最終更新日 ： 2012/03/23

TOPPERS/TLVマニュアル

（2.0 対応）

名古屋大学

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
/	README.txt	TLVの簡単な紹介
	TraceLogVisualizer.exe	TLVの本体
doc/	TLV.pdf	TLV本体のマニュアル
	TLV_convert_rules.pdf	変換ルールのマニュアル
	TLV_visualize_rules.pdf	可視化ルールのマニュアル
	statistics_viewer_manual.pdf	統計情報表示機能のマニュアル
	converter_manual.ppt	標準形式変換外部プロセス化機能マニュアル
convertRules/	asp.cnv	TOPPERS/ASPカーネルの変換ルール
	fmp.cnv	TOPPERS/FMPカーネルの変換ルール
	tecs.cnv	TECSの変換ルール
resourceHeaders/	asp.resh	TOPPERS/ASPカーネルのリソースファイル
	fmp.resh	TOPPERS/FMPカーネルのリソースファイル
	tecs.resh	TECSのリソースファイル
visualizeRules/	asp_rules.viz	TOPPERS/ASPカーネルの可視化ルール
	asp_shapes.viz	TOPPERS/ASPカーネルの図形定義
	fmp_rules.viz	TOPPERS/FMPカーネルの可視化ルール
	fmp_shapes.viz	TOPPERS/FMPカーネルの図形定義
	tecs.viz	TECSの可視化ルール
	toppers_rules.viz	TOPPERS共通の可視化ルール
	toppers_shapes.viz	TOPPERS共通の図形定義

フォルダ名		ファイル名	説明
sampleFiles/SampleLog/	asp/	asp_short.log	TOPPERS/ASPカーネルのトレースログのサンプル
		asp_short.res	
		asp_short.tlv	
		asp_long.log	
		asp_long.res	
		asp_long.tlv	
	fmp/	fmp_short.log	TOPPERS/FMPカーネルのトレースログのサンプル
		fmp_short.res	
		fmp_short.tlv	
		fmp_long.log	
		fmp_long.res	
		fmp_long.tlv	
	tecs/	tecs.log	TECSのトレースログのサンプル
		tecs.res	
		tecs.tlv	
logtrace/	asp/	kernel_fncode.h	TOPPERS/ASPカーネルのログトレースモジュール
		tlv.tf	
		trace_config.c	
		trace_config.h	
		trace_dump.c	
	fmp/	tlv.tf	TOPPERS/FMPカーネルのログトレースモジュール

実行環境

実行環境

TLV実行環境

・WindowsXP / Vista / 7

・Microsoft .NET Framework 4.0　をインストールすること.

ログ取得対象

・TOPPERS/ASPカーネル 1.7.0のトレースログ

・TOPPRES/FMPカーネル 1.2.0 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.

・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.

```
ENABLE_TRACE = true
```

TOPPERS/ASPカーネルのトレースログの取得

- asp/doc/user.txt の 11.6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方法の一例として，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_cyccb)/LOG_CYC_LEAVE(p_cyccb),
    LOG_ALM_ENTER(p_almcb)/LOG_ALM_LEAVE(p_almcb),
    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.

改行を入れる

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 11.6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方法の一例として，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_cyccb)/LOG_CYC_LEAVE(p_cyccb),
    LOG_ALM_ENTER(p_almcb)/LOG_ALM_LEAVE(p_almcb),
    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.

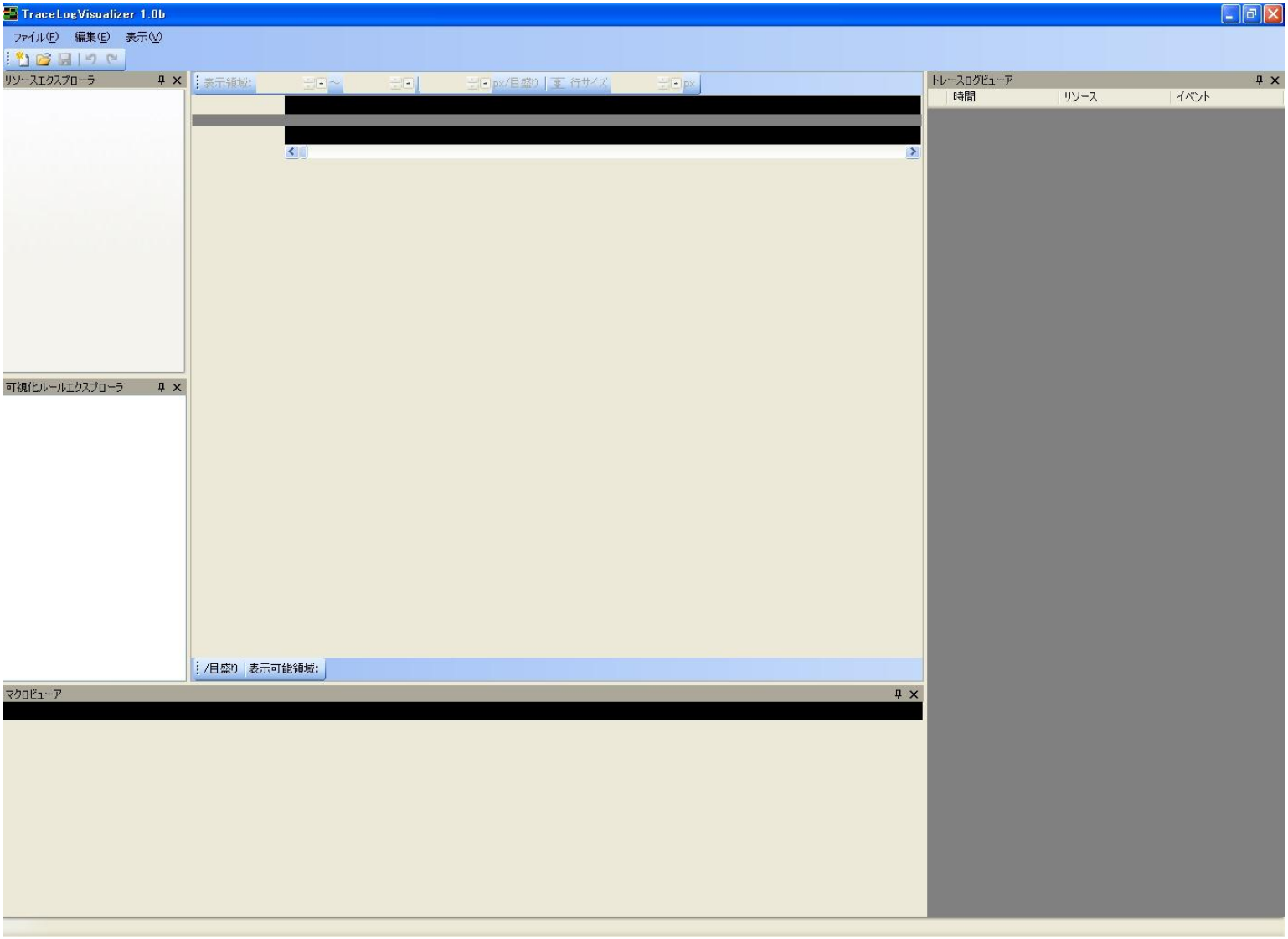
機能紹介

機能紹介

・sampleFile/SampleLog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実行

TLV初期画面



TLVメニュー一覧

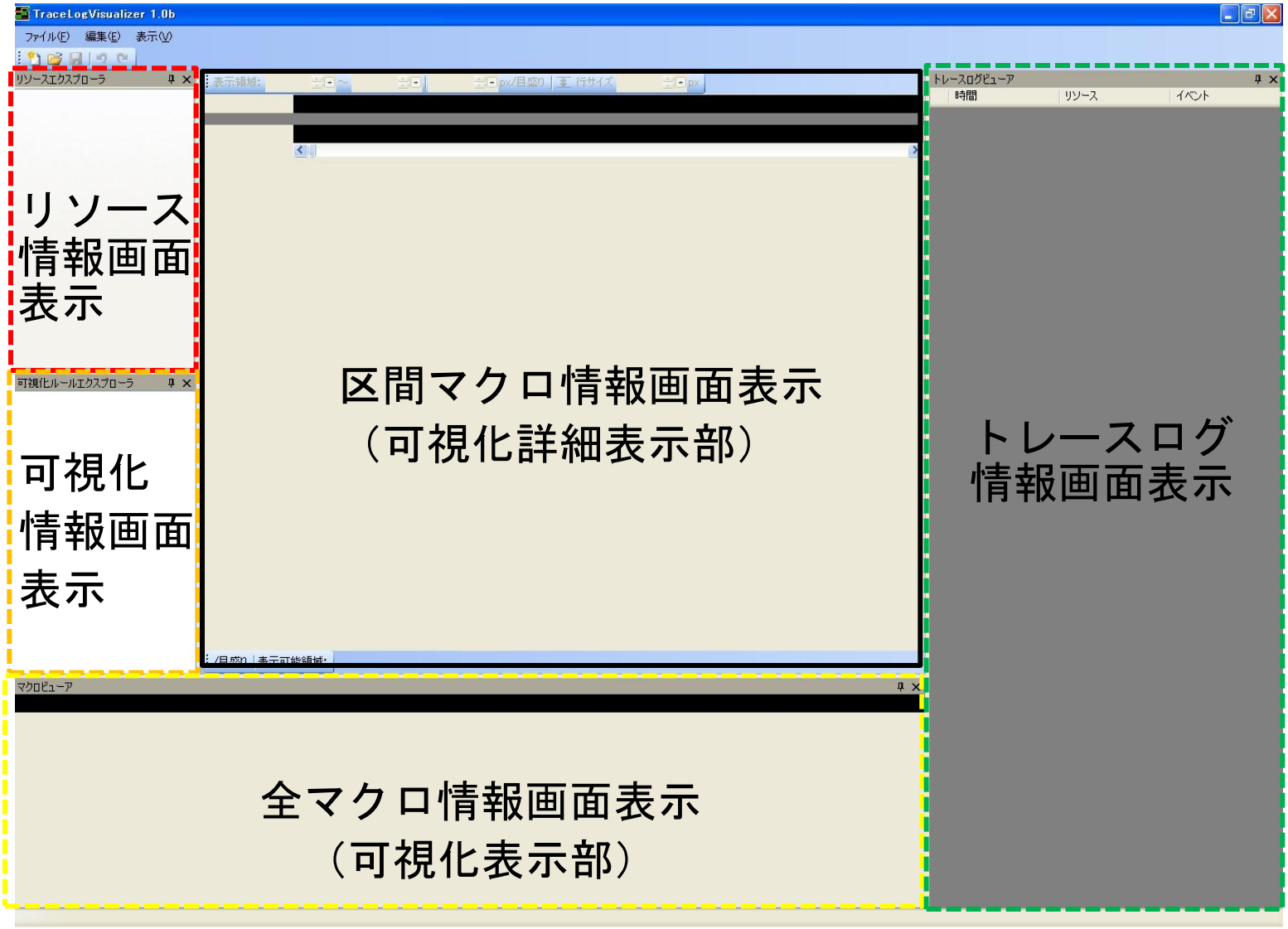
・ファイル
  ・新規作成－新しいログファイルをオープン
  ・開く－既存のログファイルをオープン(xxx.tlv)

・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ー可視化情報画面表示
  ・マクロビュアー(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

TLV実行

TLV初期画面



ログファイルオープン

・TLV2.0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

xxx.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

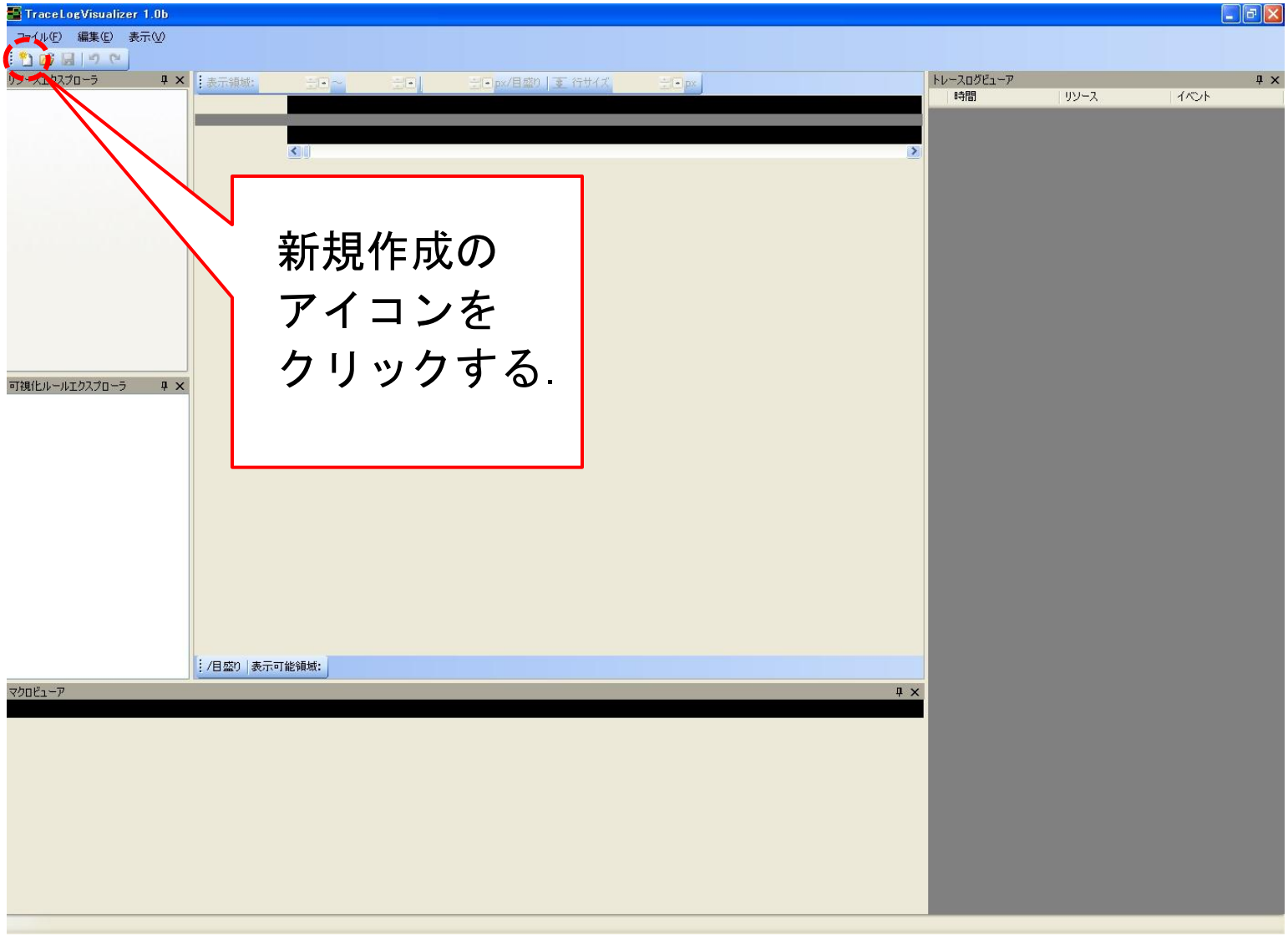
２．開く（保存したものを開く）

xxx.tlv　ファイルが必要．

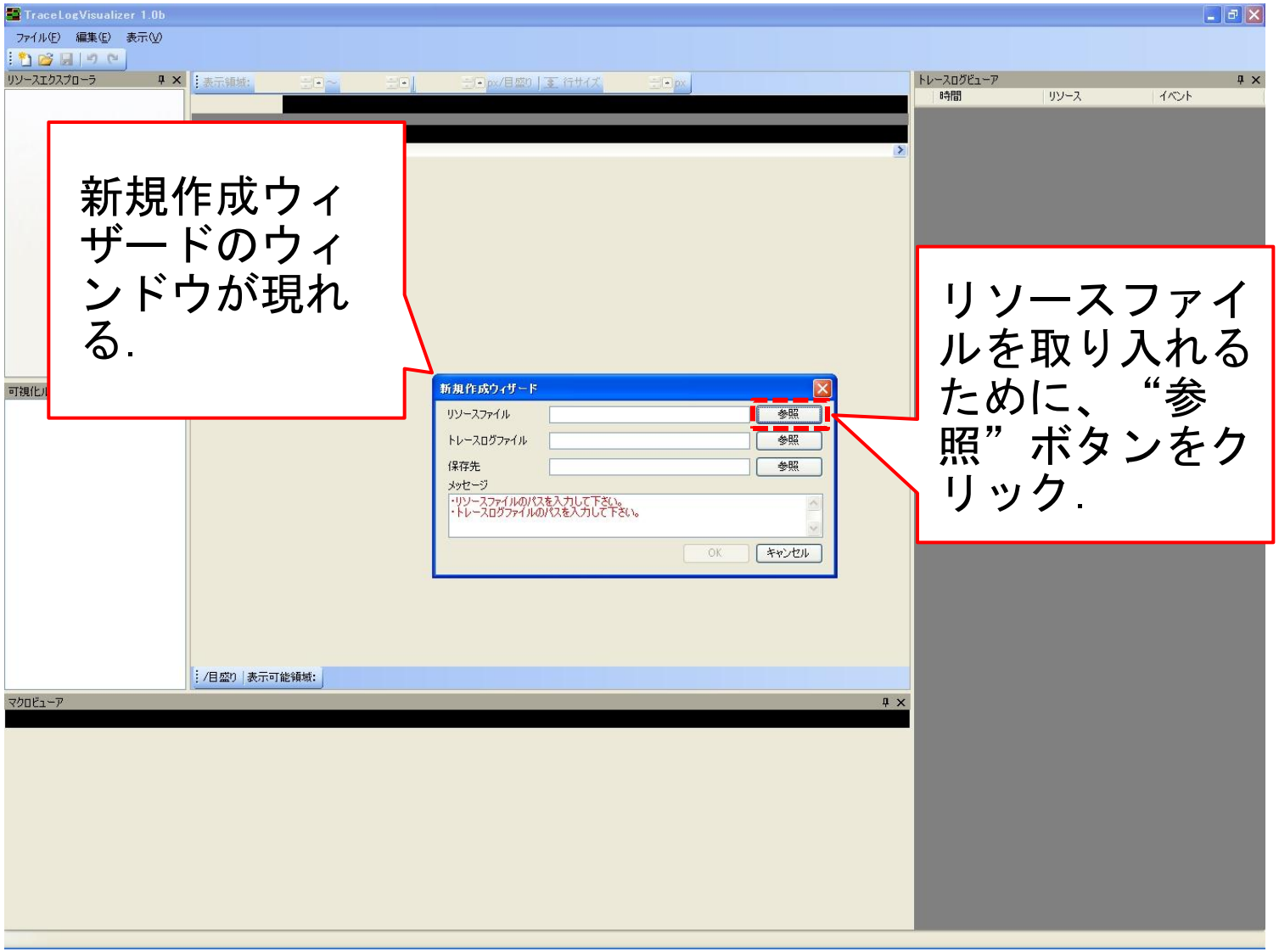
「メニュー」→「ファイル」→「開く」をクリック

1．新規作成ースタート

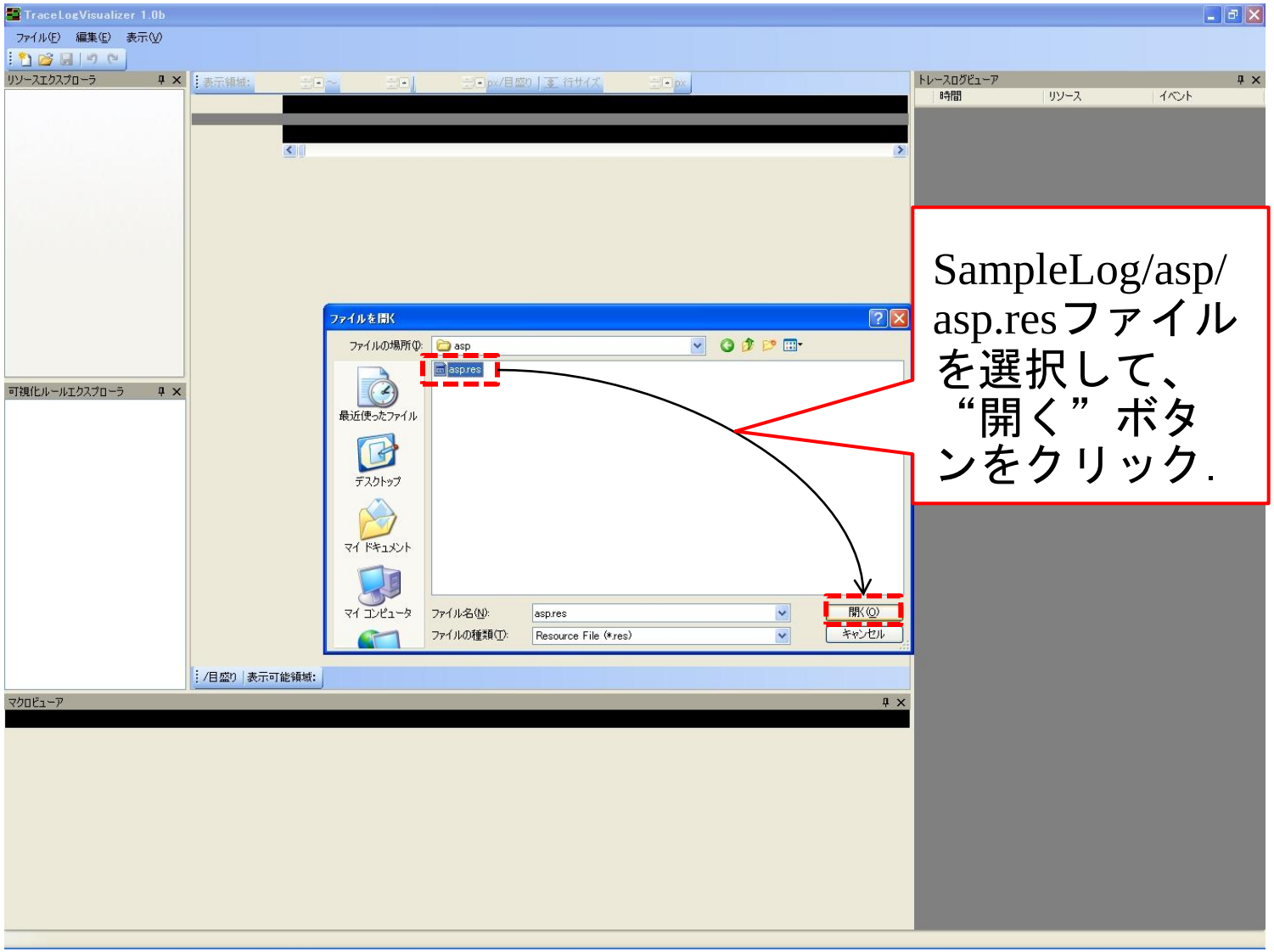
TLV初期画面



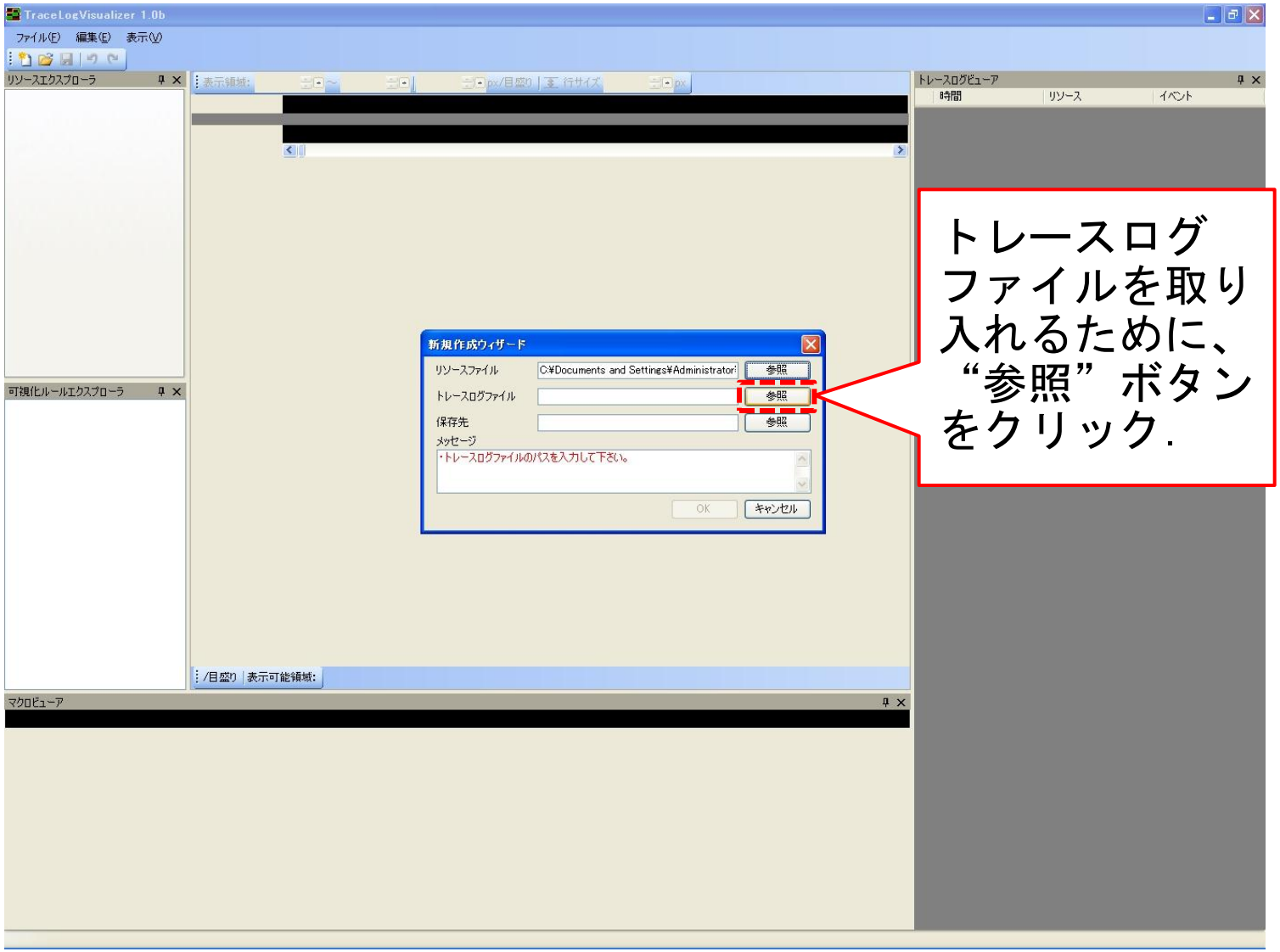
新規作成ーリソースファイルの参照



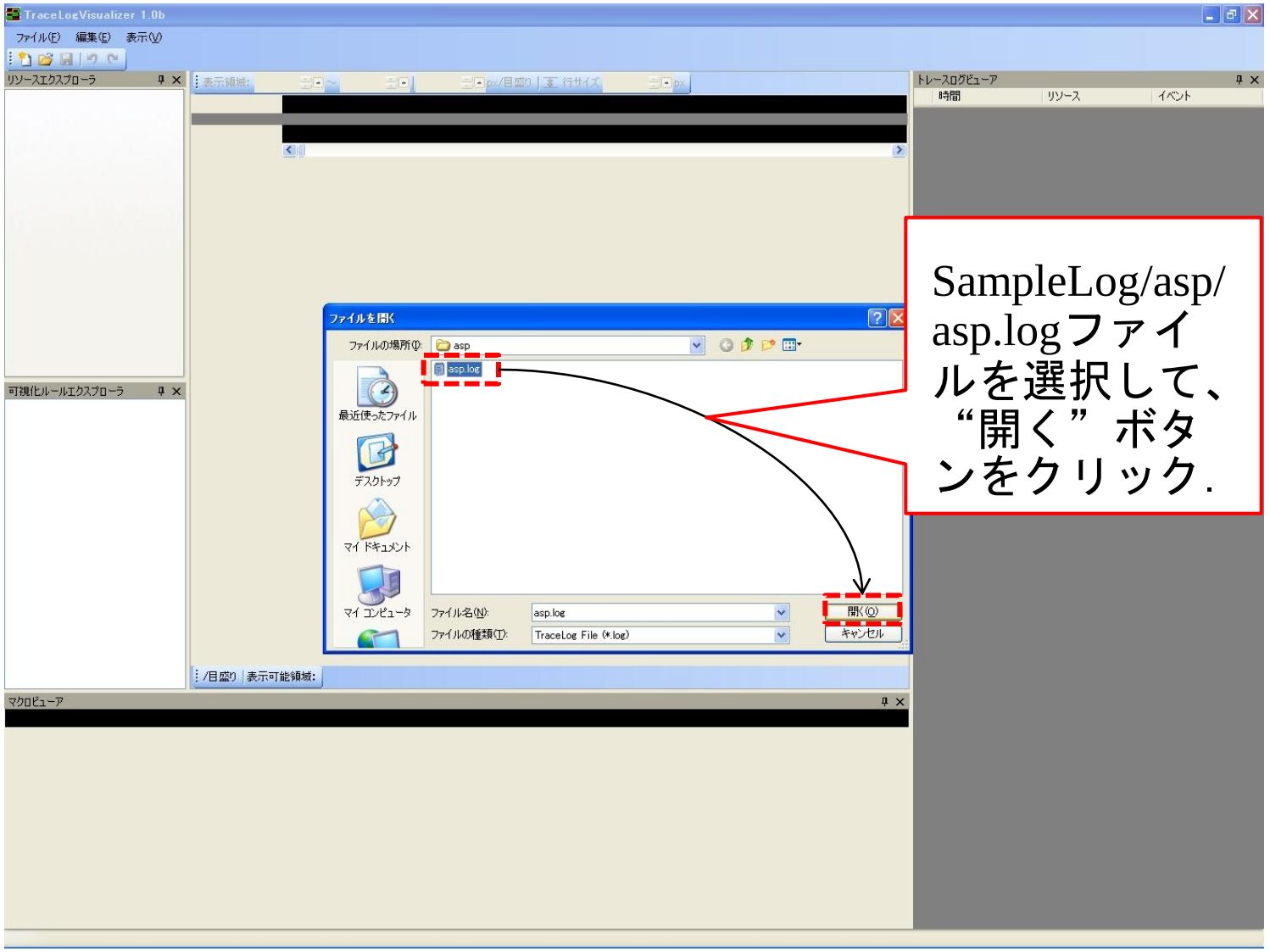
新規作成ーリソースファイルの参照



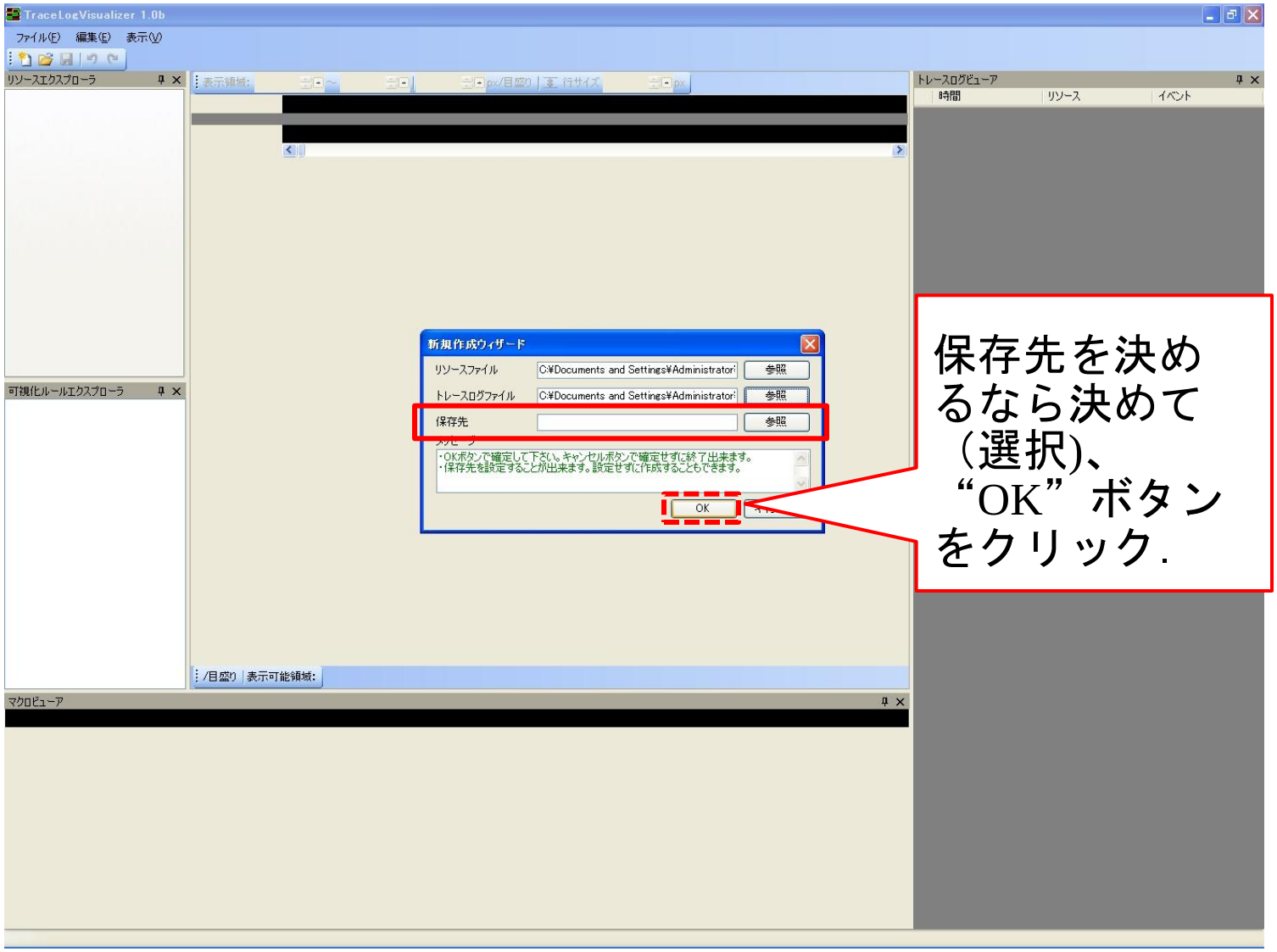
新規作成ー トレースログファイルの参照



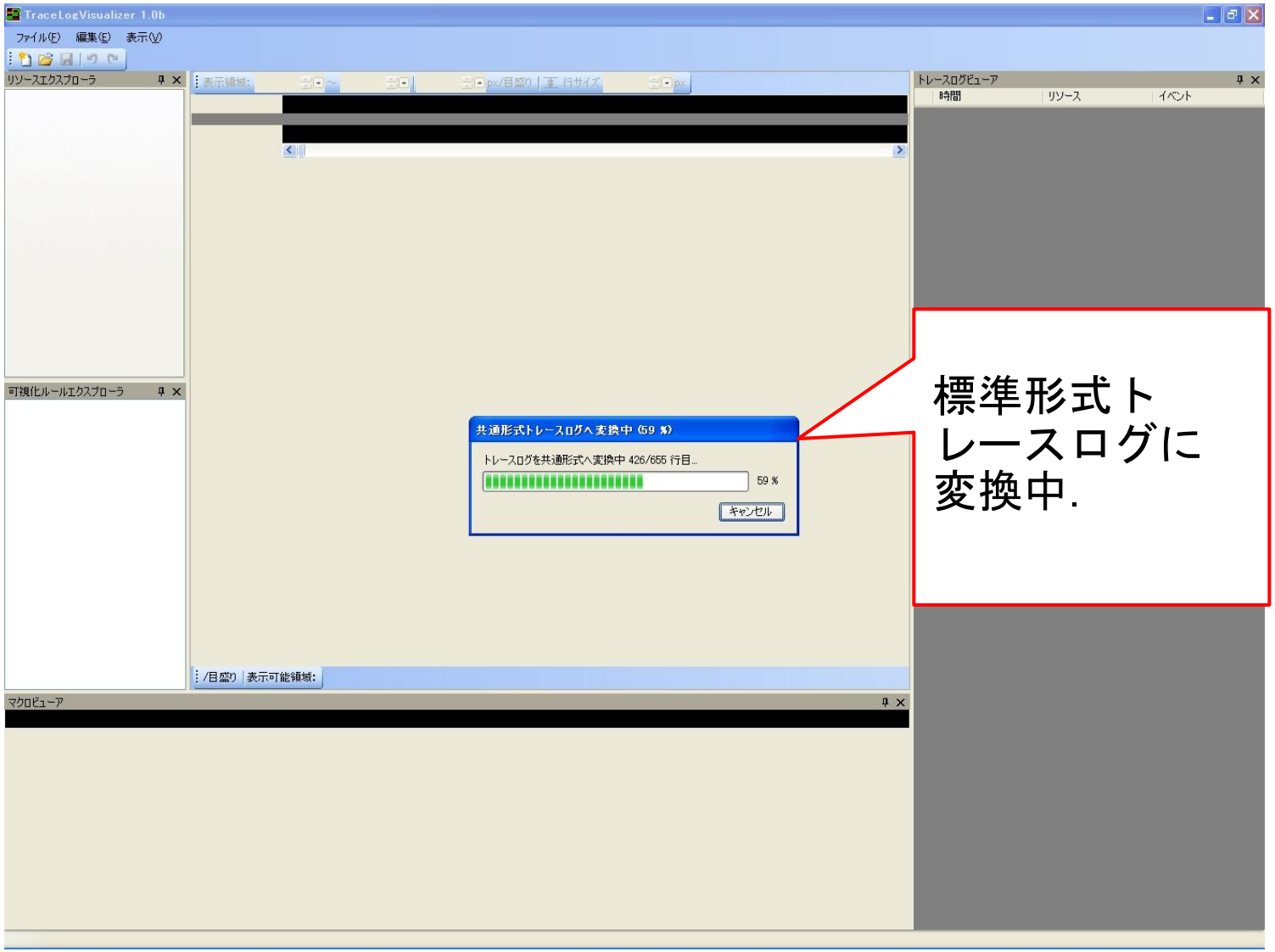
新規作成ー トレースログファイルの参照



新規作成ー保存先の設定



新規作成ー標準形式トレースログに変換



ログファイルオープン

・TLV2.0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

xxx.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）

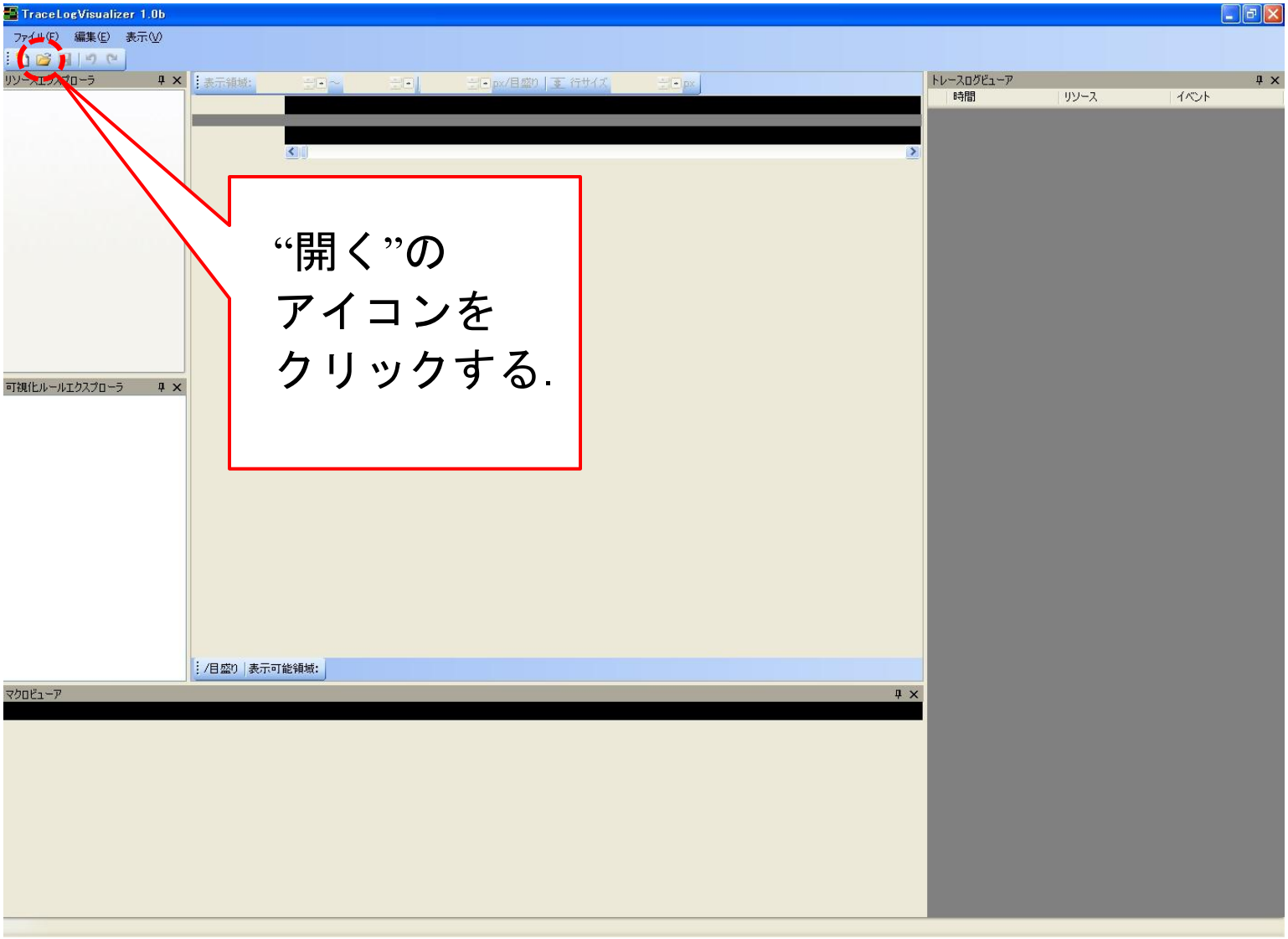
xxx.tlv　ファイルが必要.

「メニュー」→「ファイル」→「開く」をクリック

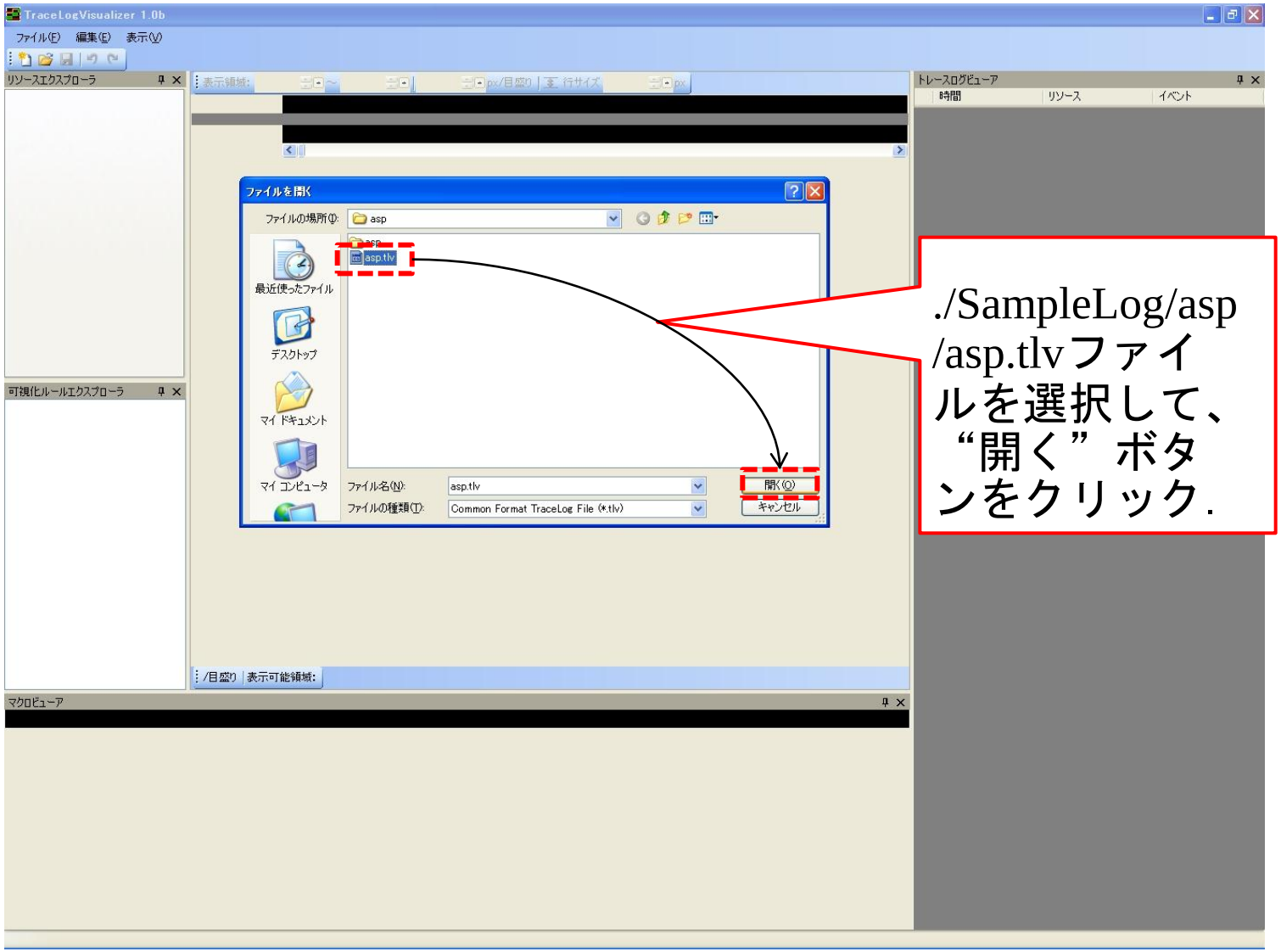
・今回は、 SampleFileフォルダに入っているファイルを例に説明する.

２．開く(保存したものを開く)

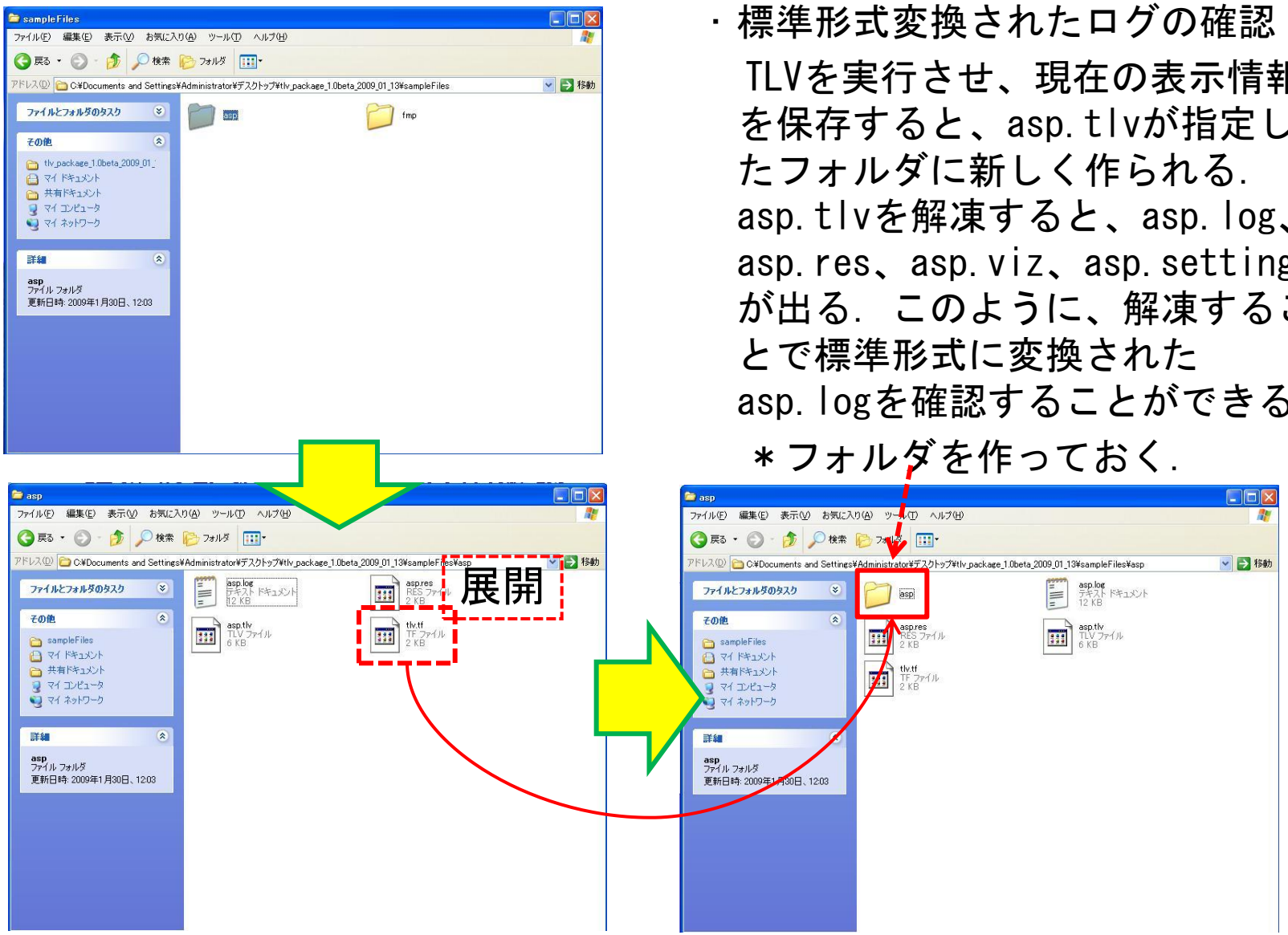
TLV初期画面



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

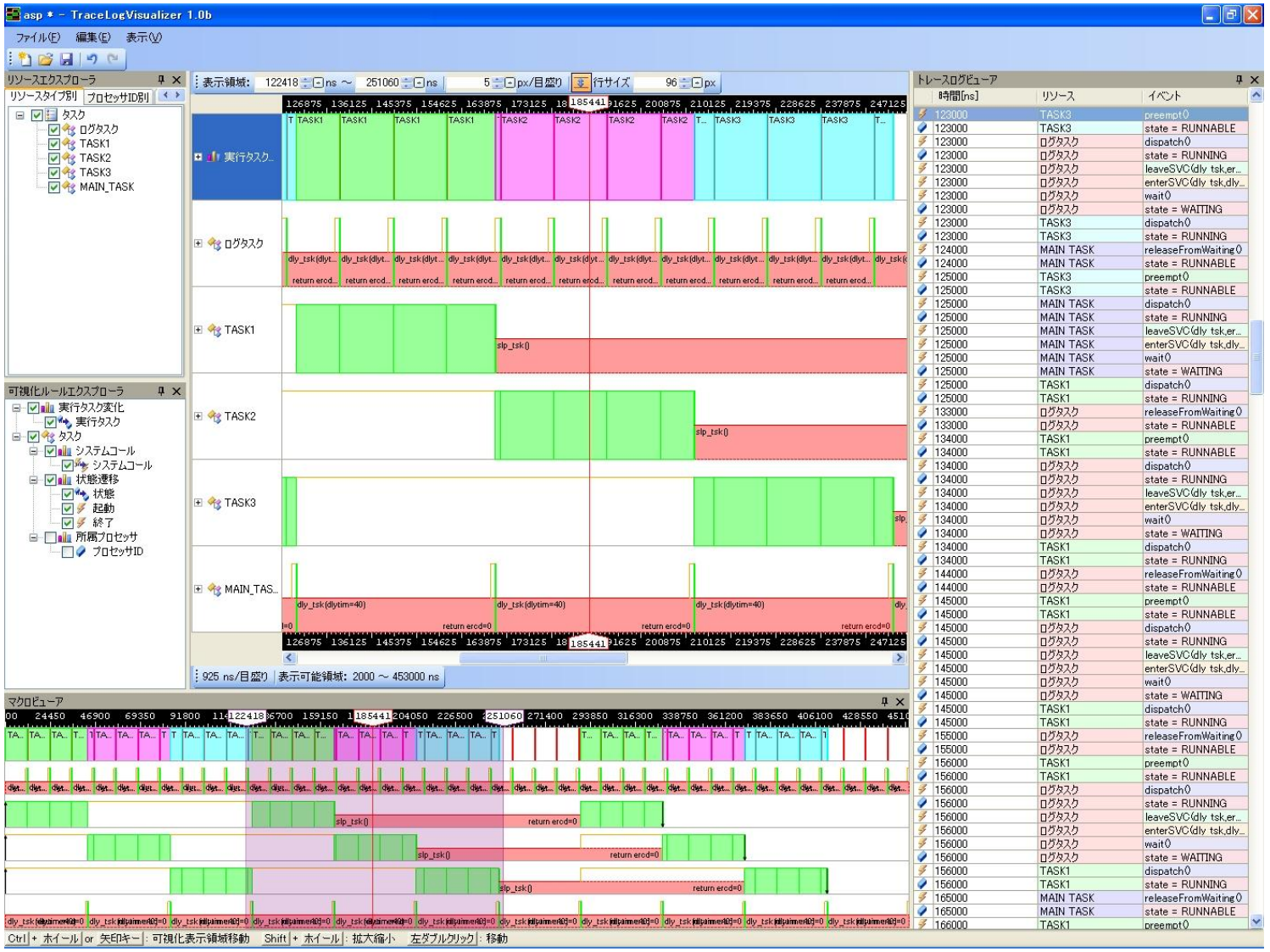


・標準形式変換されたログの確認

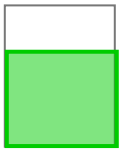
TLVを実行させ、現在の表示情報を保存すると、asp.tlvが指定したフォルダに新しく作られる.
asp.tlvを解凍すると、asp.log、asp.res、asp.viz、asp.settingが出る. このように、解凍することで標準形式に変換されたasp.logを確認することができる.

＊フォルダを作っておく.

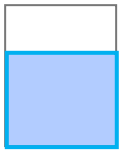
サンプルファイルを開いている画面



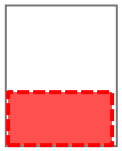
可視化表示部ー波形表示



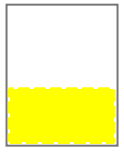
実行状態
(タスクコンテキスト)



非タスクコンテキスト
(slp_tsk,dly_tsk以外)

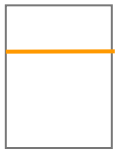


システムコール
(slp_tsk or dly_tsk)

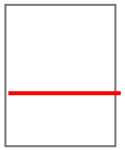


システムコール
(slp_tsk,dly_tsk以外)

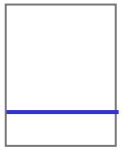
休止状態



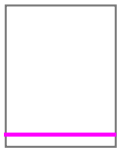
実行可能状態



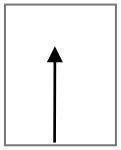
待ち状態



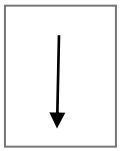
強制待ち状態



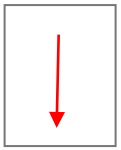
二重待ち状態



起動



終了



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

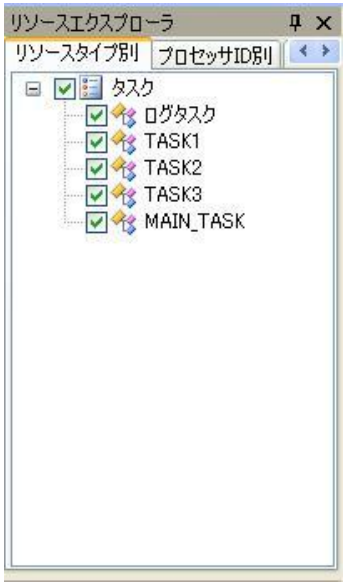
リソースエクスプローラ

・リソース情報画面表示

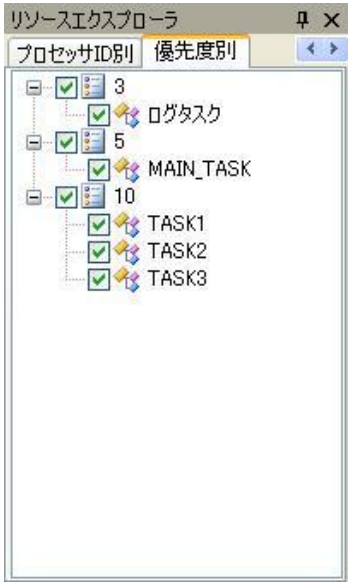
・タブをクリックして変えることにより、リソースタイプ別、優先度別にリソースを見ることができる.

・プロセッサIDはFMPの場合だけ表示

・リソースタイプ別



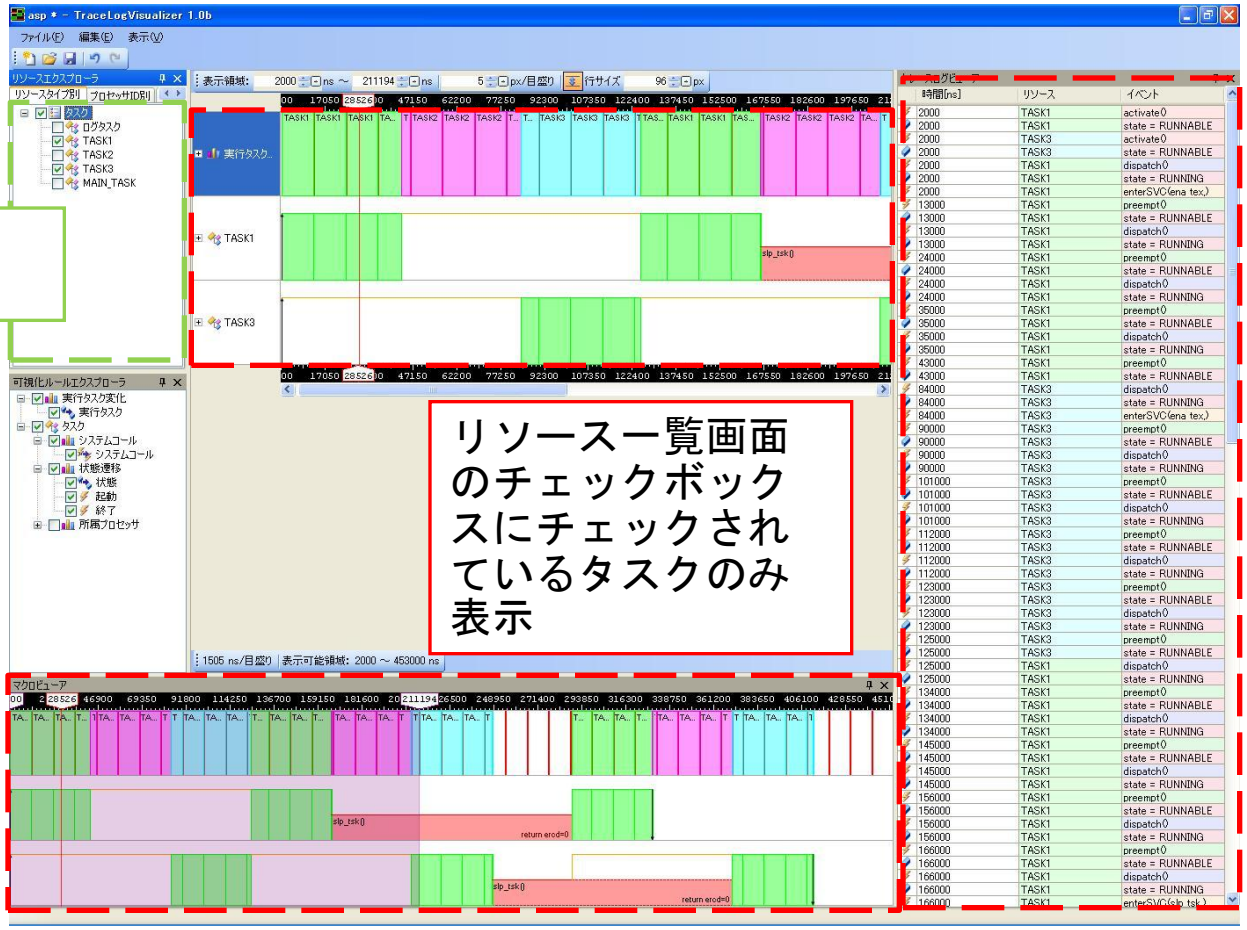
・優先度別



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



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

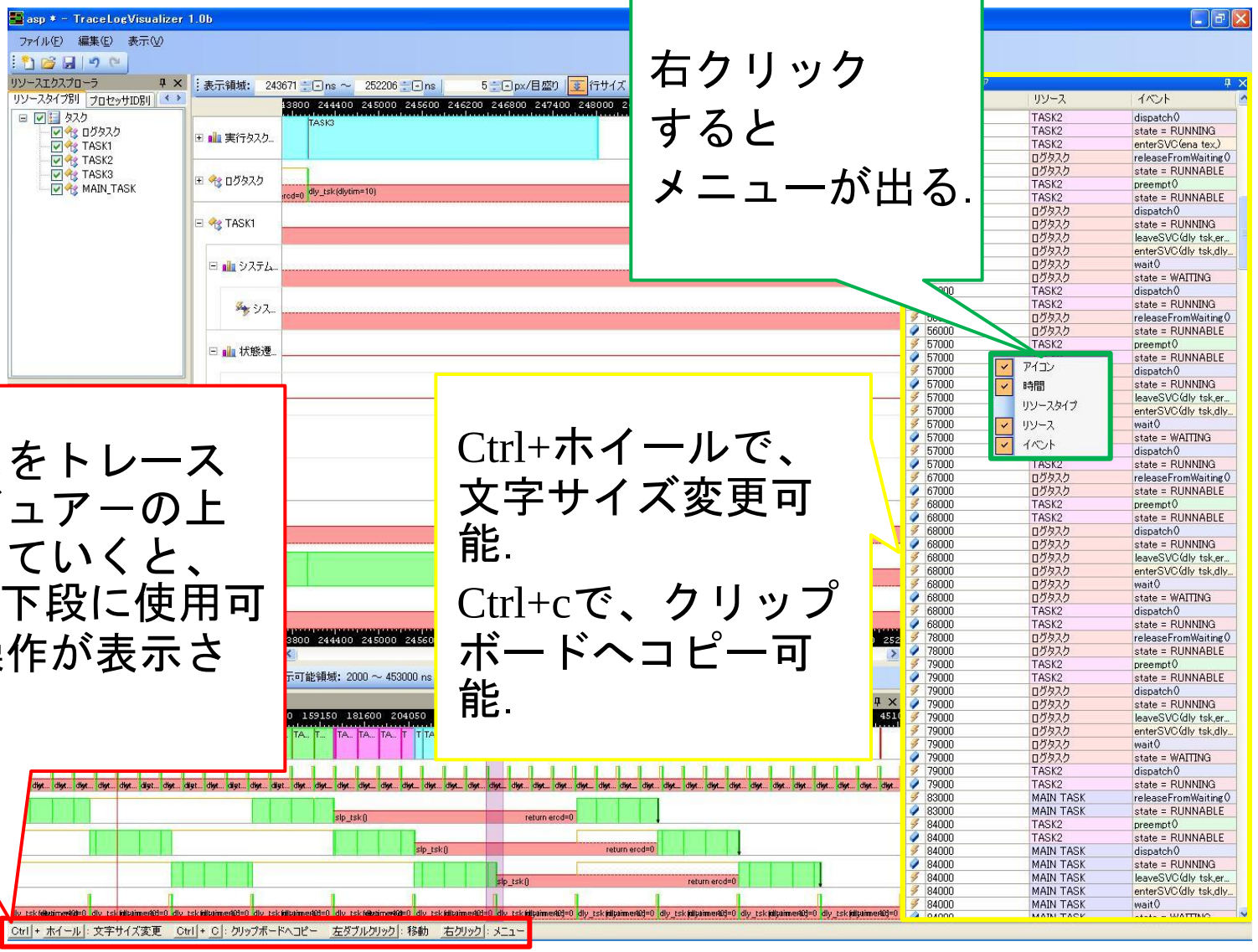
4. マクロビューア: マクロ画面表示

トレースログビューア

- 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


asp * - TraceLogVisualizer 1.0b
ファイル(F) 編集(E) 表示(V)
右クリック
すると
メニューが出る.
マウスをトレース
ログビュアーの上
に持っていくと、
TLVの下段に使用可
能な操作が表示さ
れる.
Ctrl+ホイールで、
文字サイズ変更可
能.
Ctrl+cで、クリップ
ボードへコピー可
能.
Ctrl + ホイール : 文字サイズ変更
Ctrl + C : クリップボードへコピー
左ダブルクリック : 移動
右クリック : メニュー

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

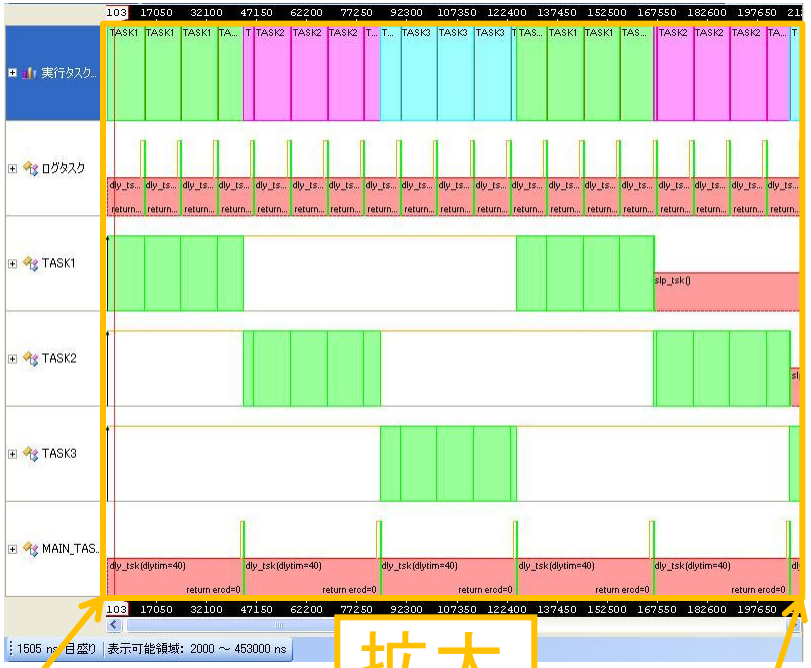
4. マクロビューア: マクロ画面表示

マクロビュアー

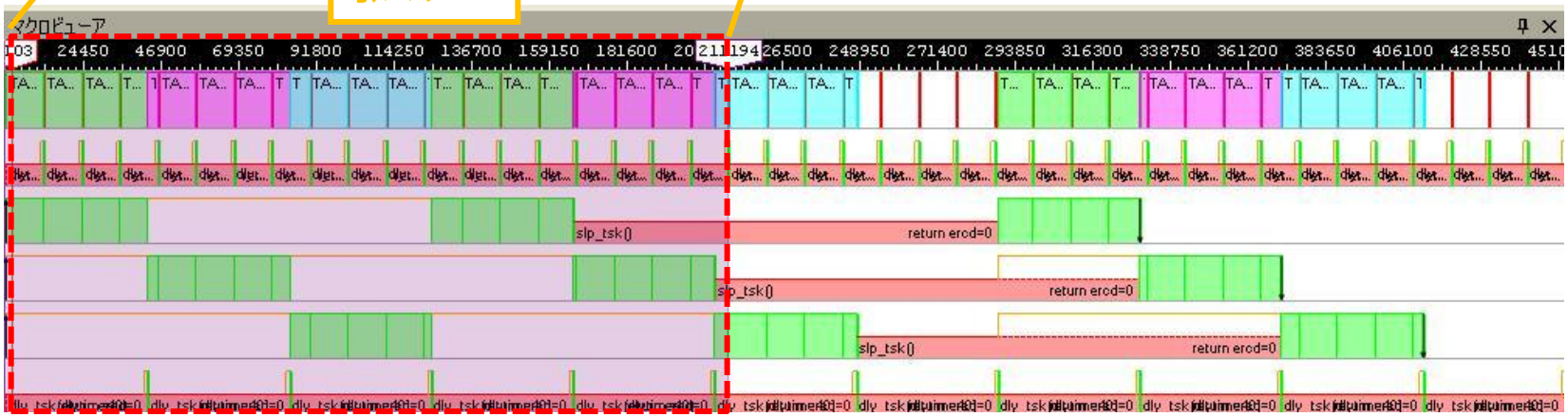
・全マクロ画面表示

・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

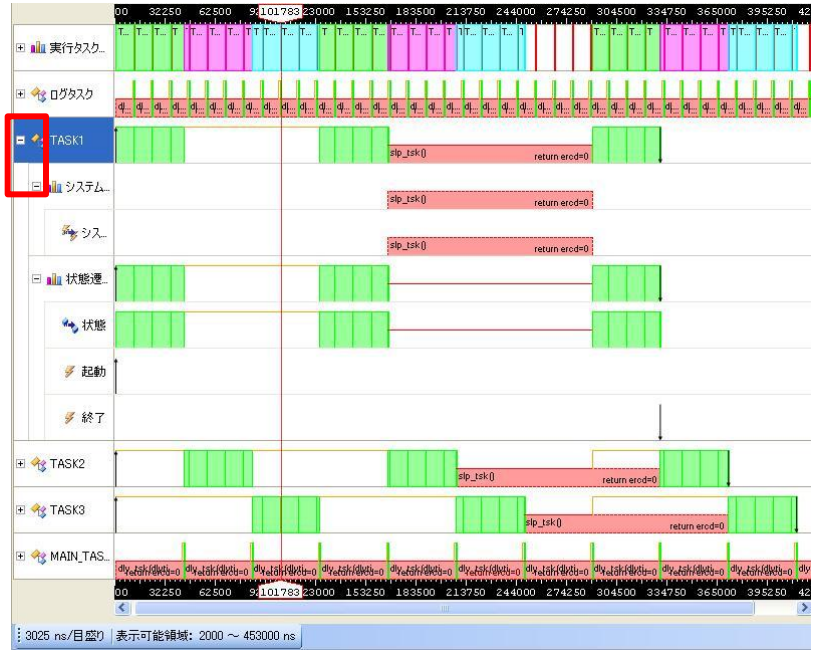


拡大



- 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

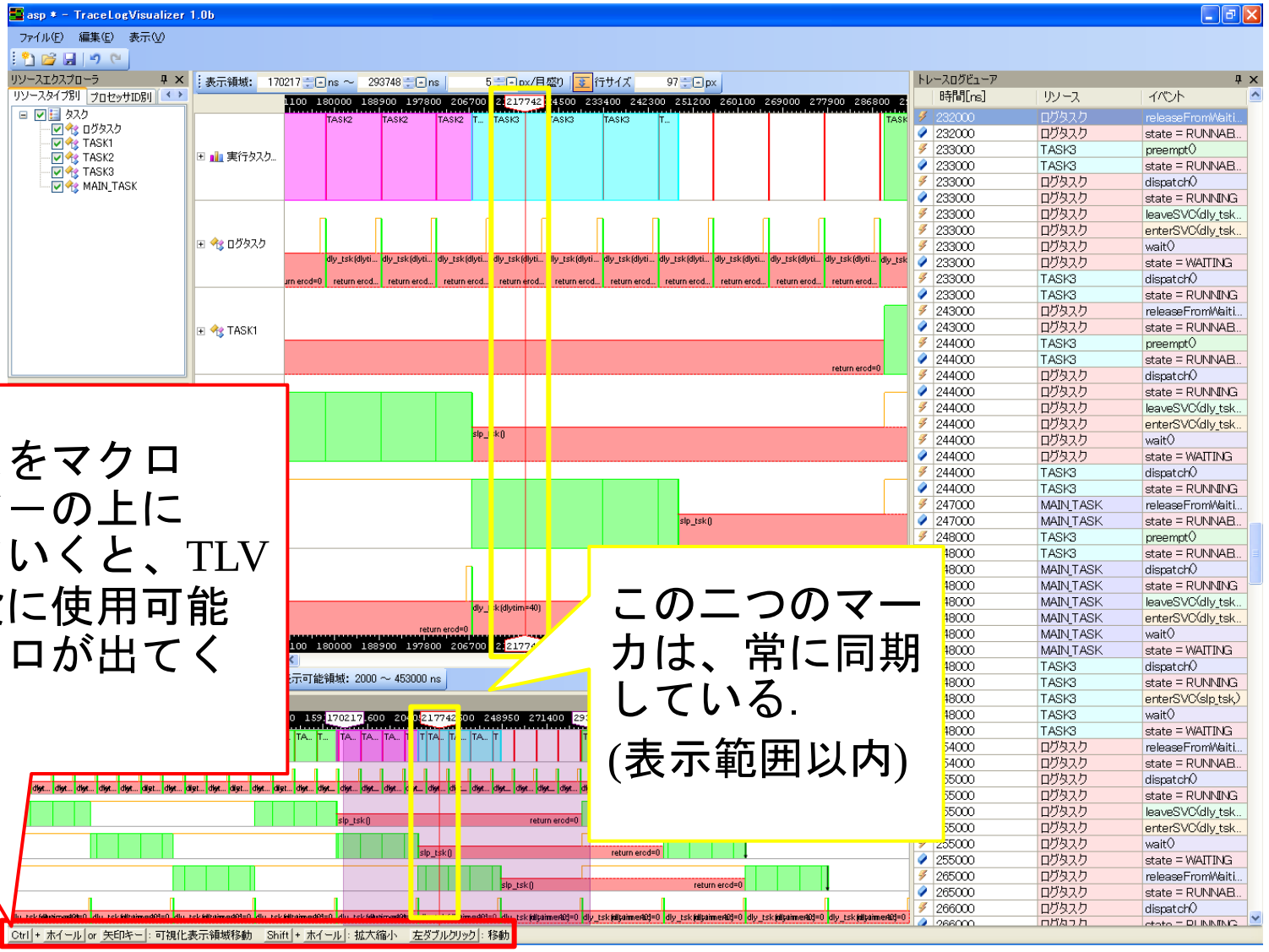
・リソースのタブを開くことによって、詳細項目ごとの振舞いを見ることができる.

・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

マクロビュアー



マウスをマクロビュアーの上に持っていくと、TLVの下段に使用可能なマクロが出てくる.

この二つのマーカは、常に同期している.
(表示範囲以内)

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
- 詳細検索

その他ー各画面位置移動、表示、非表示

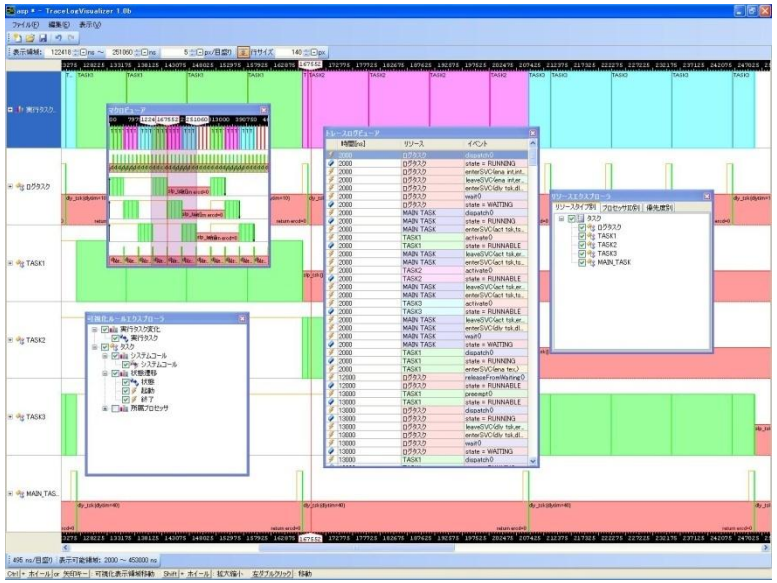
・可視化情報、リソース情報、トレースログ画面、
　マクロビューアのみ可能.

・表示は「メニュー」の「表示」でクリック.

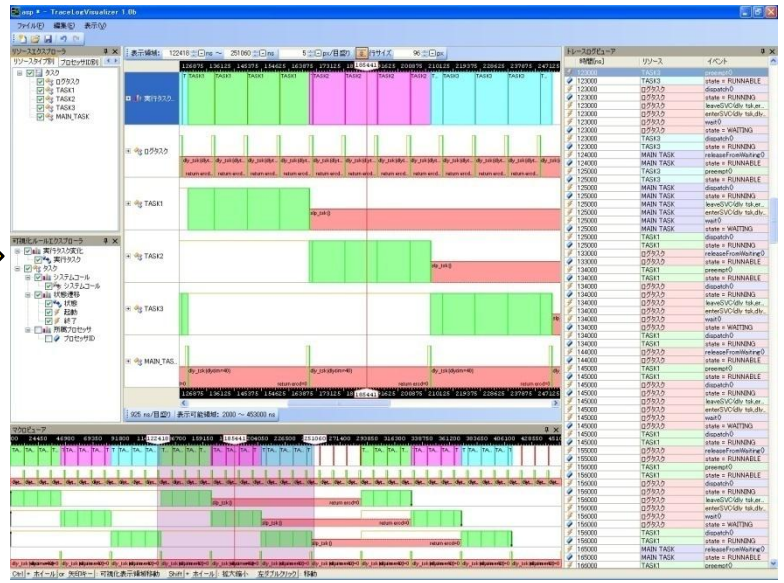
・位置移動は各画面をドラッグ＆ドロップで移動.

・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



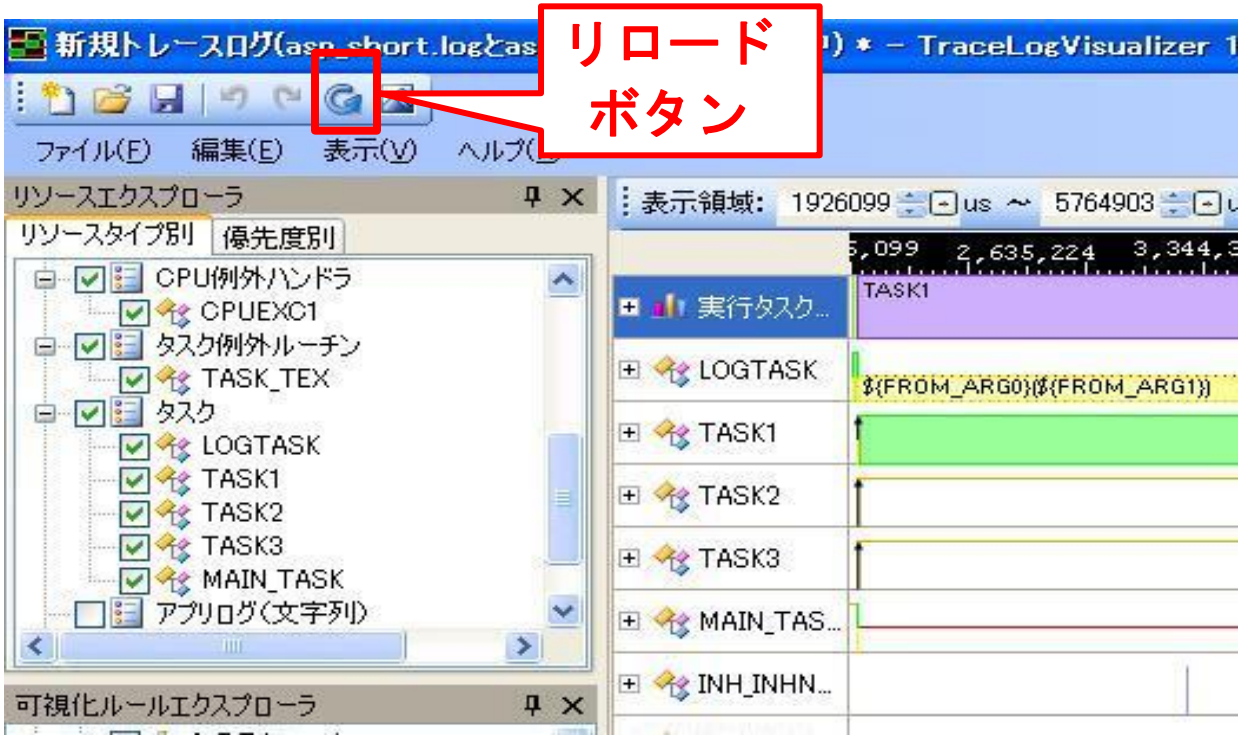
元の位置に移動した画面



その他 ― リロード

リロード：

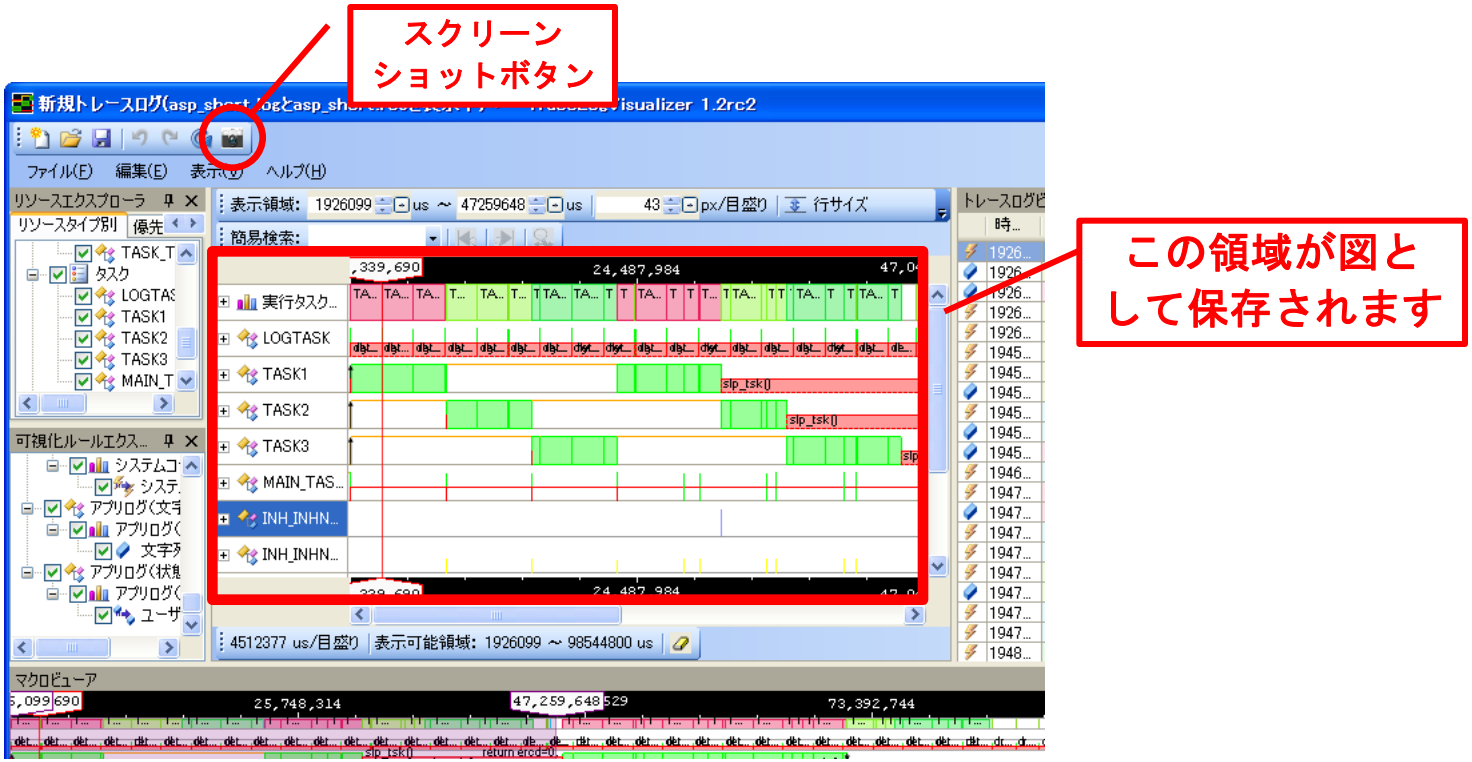
一度読込んだ .res ファイルと .log ファイルを
再度読込みます。両ファイルをドラッグ・ドロップ
することでもリロードが可能です



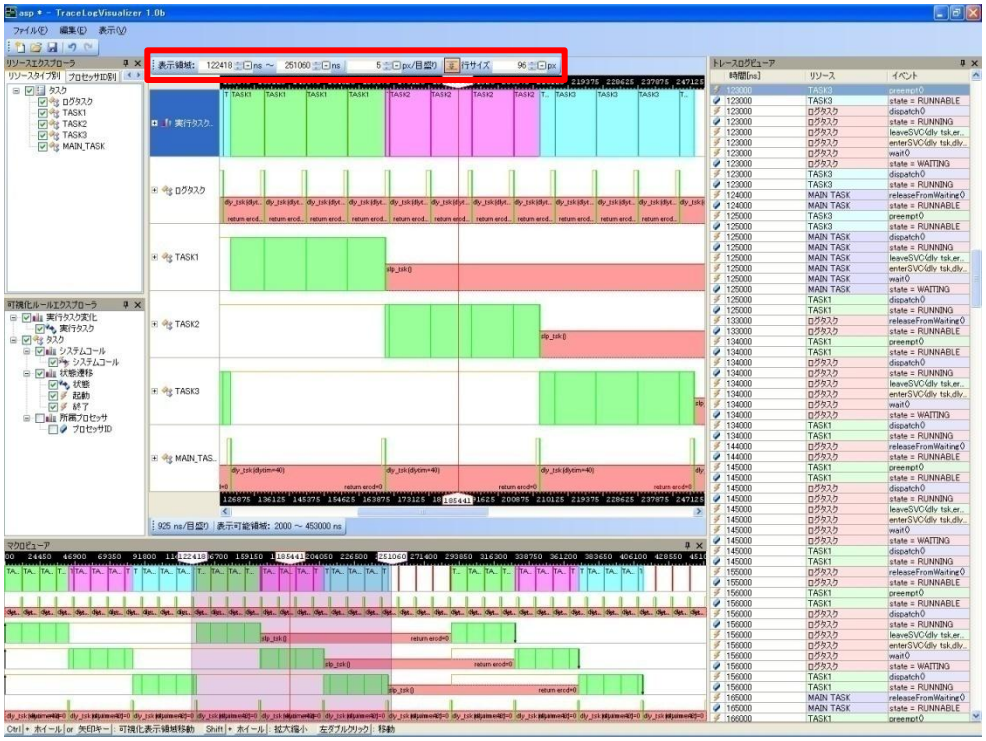
その他 ー スクリーンショット

・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。



その他ーツールバー



※ ツールバーの左から順に

- ns/目盛の拡大、縮小
  - ＋、ーボタンで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
  - ＋、ーボタンで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

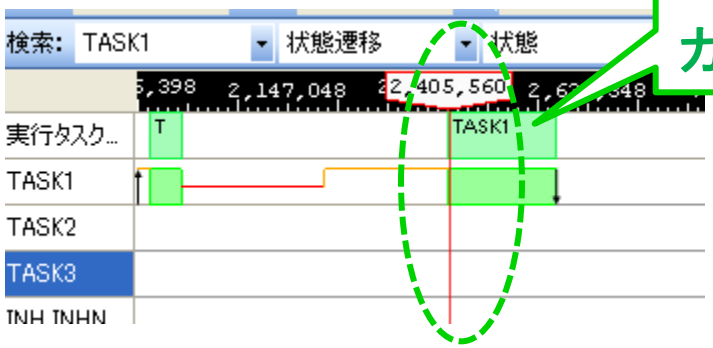
その他 – イベント検索機能：簡易検索

- **簡易検索とは**

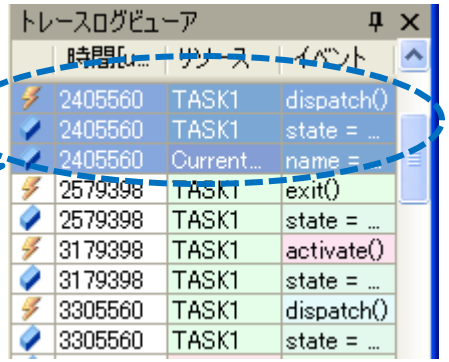
  リソース、ルール、イベントといった検索条件を指定し、
  条件に合致した時刻を探す機能

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

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

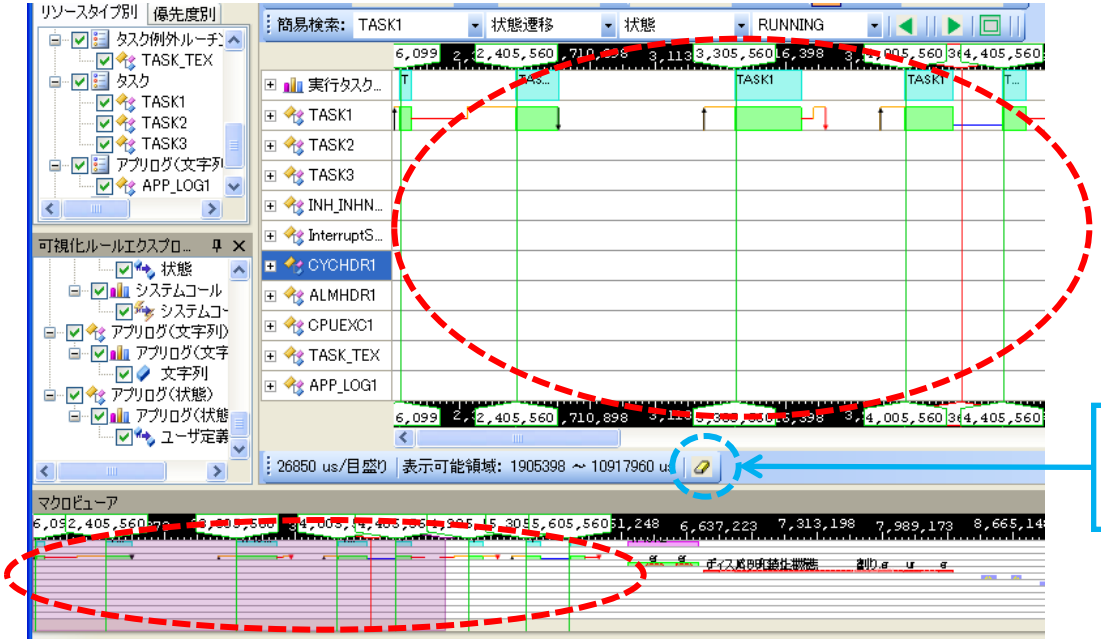
1. 後方検索
検索条件に合致する、現在から一番近い前の時刻へ移動する

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

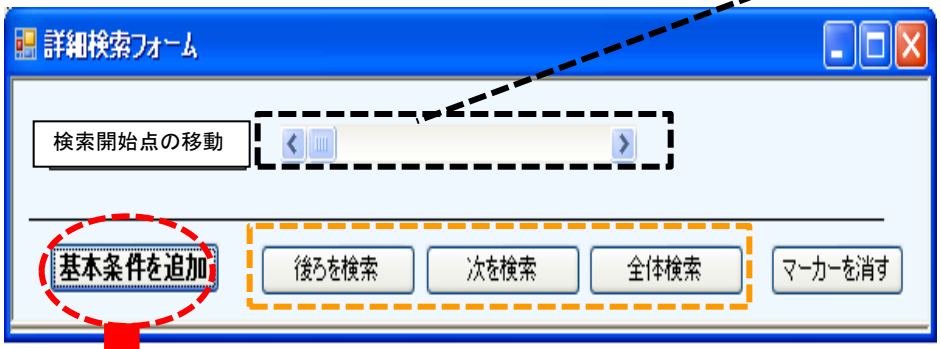
例1. TASK1 が実行状態へ遷移したイベントのうち以下を満たすもの
- そのイベント発生時刻の５秒以後に、TASK1が実行可能状態に遷移するイベントが発生

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

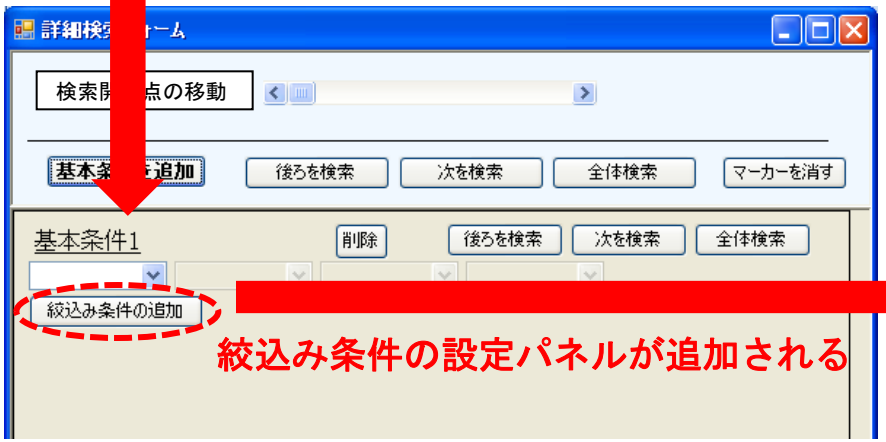
詳細検索フォーム

※初期状態

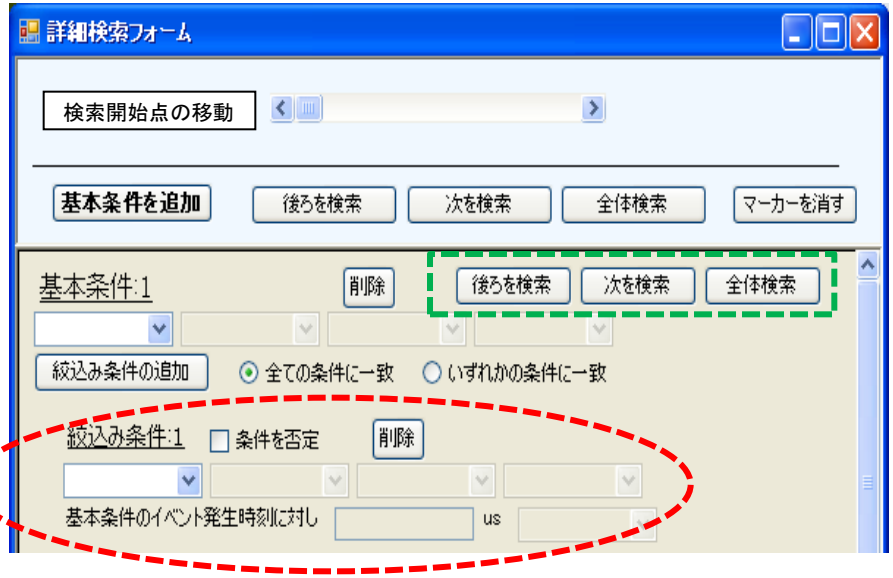
タイムライン上のカーソル位置を移動させる



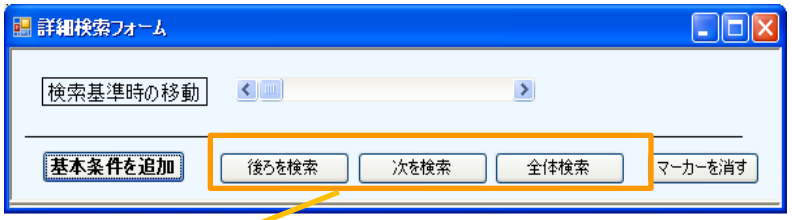
基本条件の設定パネルが追加される



絞込み条件の設定パネルが追加される



検索の実行



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

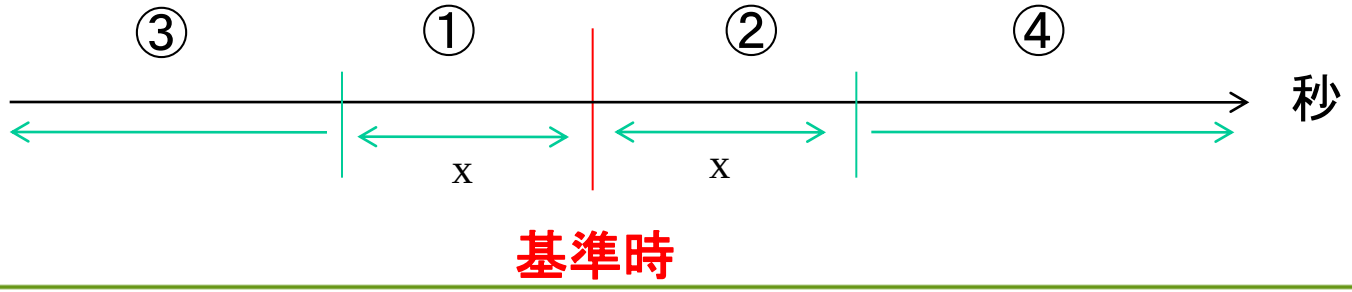
① x 秒以内（基準時以前）

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

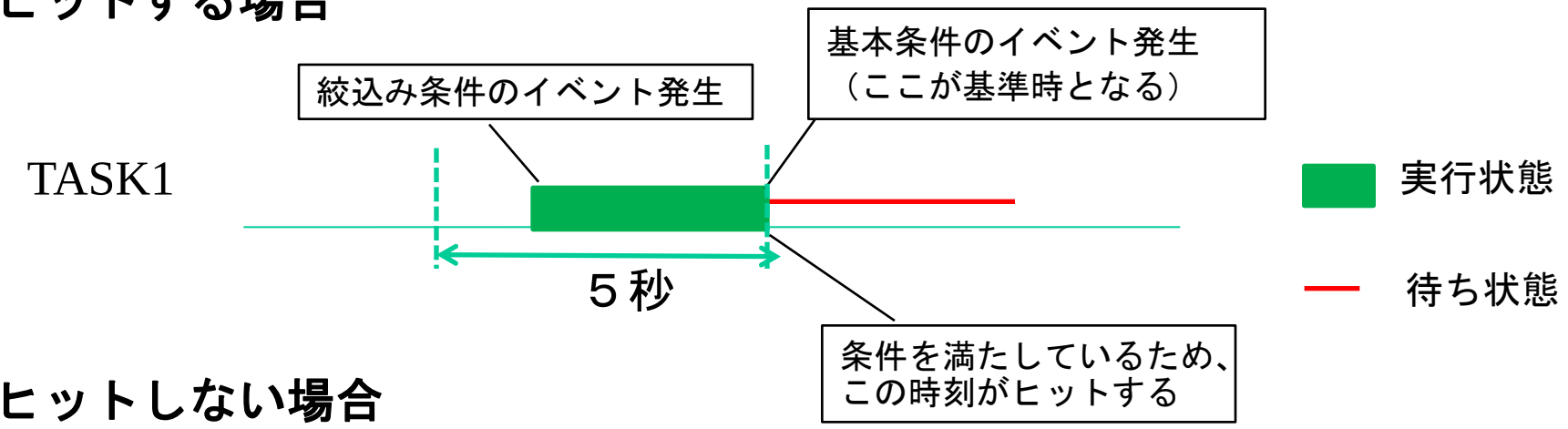


検索例： ｘ 秒以内（基準時以前）

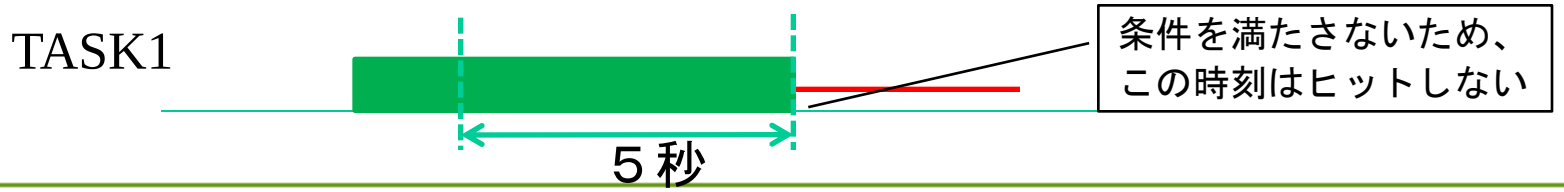
基本条件： TASK1が待ち状態になったイベント

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

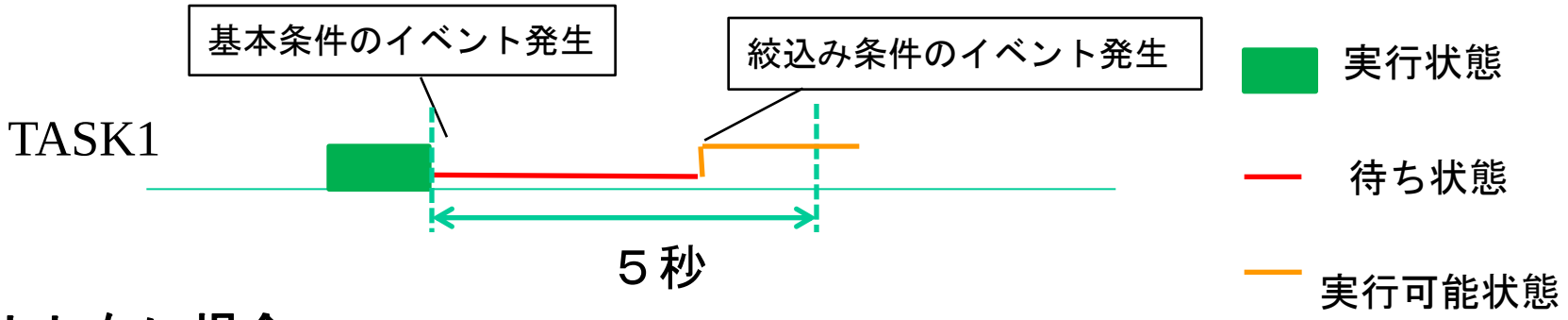


検索例： ｘ 秒以内（基準時以後）

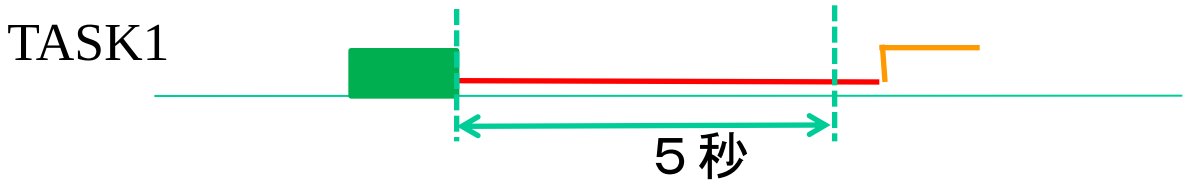
基本条件： TASK1が待ち状態になったイベント

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

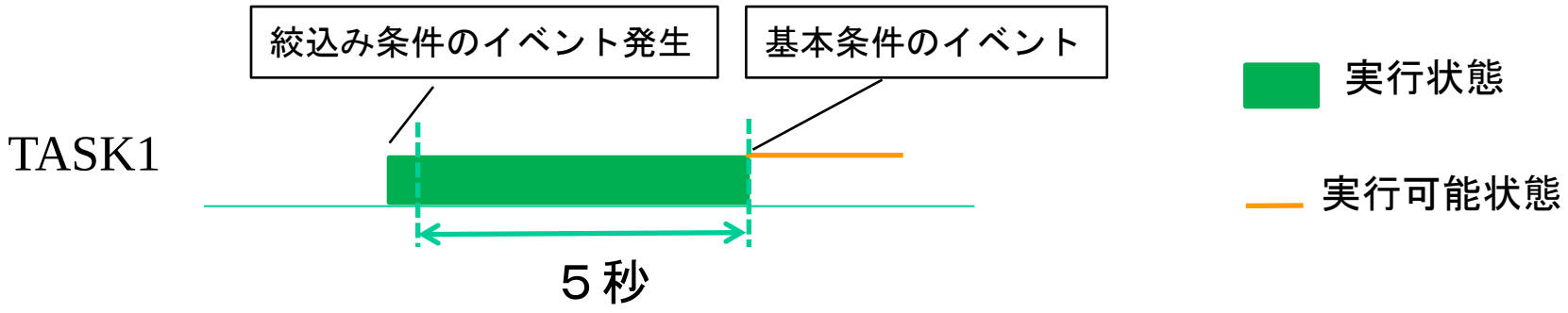


検索例： x 秒以上前

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



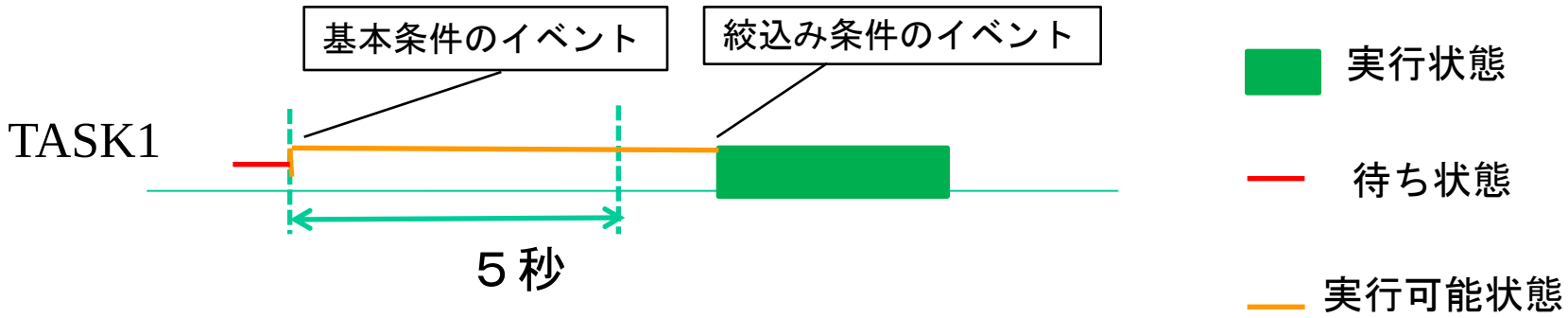
②ヒットしない場合

検索例： ｘ 秒以上後

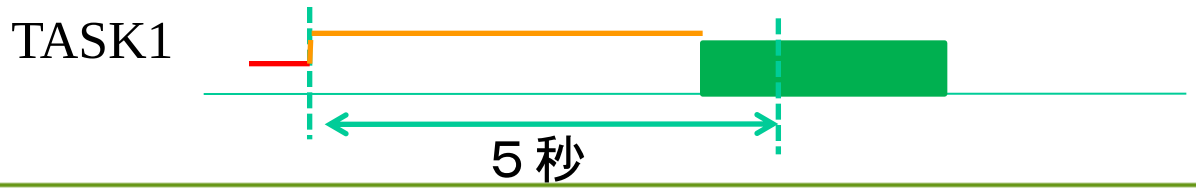
基本条件： TASK1が実行可能状態になったイベント

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