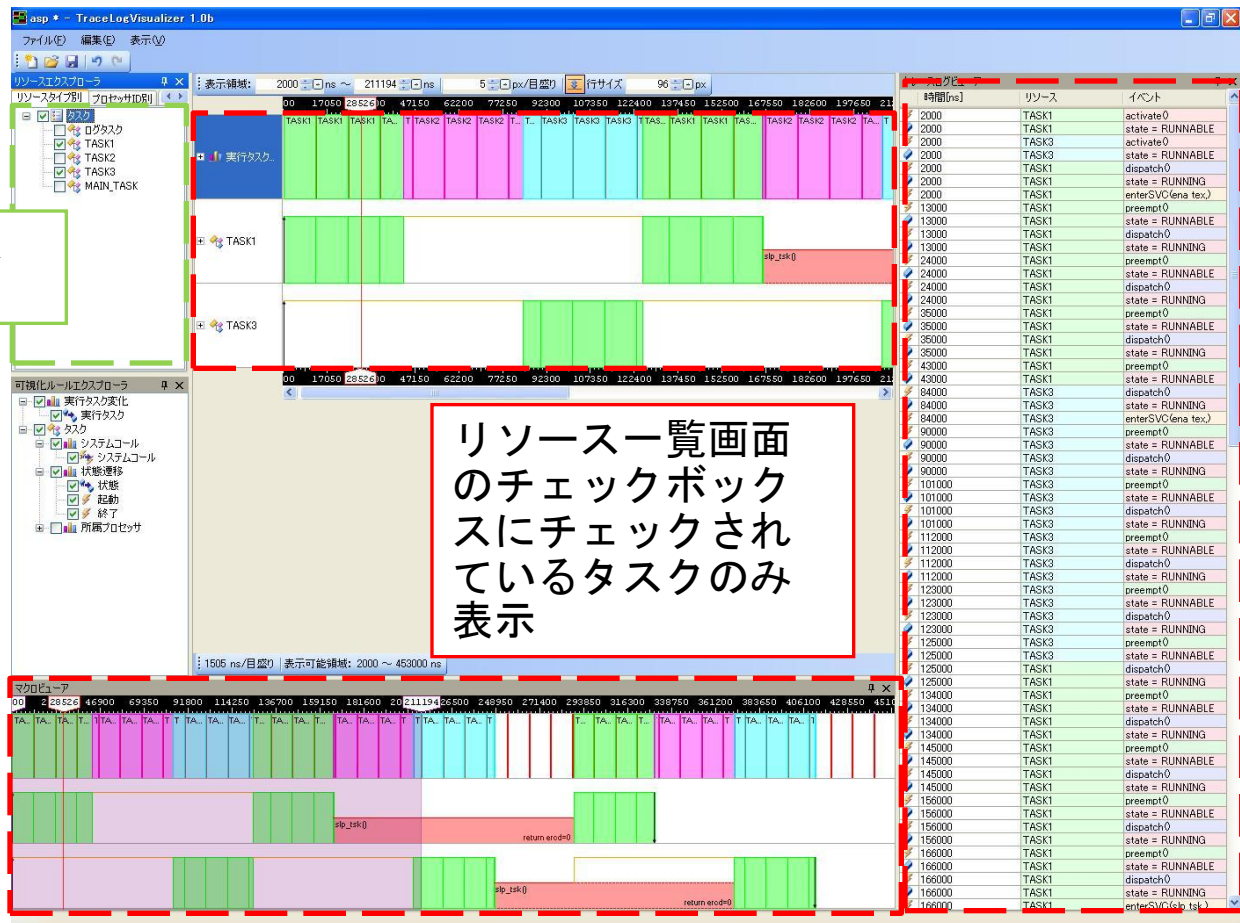
- ・ リソース情報画面表示
- ・ タブをクリックして変えることにより、リソースタイプ別、優先度別にリソースを見ることができる.
- ・ プロセッサIDはFMPの場合だけ表示
  - ・ リソースタイプ別
  - ・ 優先度別



# リソースエクスプローラ

- ・リソース情報画面表示部で表示するタスクのチェックボックスを選択すると可視化表示部へタスク状態が表示される

チェックボックス  
をクリックする



リソース一覧画面  
のチェックボッ  
クスにチェックされ  
ているタスクのみ  
表示



# TLVの各情報表示画面

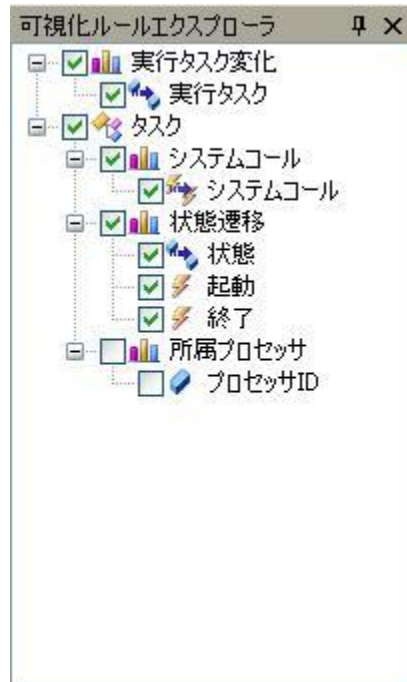
---

1. リソースエクスプローラ : リソース情報画面表示
2. 可視化ルールエクスプローラ : 可視化情報画面表示
3. トレースログビューアー : トレースログ画面表示
4. マクロビューア: マクロ画面表示

# 可視化ルールエクスプローラ

- ・可視化情報画面表示

- ・チェックボックスに  
チェックされている  
属性のみ表示.



# TLVの各情報表示画面

---

1. リソースエクスプローラ : リソース情報画面表示
2. 可視化ルールエクスプローラ : 可視化情報画面表示
3. トレースログビューアー : トレースログ画面表示
4. マクロビューア: マクロ画面表示

# トレースログビューア

時間[ms]	リソース	イベント
2000	MAIN TASK	leaveSVC(dly tsk,er...
2000	MAIN TASK	enterSVC(dly tsk,dly...
2000	MAIN TASK	wait0
2000	MAIN TASK	state = WAITING
2000	TASK1	dispatch0
2000	TASK1	state = RUNNING
2000	TASK1	enterSVC(ena tex.)
12000	ログタスク	releaseFromWaiting0
12000	ログタスク	state = RUNNABLE
13000	TASK1	preempt0
13000	TASK1	state = RUNNABLE
13000	ログタスク	dispatch0
13000	ログタスク	state = RUNNING
13000	ログタスク	leaveSVC(dly tsk,er...
13000	ログタスク	enterSVC(dly tsk,dly...
13000	ログタスク	wait0
13000	ログタスク	state = WAITING
13000	TASK1	dispatch0
13000	TASK1	state = RUNNING
23000	ログタスク	releaseFromWaiting0
23000	ログタスク	state = RUNNABLE
24000	TASK1	preempt0
24000	TASK1	state = RUNNABLE
24000	ログタスク	dispatch0
24000	ログタスク	state = RUNNING
24000	ログタスク	leaveSVC(dly tsk,er...
24000	ログタスク	enterSVC(dly tsk,dly...
24000	ログタスク	wait0
24000	ログタスク	state = WAITING
24000	TASK1	dispatch0
24000	TASK1	state = RUNNING
34000	ログタスク	releaseFromWaiting0
34000	ログタスク	state = RUNNABLE
35000	TASK1	preempt0
35000	TASK1	state = RUNNABLE
35000	ログタスク	dispatch0
35000	ログタスク	state = RUNNING
35000	ログタスク	leaveSVC(dly tsk,er...
35000	ログタスク	enterSVC(dly tsk,dly...
35000	ログタスク	wait0
35000	ログタスク	state = WAITING
35000	TASK1	dispatch0
35000	TASK1	state = RUNNING
42000	MAIN TASK	releaseFromWaiting0
42000	MAIN TASK	state = RUNNABLE
43000	TASK1	preempt0
43000	TASK1	state = RUNNABLE
43000	MAIN TASK	dispatch0
43000	MAIN TASK	state = RUNNING
43000	MAIN TASK	leaveSVC(dly tsk,er...
43000	MAIN TASK	enterSVC(dly tsk,dly...
43000	MAIN TASK	wait0
43000	MAIN TASK	state = WAITING
43000	TASK2	dispatch0
43000	TASK2	state = RUNNING
43000	TASK2	enterSVC(ena tex.)
45000	ログタスク	releaseFromWaiting0
45000	ログタスク	state = RUNNABLE
46000	TASK2	preempt0
46000	TASK2	state = RUNNABLE
46000	ログタスク	dispatch0

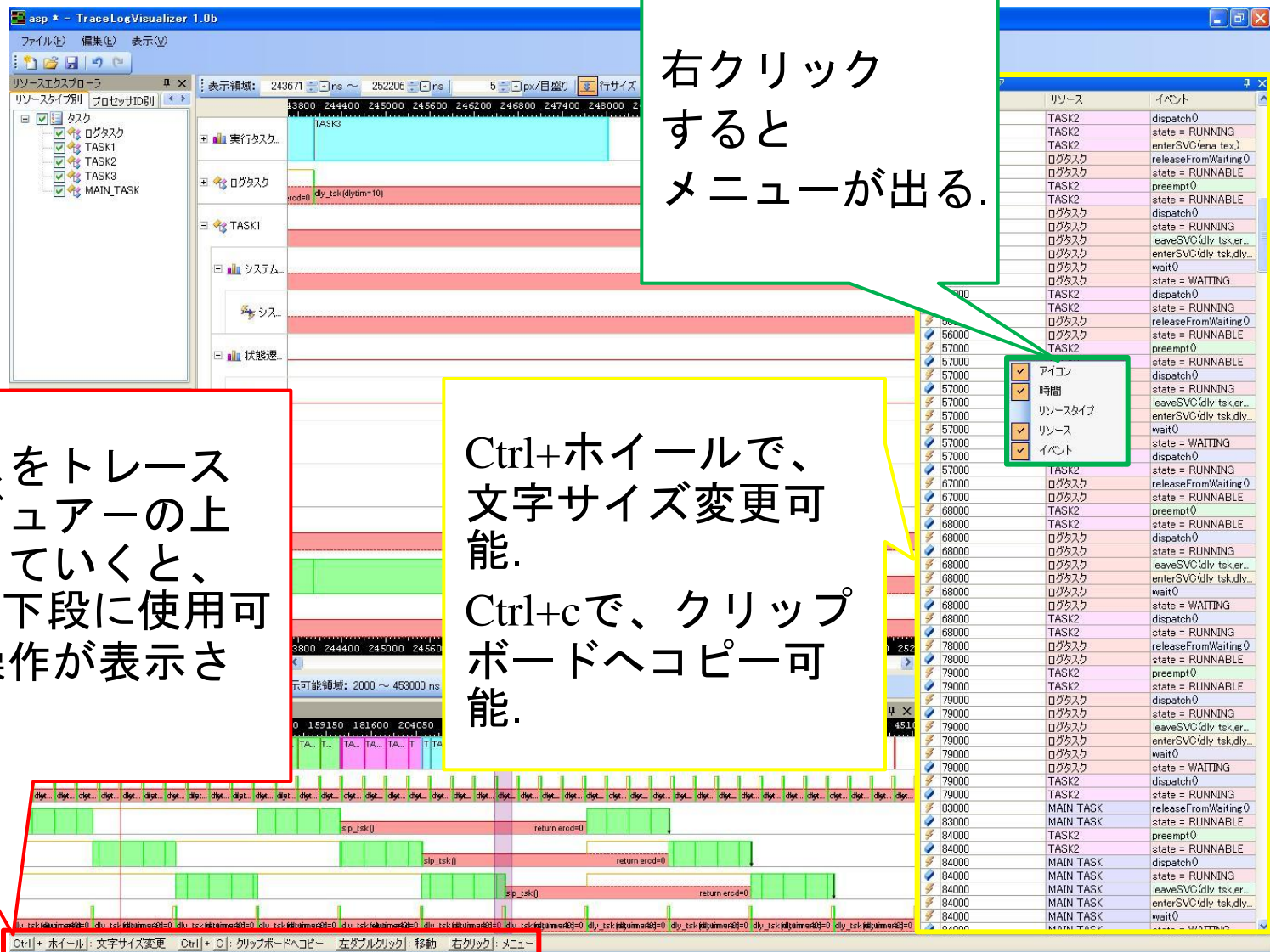
- ・トレースログ画面表示

- ・ソート

時間、リソース、イベントなどのタイトル部分をクリックするとソート可能.

- ・左ダブルクリックで、クリックしたところにマーカが移動.

# トレースログビューア



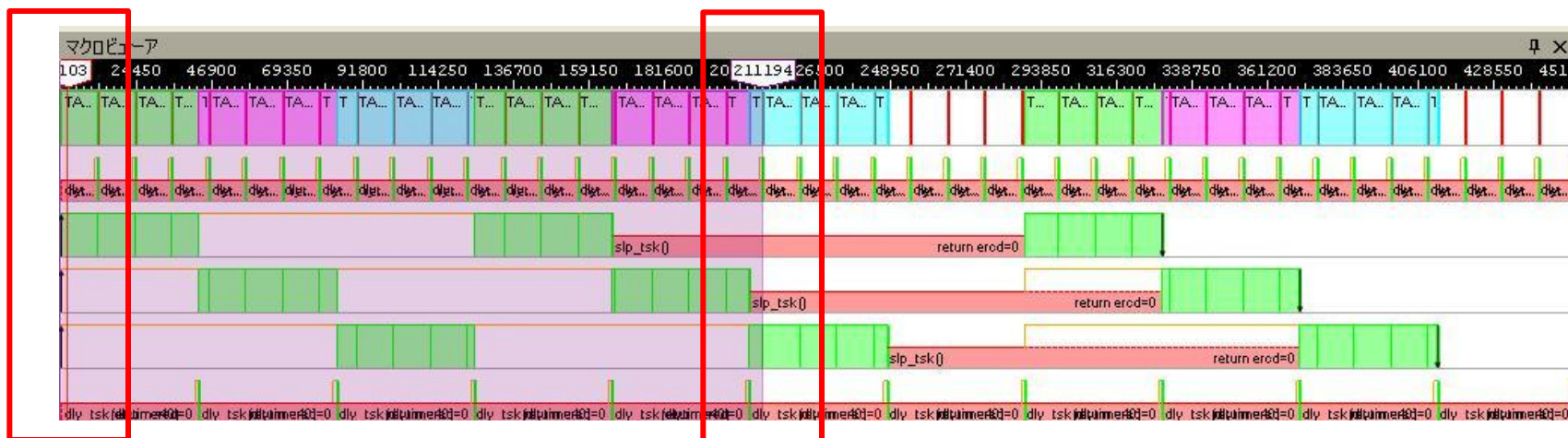
# TLVの各情報表示画面

---

1. リソースエクスプローラ : リソース情報画面表示
2. 可視化ルールエクスプローラ : 可視化情報画面表示
3. トレースログビューアー : トレースログ画面表示
4. マクロビューア: マクロ画面表示

# マクロビューアー

- ・全マクロ画面表示
- ・マウスでマーカを設定することにより、その部分の詳細情報が次のページに出てくる、可視化詳細表示部に表示される。





# マクロビューアー—可視化詳細表示部



拡大



- ・ 区間マクロ情報画面表示
- ・ 前のマクロビューアーで、マウスでマーカを設定することにより、その部分の詳細情報が拡大されて表示される。



# マクロビューアー—可視化詳細表示部



- ・区間マクロ情報画面表示

- ・リソースのタブを開くことによって、詳細項目ごとの振舞いを見ることができる。

- ・Ctrl+ホイール(矢印キー): 可視化表示領域移動
- ・Shift+ホイール: 拡大縮小
- ・左ダブルクリック: 移動

# マクロビューアー



# その他の機能

---

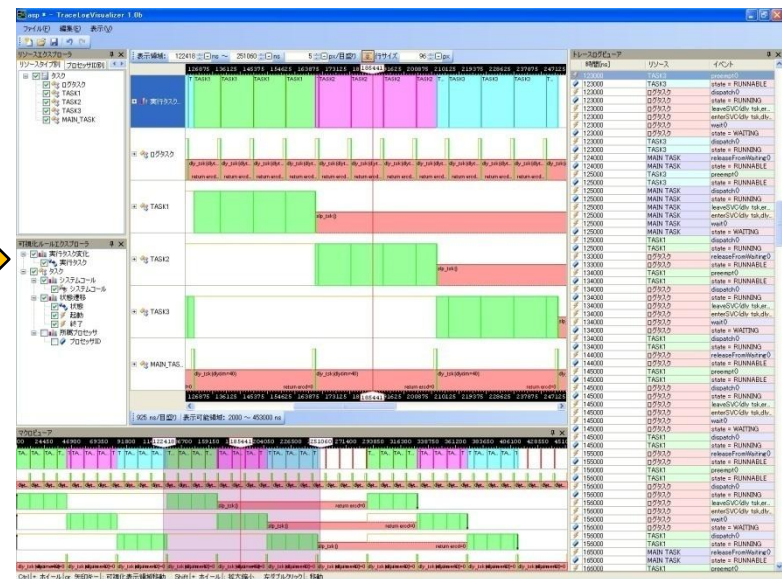
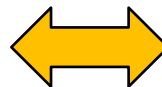
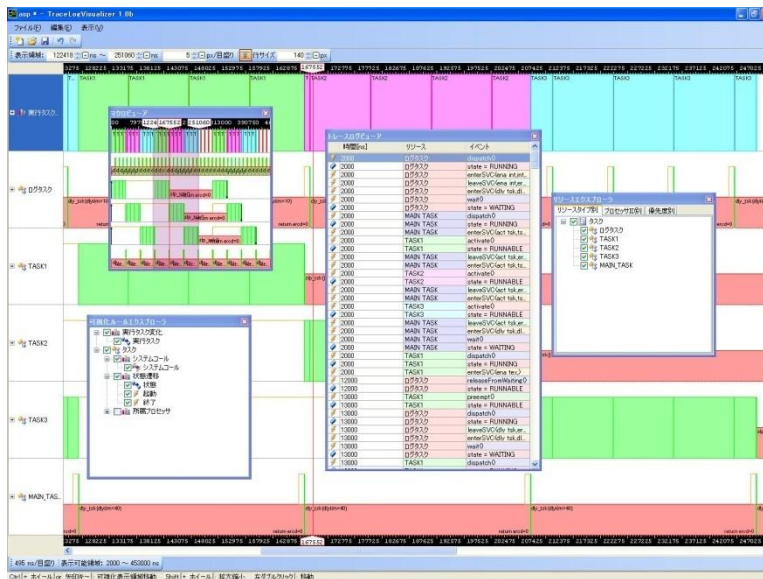
- ・ 各画面位置移動、表示、非表示
- ・ リロード
- ・ スクリーンショット
- ・ ツールバー
- ・ 簡易検索

# その他一各画面位置移動、表示、非表示

- ・可視化情報、リソース情報、トレースログ画面、マクロビューアのみ可能.
- ・表示は「メニュー」の「表示」でクリック.
- ・位置移動は各画面をドラッグ&ドロップで移動.
- ・windowbarをダブルクリックすると、元の位置に戻る

位置を移動した画面

元の位置に移動した画面



# その他 ― リロード

## リロード：

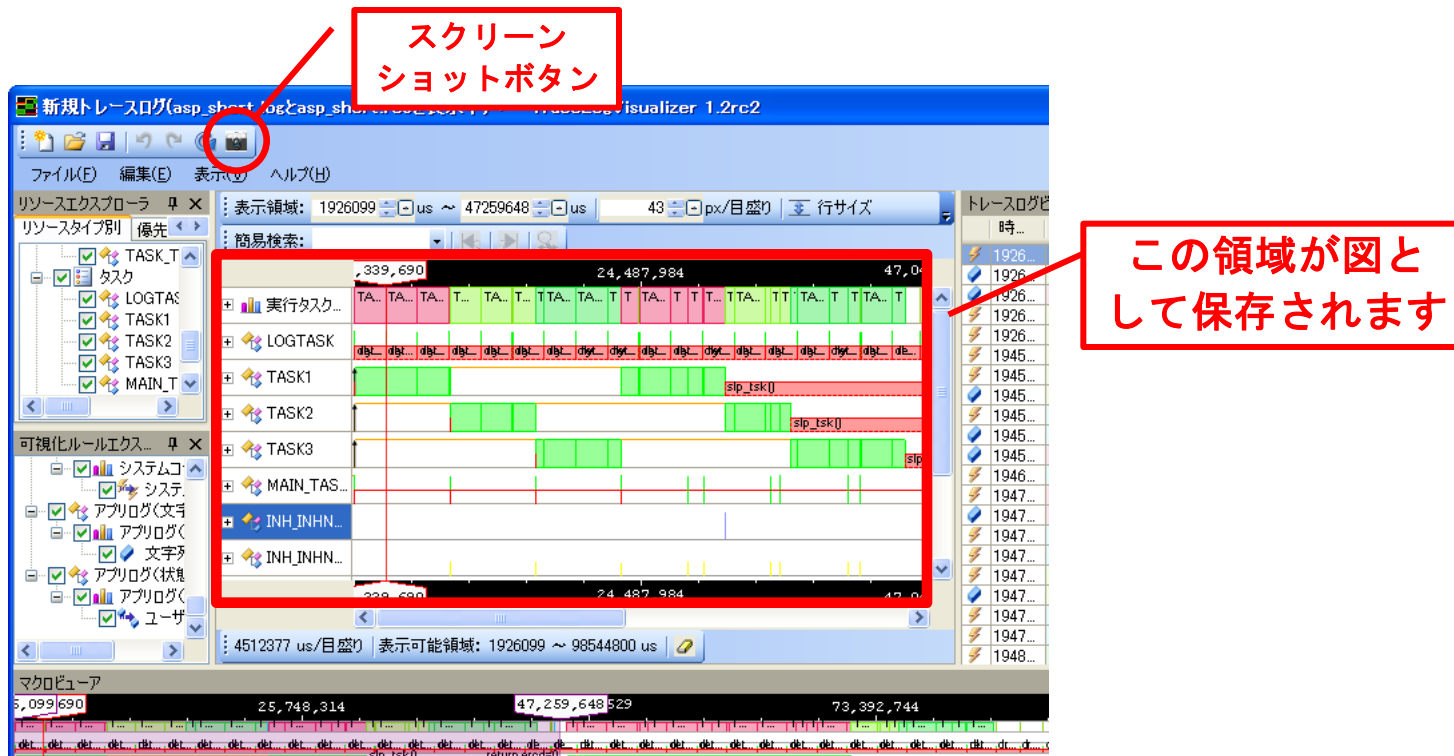
一度読込んだ .res ファイルと .log ファイルを再度読み込みます。両ファイルをドラッグ・ドロップすることでもリロードが可能です



# その他 ― スクリーンショット

## ・スクリーンショット

一部表示区間の情報を図として保存することができます。対応形式は bmp, jpg, png, gif の4種類です。





# その他ツールバー

表示領域: 145000 ns ~ 195285 ns 5 px/目盛り 行サイズ 108 px

## ※ ツールバーの左から順に

### ・ ns/目盛の拡大、縮小

- ・ +、- ボタンで ns/目盛を拡大、縮小可能

- ・ 現在の ns/目盛表示部分をクリックするとトラックバー(スライダー)で変更可能

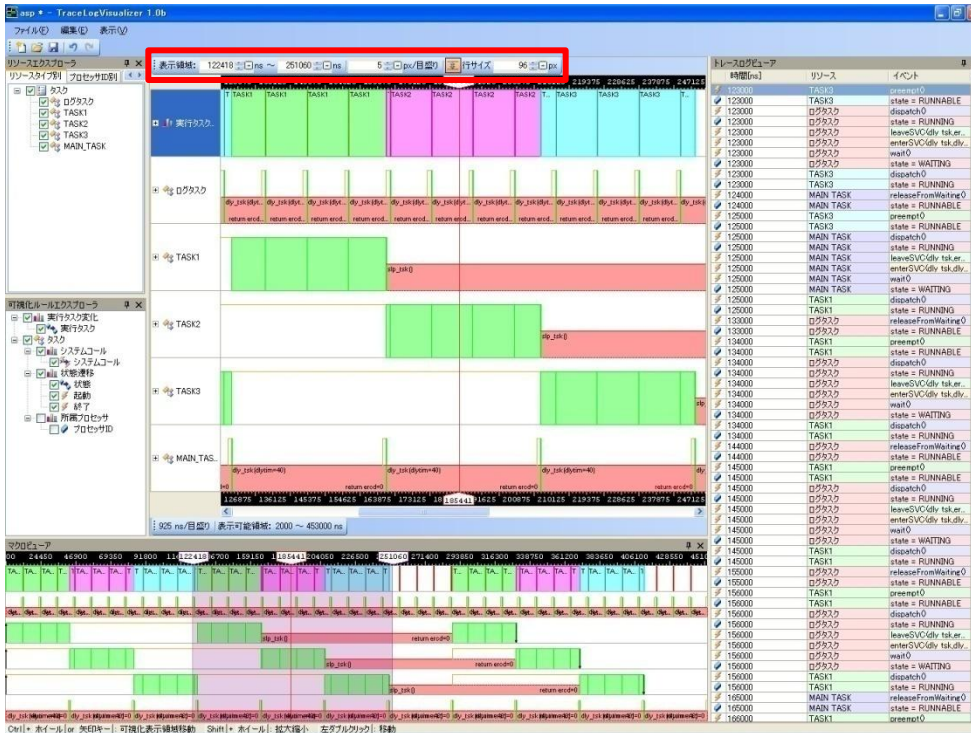
### ・ pixel/目盛の拡大、縮小

- ・ +、- ボタンで pixel/目盛を拡大、縮小可能

- ・ 現在の pixel/目盛表示部分をクリックするとトラックバー(スライダー)で変更可能

### ・ 行サイズ固定

- ・ タスク状態表示部をウィンドウズに合わせて行サイズ固定



# その他 - イベント検索機能：簡易検索

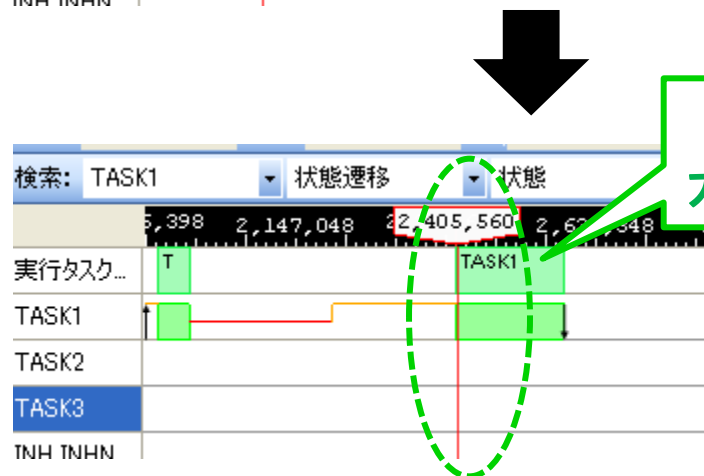
## ・簡易検索とは

リソース、ルール、イベントといった検索条件を指定し、  
条件に合致した時刻を探す機能



1. 検索条件を指定  
(ドロップダウンリストから選択)

2. 検索ボタン  
を押す



3. 該当時刻へ  
カーソルが移動

4. 該当時刻のログ  
に色がつく

The screenshot shows a trace log viewer with a table of events. The table has three columns: '時間', 'タスク', and 'イベント'. The events are listed in a table with alternating blue and white rows. A blue dashed circle highlights the first three rows of the table.

時間	タスク	イベント
2405560	TASK1	dispatch()
2405560	TASK1	state = ...
2405560	Current...	name = ...
2579398	TASK1	exit()
2579398	TASK1	state = ...
3179398	TASK1	activate()
3179398	TASK1	state = ...
3305560	TASK1	dispatch()
3305560	TASK1	state = ...



# その他 – 簡易検索 ～検索条件について～

## •簡易検索では以下の4つの検索条件を指定できる

1. リソース名
2. ルール名
3. イベント名
4. イベント詳細

※1 は リソースファイルで定義されている要素

2,3,4 は可視化ルールファイルで定義されている要素である

```
“taskStateChange”:{ // ルール名
  "DisplayName":"状態遷移",
  "Target":"Task",
  "Shapes":{
    “stateChangeEvent”:{ //イベント名
      "DisplayName":"状態",
      "From":"${TARGET}.state",
      "To":"${TARGET}.state",
      "Figures":{
        “${FROM_VAL}==RUNNING”      :“runningShapes”, //イベント詳細
        “${FROM_VAL}==RUNNABLE”     :“runnableShapes”,
      }
    }
  }
}
```

※ “TLV\_1.2/visualizeRules/toppers\_rules.vis” の一部を抜粋

# その他 – 簡易検索 ～検索の種類について～

## •検索の種類

### 1. 後方検索

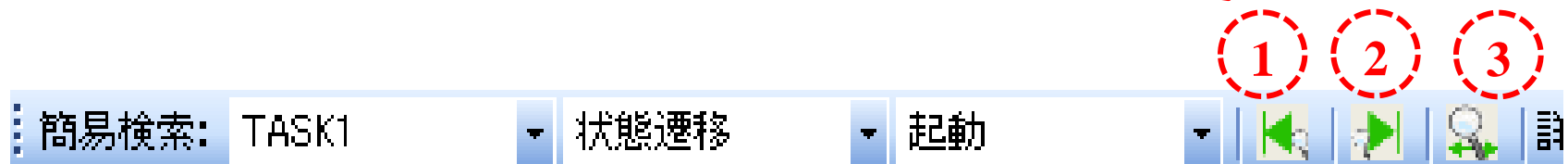
検索条件に合致する、現在から一番近い前の時刻へ移動する

### 2. 前方検索

検索条件に合致する、現在から一番近い次の時刻へ移動する

### 3. 全体検索

検索条件に合致する全ての時刻にマーカーを引く



# その他 - 簡易検索 ~全体検索について~

- 全体検索では、可視化表示部分の該当箇所にマーカーが引かれます



※マーカーは全消去ボタンで  
全て消すことができます

※全体検索ボタンを押した際、一部表示画面には  
すぐにマーカーが引かれます。  
一度マウスカーソルを一部表示画面、もしくは全  
体表示部分に移動させると全体表示部分にもマ  
ーカーが反映されます

# その他 – イベント検索機能：詳細検索

## ➤ 詳細検索

簡易検索よりも指定できる条件を増やし、より複雑な条件で目的のイベントを発見できるようにした検索方法

## ➤ 詳細検索で可能になること

ある**イベント**を、他の**イベントの発生**とその**発生タイミング**を用いて絞り込むことができる

例1. TASK1 が**実行状態へ遷移したイベント**のうち以下を満たすもの

- そのイベント発生時刻の**5秒以後**に、TASK1が**実行可能状態に遷移するイベント**が発生

例2. TASK2が**起動したイベント**のうち以下を満たすもの

- そのイベント発生時刻の**3秒以上前**に、TASK1が**起動するイベント**が発生

簡易検索:



詳細検索:



詳細検索フォーム呼び出し

# 詳細検索フォーム

## ※初期状態

検索開始点の移動

基本条件を追加

後ろを検索 次を検索 全体検索 マーカーを消す

タイムライン上のカーソル位置を移動させる

基本条件の設定パネルが追加される

基本条件1

絞り込み条件の追加

絞り込み条件の設定パネルが追加される

絞り込み条件1

後ろを検索 次を検索 全体検索

# 検索の実行

**基本条件1** [削除] [後ろを検索] [次を検索] [全体検索]

TASK1 [状態遷移] [状態] RUNNING

絞込み条件の追加 ☒ 全ての条件に一致 ☐ いずれかの条件に一致

絞込み条件1 ☐ 条件を否定 [削除]

TASK1 [状態遷移] [状態] RUNNING

基本条件のイベント発生時刻に対し 10000 us 以内に発生(基準時以前)

絞込み条件2 ☐ 条件を否定 [削除]

TASK2 [状態遷移] [状態] RUNNING

基本条件のイベント発生時刻に対し 500000 us 以内に発生(基準時以後)

**基本条件2** [削除] [後ろを検索] [次を検索] [全体検索]

TASK2 [状態遷移] [状態] RUNNABLE

絞込み条件の追加 ☒ 全ての条件に一致 ☐ いずれかの条件に一致

絞込み条件1 ☐ 条件を否定 [削除]

TASK1 [状態遷移] [状態] 起動

基本条件のイベント発生時刻に対し 10000 us 以上前に発生

絞込み条件2 ☐ 条件を否定 [削除]

TASK3 [状態遷移] [状態] RUNNING

基本条件のイベント発生時刻に対し 9999 us 以内に発生(基準時以前)

詳細検索フォーム

検索基準時の移動

[基本条件を追加] [後ろを検索] [次を検索] [全体検索] [マーカーを消す]

## 詳細検索フォームで検索を実行：

前方検索、後方検索の場合は①か②にマッチするイベント発生時刻のうち、検索開始点により近い方が検索される（①と②はORの関係となる）。

全体検索の場合は①にマッチするイベント発生時刻と②に該当するイベント発生時刻がすべて検索され、タイムライン上にマーカーが置かれる。

## 条件設定パネル上で検索を実行：

そのパネル上の検索条件を用いて検索がなされる。「全ての絞込み条件を満たす」、もしくは「絞込み条件のどれか1つを満たす」基本条件のイベントを検索できる。

# 新しい検索条件：イベント発生タイミング

## ➤ イベント発生タイミング：

絞込み条件のイベントが、基本条件のイベント発生時刻（基準時と呼ぶ）に対して、どのようなタイミングで発生したかを指定する条件であり、以下の4項目から選択可能

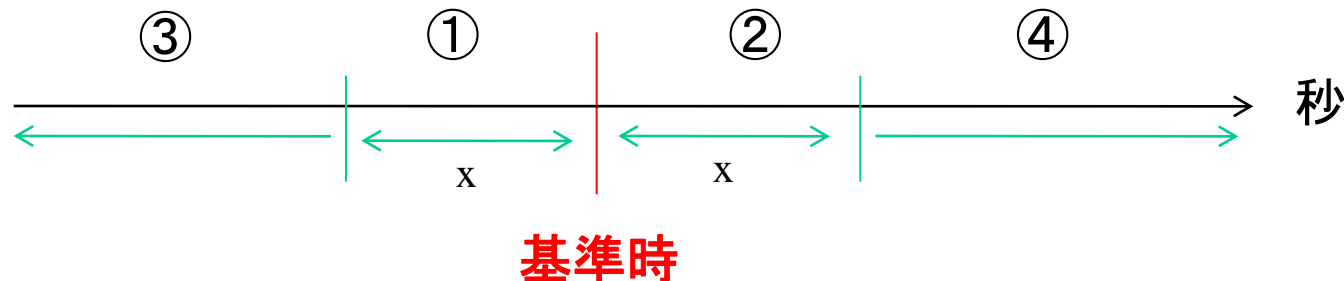
① x 秒以内（基準時以前）

② x 秒以内（基準時以後）

③ x 秒以上前

④ x 秒以上後

※時間単位はリソースファイルで指定した値となる

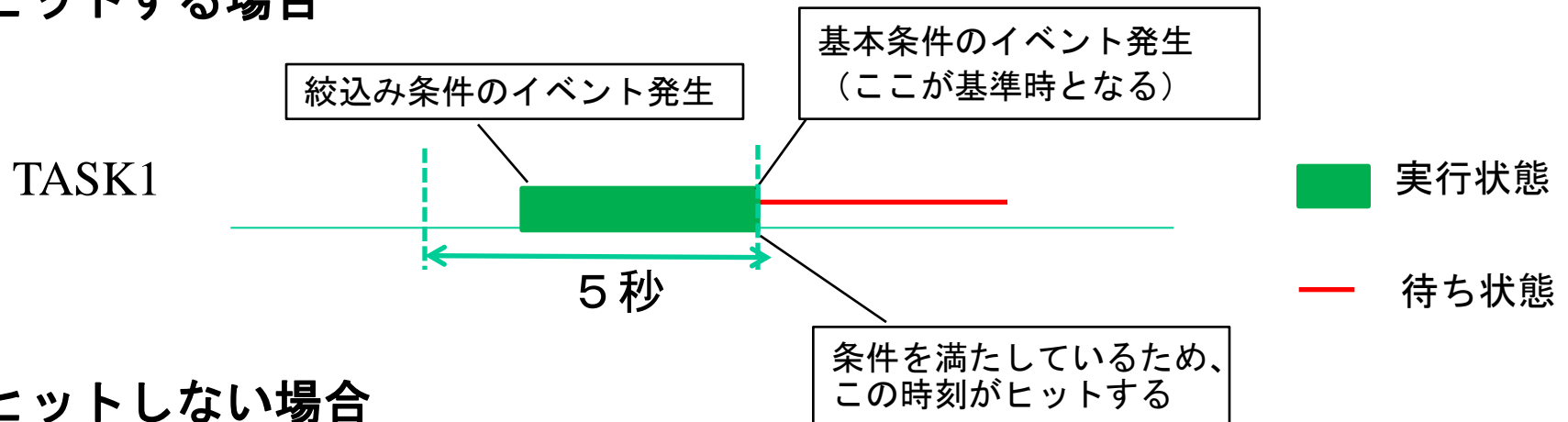


# 検索例： x 秒以内（基準時以前）

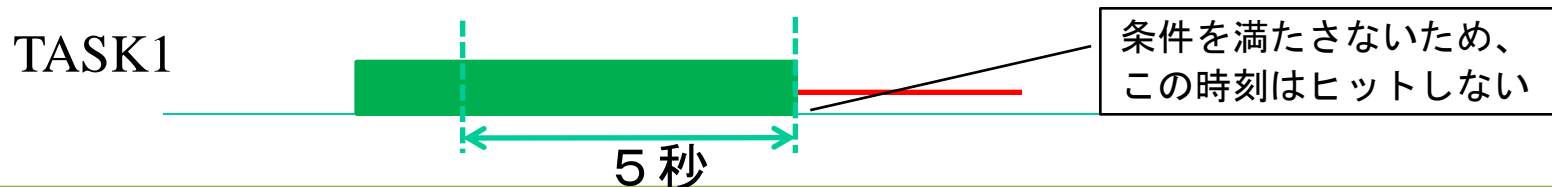
基本条件： TASK1が待ち状態になったイベント

絞り込み条件： 基準時に対して 5 秒以内にTASK1に  
実行状態への遷移イベントが発生

## ①ヒットする場合



## ②ヒットしない場合



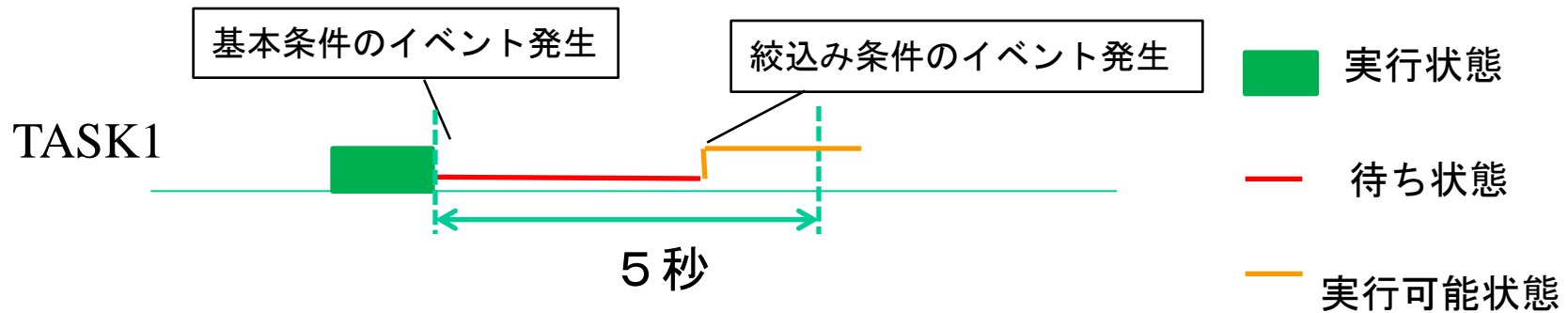


# 検索例： x 秒以内（基準時以後）

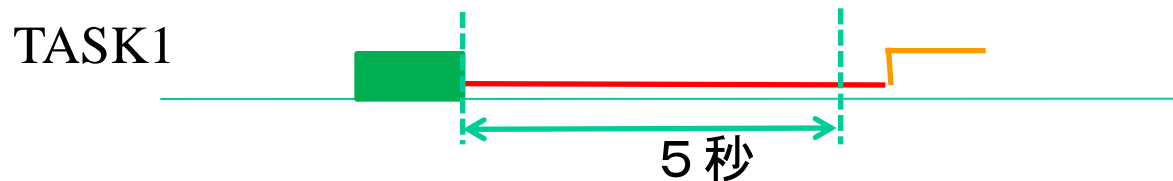
基本条件： TASK1が待ち状態になったイベント

絞込み条件： 基準時に対して 5 秒以内にTASK1に  
実行可能状態への遷移イベントが発生

## ①ヒットする場合



## ②ヒットしない場合

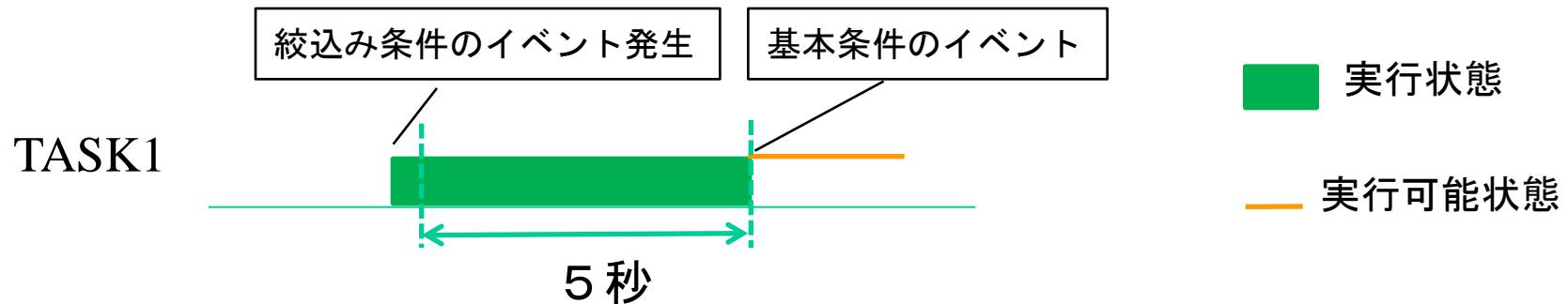


# 検索例： x 秒以上前

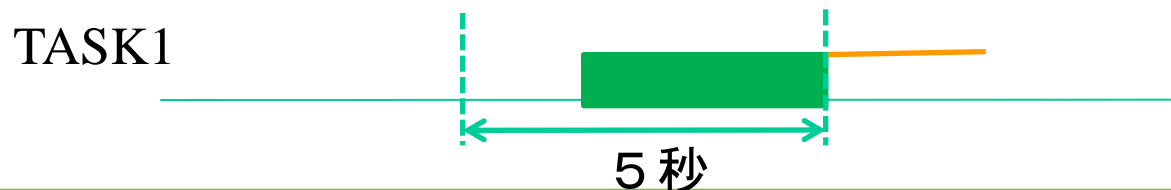
基本条件： TASK1が実行可能状態に遷移したイベント

絞り込み条件： 基準時に対して 5 秒以上前に TASK1に実行状態への遷移イベントが発生

## ①ヒットする場合



## ②ヒットしない場合

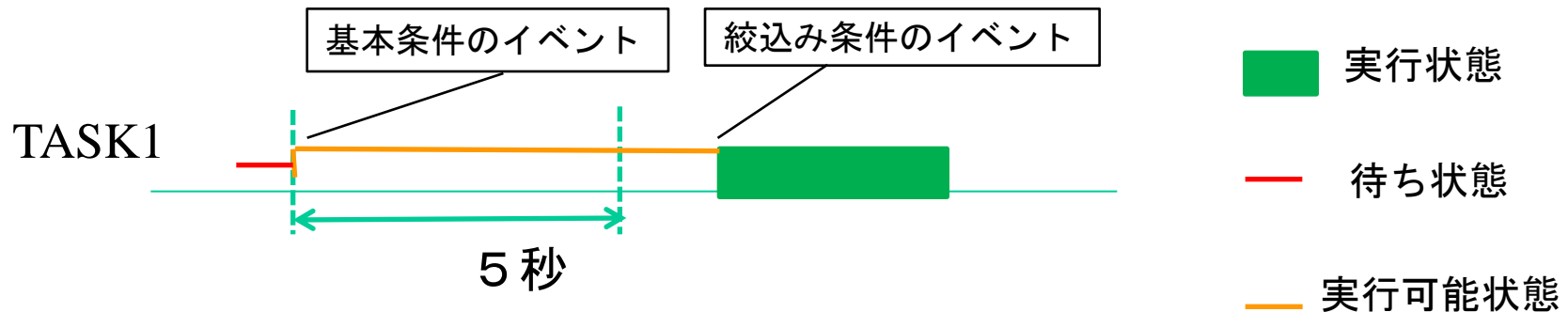


# 検索例： x 秒以上<sup>あと</sup>後

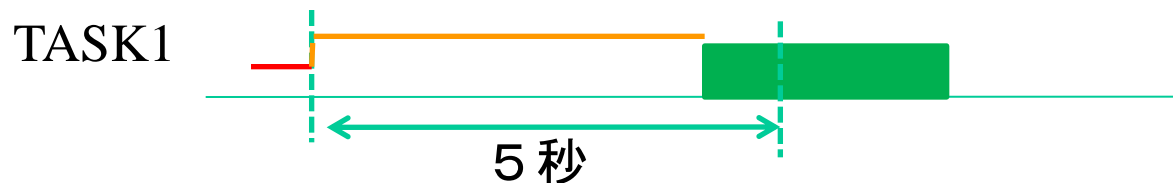
基本条件： TASK1が実行可能状態になったイベント

絞込み条件： 基準時に対して 5 秒以上後にTASK1に  
実行状態への遷移イベントが発生

## ①ヒットする場合



## ②ヒットしない場合



# タイミングを指定した検索の注意点

- 基本条件と絞込み条件で指定されたイベント以外のイベントは検索に考慮しない

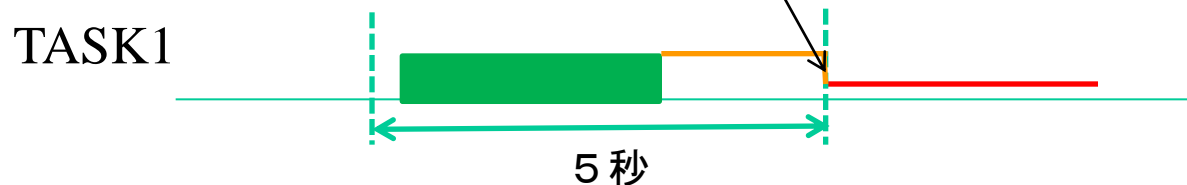
→ 弊害が発生する場合あり

## ➤ 弊害が起きる例

基本条件 : TASK1が待ち状態になったイベント

絞込み条件 : 基準時に対して5秒以内にTASK1に実行状態への遷移イベントが発生

以下の状況においても待ち状態に遷移した時刻がヒット



現状の検索では他のイベント（ここでは実行可能状態への遷移）を考慮していないため、実行状態の次が待ち状態となった場合のみを検索できない