一般論文

Linux環境におけるマンマシン操作の記録/再生ツールの開発と適用事例

神戸事業所　技術第4部　交通システム第2課
塩谷　雅督

1. まえがき

当社では、列車の複雑多岐な運行ダイヤを作成、信号機や転轍機を制御し、列車の安全運行を支援する運行管理システムの開発を行っている。本システムは、社会的に重要な役割を担ったシステムであり、欠陥流出は、メーカーの信頼失墜や人命にまで関わる可能性がある。そのため、ソフトウェアの信頼性確認に多くの手数と時間をかけている。加えて、列車走行に必要な連動制御パターンは、単純構造の駅でも数百項目となるため、最終顧客である鉄道会社には、膨大な承認図・試験エビデンスの提出が必要である。

しかし、試運転期間中の仕様変更/追加の要求、ダイヤ改正や設備変更の改造工事が輻輳し、業務負荷が集中することも度々ある。

そこで、試験効率向上を目的として、試験工数削減・試験における人為的ミスの防止に取り組み、幾つかの施策を行った。

本稿では、運行管理ビジネスで採用している多様なLinuxディストリビューション環境で動作するマンマシン操作の記録/再生ツールの開発及びその適用事例を紹介する。

2. マンマシン操作の記録/再生ツールの開発

ツール開発にあたっては、実機での健全性確認試験においても使用することを前提とし、以下の仕様要件を設定した。

① 多様なLinuxディストリビューション環境で動作
② 実機試験で使用できる
③ 汎用ソフトウェアを使用しない
④ 仮想Linux環境でも動作
⑤ 緻密な操作も記録/再生可能
⑥ 試験エビデンスを自動取得ツールと連携できる
⑦ 試験合否自動判定ツールと連携できる

仕様要件を満たすため、Linux Input SubsystemとX Window Subsystemのライブラリであるxlib、XTEST Extension Libraryを利用して開発を行った。

2.1 Linux Input Subsystemの利用

Linux Input Subsystemとは、デバイスドライバとアプリケーション間の入出力をイベントデバイスファイルを通して行うためのLinuxカーネルの一部である。キーボード入力、マウス操作をアプリケーションが認識するまでの流れを図1に示す。

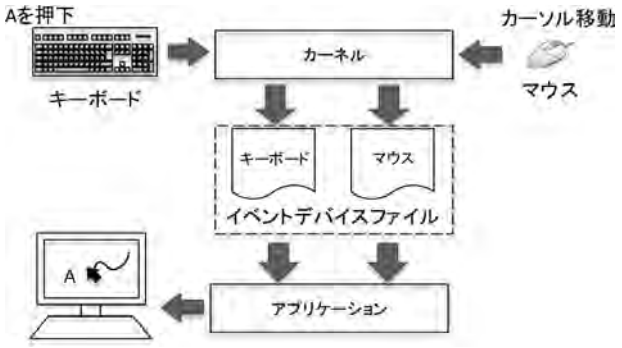

図1. キーボード入力を認識するまでの流れ

カーネルは、デバイスドライバを通してデバイスの入出力を解釈し、イベントデバイスファイルに出力する。イベントデバイスファイルは、接続されているデバイス毎に存在し、アプリケーションはこれらのイベントデバイスファイルを通して、デバイスの動作を認識することで、入力内容の表示、マウスカーソルの移動を行っている。イベントデバイスファイルに出力する内容は、/usr/include/linux/input.hに定義する。定義内容を図2及び表1に示す。

```
struct input_event {
        struct timeval time;    ①
        __u16 type;             ②
        __u16 code;             ③
        __s32 value;            ④
};
```

図2. イベントデバイスファイル出力定義

表1. イベントデバイスファイル出力定義の説明

番号	説明
①	イベントの発生時間を示す。
②	キーボード入力、マウスクリック、マウスカーソル移動などイベントの種類を示す。
③	②イベントの内容を示し、②がキーボード入力の場合はキーの種別（A,B,Cなど）、②がマウスクリックの場合はクリックの種別、マウスカーソル移動の場合は方向軸を示す。
④	②がキーボード入力またはマウスクリックの場合、③の内容に対して押したか離したかを示し、②がマウスカーソル移動の場合は現在位置からの移動量を示す。

ツール開発にあたり、図3に示すように、イベントデバイスファイルのインプットイベントを外部ファイルへ記録し、再生時は記録したインプットイベントをイベントデバイスファイルへ出力し、デバイスの入力を模擬するツールを試作した。

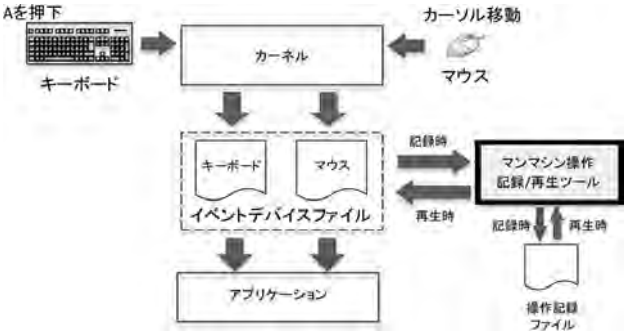

図3. 試作したツールの動作例

試作したツールで動作検証をした結果、キーボード入力を正確に再生することはできたが、緻密な画面のマウス操作は正確に再生することができなかった。

マウスカーソル移動時に、デバイスファイルに出力されるインプットイベントは現在位置からの移動軸と移動量である。このため、記録時と再生時でマウスカーソルの開始位置が異なると、意図しない位置へマウスカーソルが移動してしまい、正確な再生ができなかった。そこで、マウスカーソルの移動は、2.2項に記述するX Window Systemのライブラリであ るXlibとXTEST Extension Libraryを使用することとした。

2.2 X Window Systemの利用

X Window Systemとは、UNIXで標準的なGUI環境を実現するための仕組みである。X Window Systemはクライアント･サーバー方式で動作するシステムであり、サーバーはディスプレイやマウスなどの入出力デバイスを管理している。クライアントはディスプレイ上に描画する情報を作成する。クライアントからサーバーに対して、作成した描画の情報を送信し、サーバーがディスプレイに出力することで、画面の表示を行っている。画面上に表示されるマウスカーソルも同様の仕組みで移動し、表示される。Xサーバーと Xクライアントの関係を図4に示す。

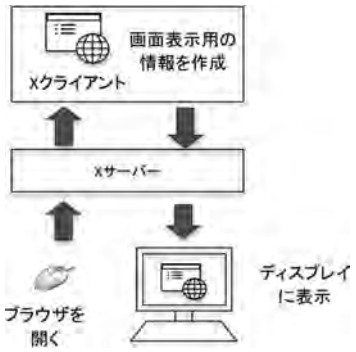

図4. XサーバーとXクライアントの関係

2.1項に述べたとおり、インプットイベントの記録だけではマウス操作の正確な再生は実現できないため、X Window Systemを利用して、ディスプレイ上の座標を取得し処理することにした。

Xlib、XTEST Extension Libraryは、Xクライアント用のライブラリであり、Xサーバーに対する通信を簡単に実現できる。ツールの改修にはこれらのライブラリを使用した。

マウス操作を正確に再生するには、デバイスファイルへのインプットイベントに加えて、マウスカーソルの位置を記録する必要がある。マウスカーソルの正確な位置を記録するために、Xlibで提供されているXQueryPointer関数を利用した。XQueryPointer関数は、Xサーバーに対して現在のマウスカーソルの座標を要求する関数である。マウス操作のインプットイベント出力時に、XQueryPointer関数で座標位置を取得し、操作記録ファイルにインプットイベントに加えて座標を記録するようにツールを改修した。マウスカーソルの座標記録時のツールの動作例を図5に示す。

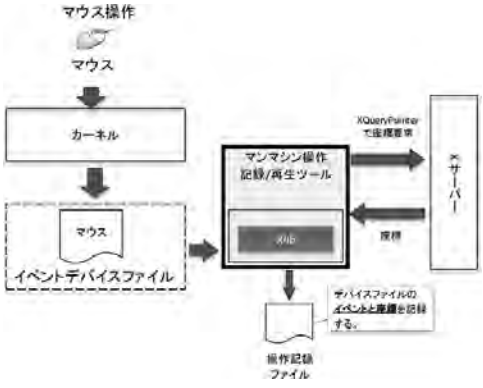

図5. マウスカーソルの座標記録時のツールの動作例

再生時は、記録した座標にマウスカーソルを移動させる必要があるため、XTEST Extension Libraryで提供されているマウス操作を模擬することができる関数を利用した。表2に提供関数の説明を示す。

表2. XTEST Extension Library提供関数

関数名	説明
XFakeMotionEvent	指定座標へのマウスカーソル移動のフェイクイベントを発行する。
XFakeButtonEvent	マウスクリックのフェイクイベントを発行する。（右、左、ミドル、ホイール）

図6に示すとおり、操作記録ファイルに記録されているインプットイベントがマウスカーソル移動であれば、記録した座標へマウスカーソルを移動するフェイクイベントをXサーバーに通知する。インプットイベントがマウスクリックであれば、マウスクリックのフェイクイベントをXサーバーに通知し、マウス操作の再生を行うようツールを改修した。

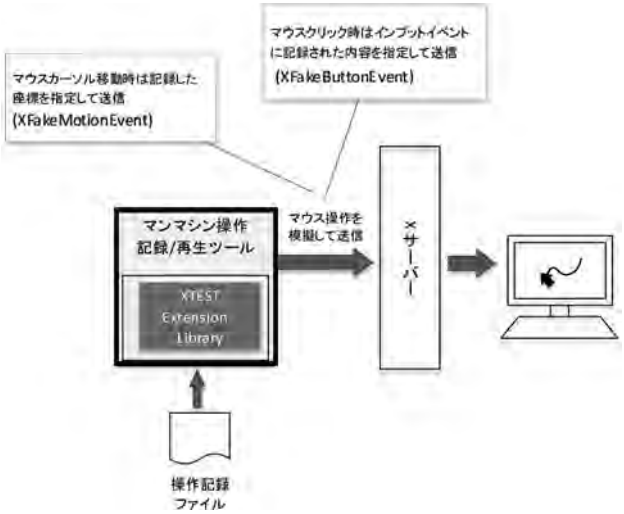

図6. マウス操作再生時のツールの動作例

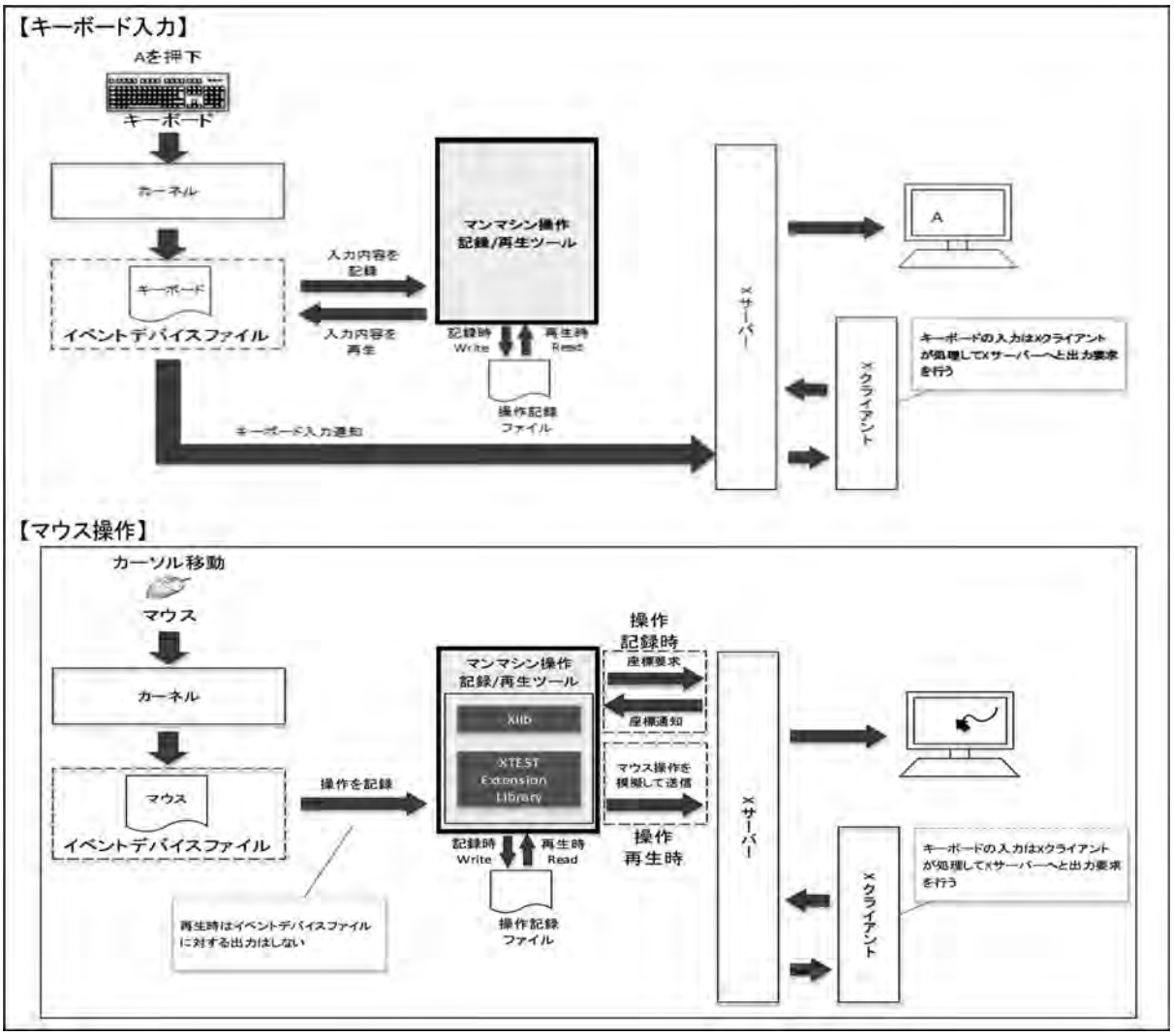

図7．マンマシン操作の記録/再生ツールの構成

ツールの動作検証を行った結果、マウス操作の正確な再生を実現することができた。

今回作成したツールの構成を図7に示す。

3. ツール適用事例

マンマシン操作の記録/再生ツールを適用した事例を紹介する。

3.1 回帰試験の自動化

試験条件の設定、試験手順を分割して記録することで、試験条件の再現や試験組み合わせ手順の再生を容易に実現することができる。同時に試験エビデンスを取得し、結果の差異チェックによる合否判定を出力する回帰試験の自動化に取り組んだ事例について紹介する。

(1) 試験条件の自動設定

回帰試験は、試験記録時と同一の条件を設定する必要がある。手動での設定は時間がかかる上、ヒューマンエラーが発生することも考えられるため、条件設定は自動で行えるよう、シェルスクリプトを作成した。

これにより、容易に同じ試験条件で回帰試験を実施できるようになった。

(2) エビデンス自動取得

試験エビデンスの取得漏れを防ぐために、試験エビデンスを自動で取得する仕組みを作成した。

マンマシン操作の記録/再生ツール実行時にシェルスクリプトを引数として指定することにより、指定された挙動でシェルスクリプトを実行する機能を追加した。

この機能を利用してエビデンス取得用のシェルスクリプトを用意すると、再生時は、記録時と同一のタイミングでエビデンスを自動取得できる。

(3) 試験結果の合否自動判定

(2)を利用して取得したエビデンスを比較し、試験結果の合否を自動判定するシェルスクリプトを作成した。合否自動判定シェルスクリプトのイメージを図8に示す。

図8. 合否自動判定シェルスクリプトのイメージ

合否判定用シェルスクリプトは、画面ハードコピー、帳票、ダイヤグラム図、ログなど様々な種類のエビデンスを比較して差異のある箇所を出力する。画面ハードコピーに差異が存在する場合の例を図9に示す。

(4) 回帰試験の自動化の応用

改造時の回帰試験に活用するだけでなく、OS移行を伴うハードウェア更新工事においても、旧装置とOS移行後の

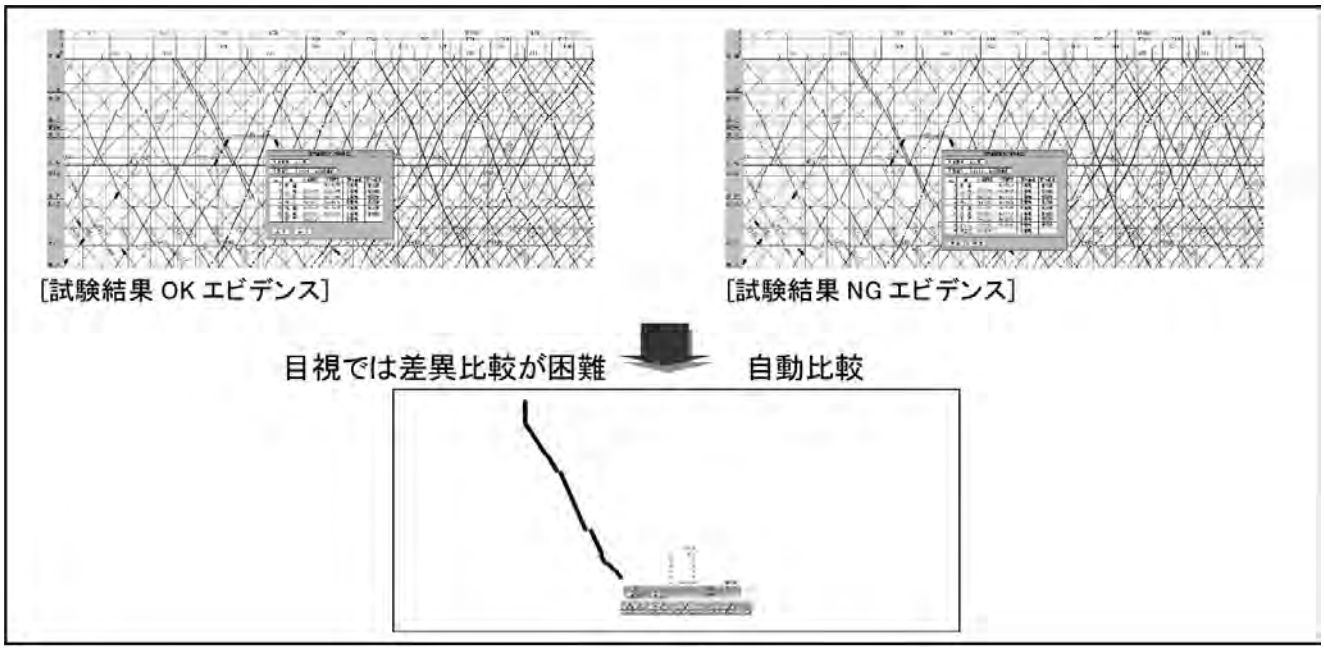

図9. 画面ハードコピー差異出力の例

新装置での動作比較に活用した。差異確認に活躍し、試験工数の削減・正確な比較結果の抽出に効果を発揮した。

3.2　マンマシン操作を伴う負荷測定試験

マンマシン操作を伴う負荷測定試験をマンマシンの記録/再生ツールを用いて行った。記録しておいたマンマシン操作を長時間繰り返し、メモリリーク等を測定した。人手では困難な連続操作を、効率的に実現することができた。

4. むすび

当初設定したツールの仕様要件を十分に満たした上で、回帰試験の自動化、負荷測定試験に応用活用が可能なツールを製作することができた。操作記録や動作設定の組み合わせにより、使用方法のバリエーションは無限にある。今後の展開としては、シェルスクリプトの工事毎カスタマイズの容易性向上を計画している。

最後に、本開発及び執筆にあたり、支援頂いた関係各位に深く感謝を申し上げる。

執筆者紹介

塩谷　雅督　　シオタニ　マサトク
2012年入社。主に運行管理システムのソフトウェア開発に従事。現在、神戸事業所技術第4部交通システム第2課。