

# Framework 及び Excel を使用した自動テスト効率化試行

古江 智和 谷口 裕一

株式会社フォーラムエイト宮崎支社 TestGroup 〒889-2155 宮崎市学園木花台西 2-1-1

E-mail: furue@forum8.co.jp taniguchi@forum8.co.jp

あらまし 自動テストツール WinRunner に, EMOS Framework 及び Microsoft Excel を組み合わせて作成, 実施したテスト事例を紹介する。EMOS Framework を組み込むことにより「自動テストメンテナンスの効率化」を、Excel VBA との関係により「柔軟なテスト結果比較」を達成することを目的とした。

キーワード 自動テスト, テストメンテナンス, テスト結果, WinRunner, EMOS Framework, Excel

## Approach for efficient Automated Testing with Framework and Excel

Tomokazu Furue Yuichi Taniguchi

† FORUM8 INC. Miyazaki Branch TestGroup 2-1-1 Gakuenkibanadai-nishi, Miyazaki, 889-2155 Japan

E-mail: † furue@forum8.co.jp taniguchi@forum8.co.jp

**Abstract** It introduces the test case of WinRunner with EMOS Framework and Microsoft Excel. This test case has two purposes. One of them is 'Efficient maintenance for Automated Testing'. We have incorporated EMOS Framework in Winrunner. And another is 'Flexible comparing of test results'. We have use of Microsoft Excel for it.

**Keyword** Automated Testing, Test Maintenance, Test results, WinRunner, EMOS Framework, Excel

### 1. まえがき

弊社ではおよそ 2 年前から自動テストツールを使用して開発製品のテストを行っているが、自動テストの作成及び実施を繰り返す中で下記に示すような問題が次第に顕著になっていった。

- ・製品更新時の自動テストメンテナンス工数
- ・自動テスト結果（特に数値比較結果）の柔軟性の欠如

本論文では上記問題の改善を目的として作成した自動テスト事例を紹介する。

2.ではまず自動テストツールに組み込んだ EMOS Framework の概要を説明する。3.では本テスト事例の概要を TestCycle に沿って説明する。4.ではテスト結果比較に用いた手法について説明し、5.ではテストメンテナンス工数削減のために使用した手法を解説する。

尚、本事例で使用したツール類及び AUT(Application Under Test)は下記の通り。

- ・Mercury Interactive WinRunner 7.50
- ・Microsoft Excel 2000
- ・EMOS Framework Winrunner Add In 1.4.0
- ・Forum8 UC-win/Section Ver1.0.0

### 2. EMOS Framework の概要

EMOS Framework は Free の WinRunner Add-in でインターネット上から誰でも取得することが出来る<sup>(1)</sup>。インストール後は WinRunner 起動時に追加アドインとして組み込むことで利用可能になり、以下のような特徴を持つ。

- ・テストデータ及びテストフローの制御を TestTable(Excel)から取得してテストを実行する
- ・Framework の仕様を意識してテストスクリプトをデザインすることにより、メンテナンス効率の高い自動テストセットを作成することができる。

EMOS Framework の System 構造<sup>(2)</sup>を図 1 に示す。

ユーザ(テスト作成者)は EMOS Framework が提供する Template を参考にスクリプトを構築し、テストフロー及びテストで使用する要素(キャプション、ボタン名称等)を Excel WorkSheet にテーブルとして作成する。テーブル例を図 2 に示す。

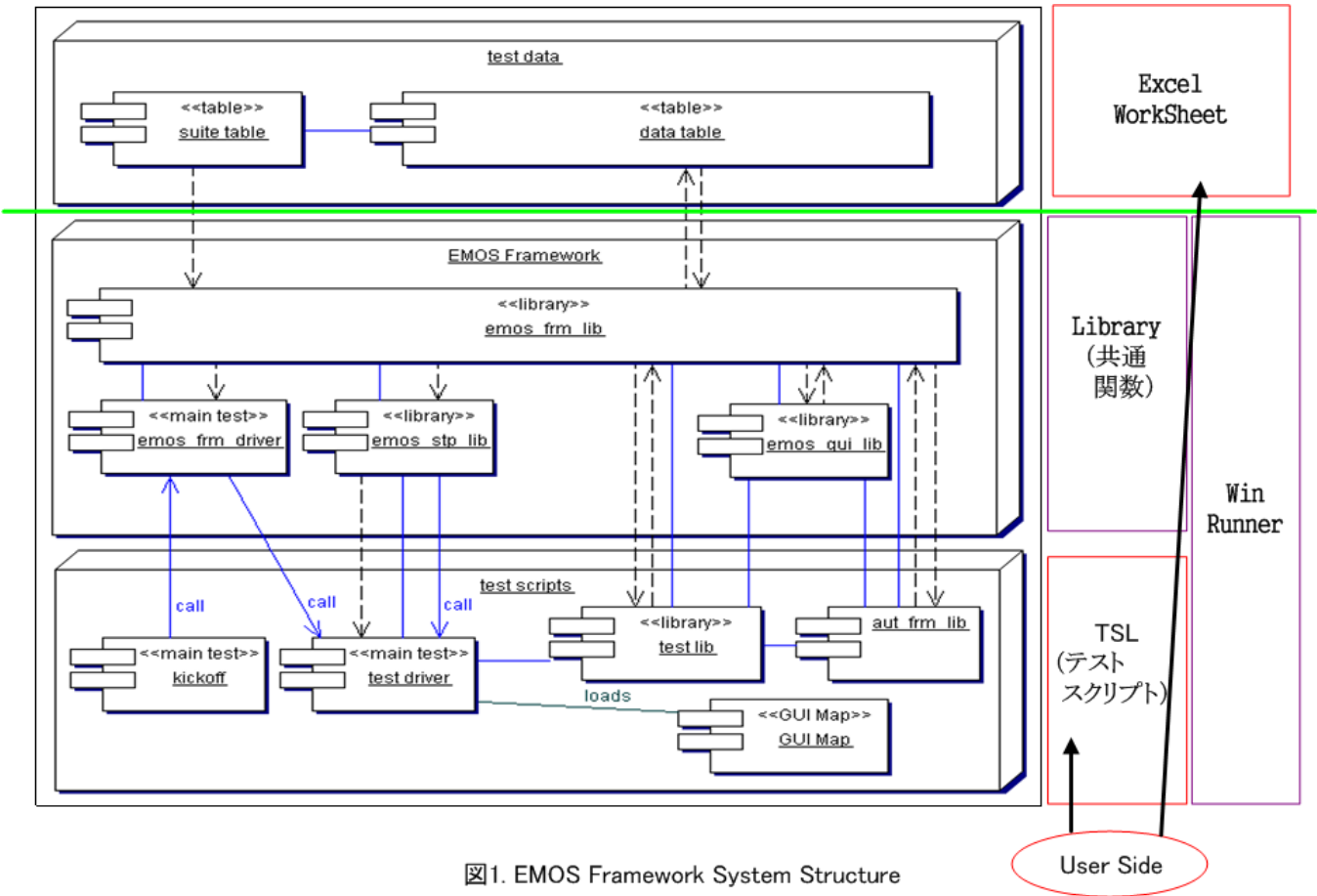


図1. EMOS Framework System Structure

1	A	B	C	D	E	F	G
2	IDX	Name	Description	1	Con1	Con2	Con3
3	x	Description	this row defines test case descriptions	Make Test Section 1	BASE断面(Con1)	BASE断面(Con2)	BASE断面(Con3)
4	x	Testsequence	this row defines the sequence for the execution of test blocks in a test case	LINK""Con1	MainTab	MainTab	MainTab
5				LINK""Con2	MainSpdBtn	MainSpdBtn	MainSpdBtn
6				LINK""Con3	CreateNew	CreateNew	CreateNew
7				LINK""101	LINK""T4_SectionD1.xls	LINK""T4_SectionD1.xls	LINK""T4_SectionD1.xls
8				LINK""102	"Con1	"Con2	"Con3
9				LINK""103			
10				LINK""201			
11				LINK""202			
12				LINK""302			
13				LINK""304			
14	x	MainTab	Tab選択				
15		label	Tab文字列		Section	Section	Section
16		expval	期待値 O X (check mode)		<<Clear>>	<<Clear>>	<<Clear>>
17	x	Popup	メニュー選択				
18		menu1	メニュー文字列				
19		menu2	sub menu文字列				
20		menu3	sub sub menu文字列				
21		expr	メニューを有効にする為のStatement				
22		expval	期待値 O Δ X (check mode)				
23	x	MainSpdBtn	処理選択(CHK=チェックモード)				
24		tbar	ツールバー分類名		Section	Section	Section
25		btn	ボタン名(ボタン機能名)		CREATE	CREATE	CREATE
26		expval	期待値 O Δ		<<Clear>>	<<Clear>>	<<Clear>>
27	x	CreateNew	新規名称				
28		label	ダイアログキャプション		New Section	New Section	New Section
29		name	名称		Con1	Con2	Con3
30		ddlb	comboboxエントリ		SFHB-III	SFHB-III	SFHB-III
31		button	押下するボタン名		OK	OK	OK

← Test ID

Test Sequence  
テストのフローを記述

変数

Function  
Arg 1  
Arg 2  
Arg n

図2. TestTable

### 3. テスト事例の概要

AUT(Application Under Test)には 1.で述べた通り弊社の製品 UC-win/Section(Ver1.00.00)を使用した。テスト内容は、AUT 内で種々の材料や形状毎に断面を作成、計算後に得られる部材力等が結果値として妥当かを検査するものである。

本テスト事例の Test Cycle を図 3 に示す。

本テスト事例では Excel WorkSheet を異なる 2 つの目的に使用していることに留意されたい。

#### (1) Test 結果比較用(期待値 WorkSheet)

結果を取得したいテストの要素名(ここでは断面 ID)と、結果期待値からなるリストとして構成されている。

シート例を図 4 に示す。

本テスト事例では、ここに記述されている要素のテスト結果のみを取得し、同じシートにテスト結果値を取り込んでいる。これらの WorkSheet の I/O には Framework は使用していない。また各 WorkSheet ファイルには結果比較に使用するための VBA コード(マクロ)が含まれている。

#### (2) Test 制御

図 3 の②(右側)に示される WorkSheet は Framework の TestTable として供されている。目的は主にテストフローの制御、GUI 周りの変数(ボタン、メニュー名称等)及び入力パラメータの提供である。内容は図 2 を参照されたい。

### 4. テスト結果の比較

結果比較において下記の要求事項を満たすような機能を ExcelVBA で作成した。

#### (1) 許容誤差の設定

ある程度の誤差は結果 OK と見なすよう、許容誤差をユーザが設定可能とした。

#### (2) 比較結果の Visual 化

比較結果において、期待値との完全一致、許容誤差内一致及び不一致の 3 種類が視覚的に判別出来る様な機能を設定した。

テスト結果例を図 5 に示す。

これらの機能は結果シートにマクロとして含まれており、結果を検討するユーザはこのシートを直接操作して検討することが出来る。

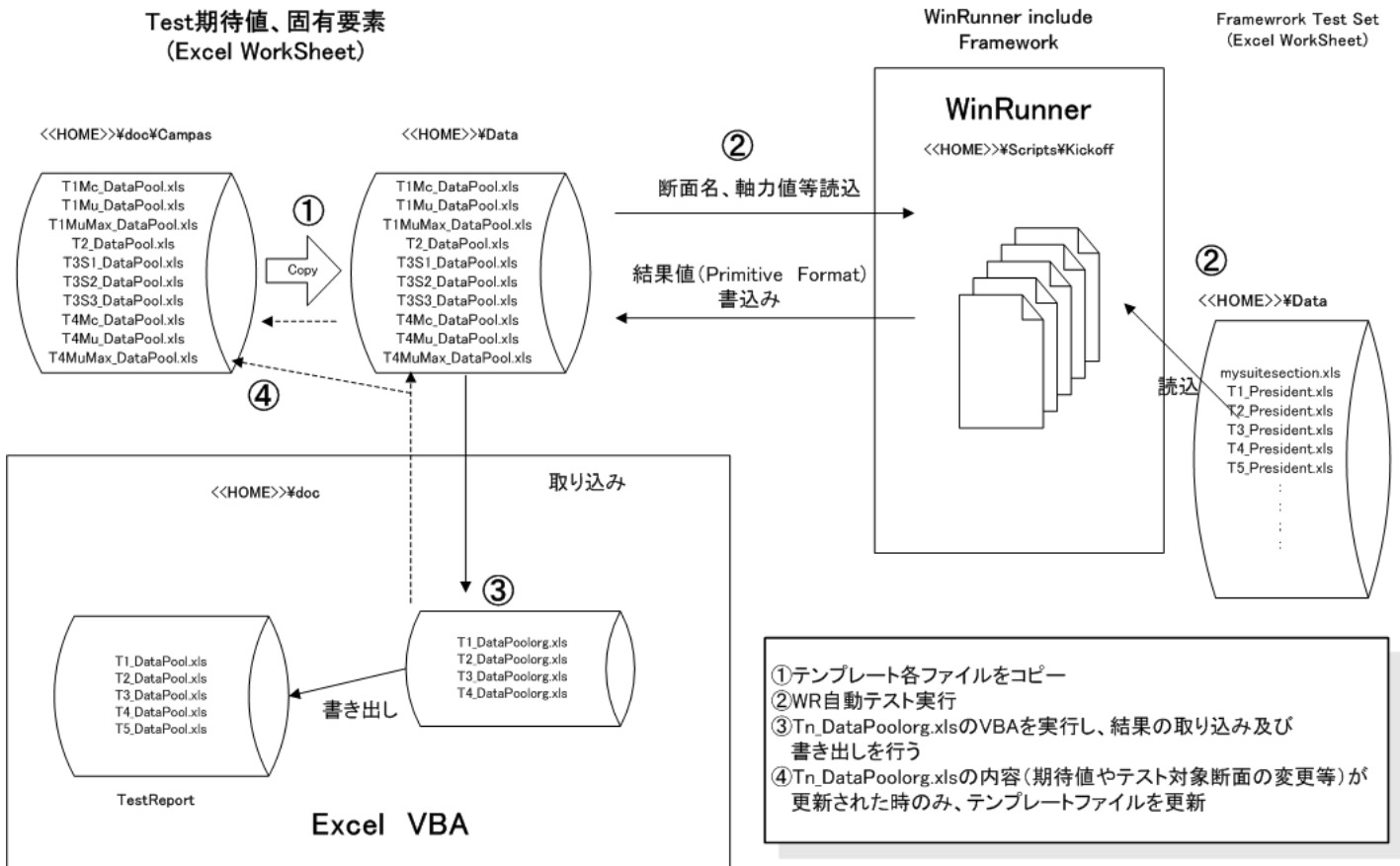


図3. Test Cycle

FLAG	ID	Name	0	Result1	5000	Result2	10000	Result3	Nmax	NmaxA	Re
	101	101 Con1(30N/mm2) + Reo	2152.746		5081.950		7003.623		39823.304	0.000	
	102	102 Con2(55N/mm2) + Reo	2175.071		5471.642		8211.627		71698.304	0.000	
	103	103 Con3(60N/mm2) + Reo	2176.753		5501.007		8302.654		78073.304	0.000	
	104	104 Con1 + PC(Inner)	5782.568		7735.991		8882.259		37246.680	0.000	
	105	105 Con1 + PC Rod(Inner)	2528.363		5356.029		7031.968		40108.150	0.000	
	106	106 Con1 + Steel	4682.920		6670.236		8185.339		46462.680	0.000	
S	107	107 Con1 + CFRP(10m)							38250.000		
S	107x	107x Con1 + CFRP(20m) N=20 Concrete broken							38250.000		
S	107y	107y Con1 + CFRP(20m) N=15 CFRP broken							38250.000		
S	107z	107z Con1 + CFRP(2m) N=20 CFRP slipped							38250.000		
	107a	107a Con1+Reos+CFRP(10m) N=20s CFRP broken	5981.080		8271.682		9791.563				
	107b	107b Con1+Reos+CFRP(20m) N=20 Concrete broken	10874.133		9886.334		9268.739				
	107c	107c Con1+Reos+CFRP(20m) N=15 CFRP broken	9384.963		8968.678		8633.209				
	107d	107d Con1+Reos+CFRP(2m) N=20 CFRP slipped	1299.805		3794.909		5645.737				
	108	108 Con1 + Reo + PC(Inner)	7553.552		9201.810		9988.562		38819.983	0.000	
	109	109 Con1 + Reo + PC(Outer) Delt_Sigma_Pe=0	4587.144		7090.631		8586.587		37593.704	0.000	
	109x	109x Con1 + Reo + PC(Outer) Delt_Sigma_Pe=400	5504.787		7828.530		9144.743		36701.864	0.000	
S	110	110 Con1 + Reo + PC Rod(Inner) Sigma_Pe=0									
	111	111 Con1 + Reo + Steel	6369.432		8227.733		9566.786		48035.984	0.000	
	112	112 Con1 + Reo + CFRP(10m) CFRP broken	7687.120		9780.046		11015.290		39823.304	0.000	
S	113	113 Con1 + Reo + AFRP									

図4. 期待値WorkSheet

ID	Name	0	Result1	5000	Result2	10000	Result3
101	101 Con1(30N/mm2) + F	2152.746	2152.746	5081.950	5081.950	7003.623	7003.623
102	102 Con2(55N/mm2) + F	2175.071	2175.071	5471.642	5471.642	8211.627	8211.627
103	103 Con3(60N/mm2) + F	2176.753	2190.753	5501.007	5501.007	8302.654	8302.654
104	104 Con1 + PC(Inner)	5782.568	5882.568	7735.991	7735.991	8882.259	8882.259
105	105 Con1 + PC Rod(Inner)	2528.363	2528.363	5356.029	5356.129	7031.968	7031.968
106	106 Con1 + Steel	4682.920	4682.920	6670.236	6670.236	8185.339	8185.339
107	107 Con1 + CFRP(10m)						
107x	107x Con1 + CFRP(20m) N=20 Concrete broken						
107y	107y Con1 + CFRP(20m) N=15 CFRP broken						
107z	107z Con1 + CFRP(2m) N=20 CFRP slipped						
107a	107a Con1+Reos+CFRP(10m) N=20s CFRP broken	5981.080	5981.080	8271.682	8271.682	9791.563	9791.563
107b	107b Con1+Reos+CFRP(20m) N=20 Concrete broken	10874.133	10874.133	9886.334	9886.500	9268.739	9268.739
107c	107c Con1+Reos+CFRP(20m) N=15 CFRP broken	9384.963	9384.963	8968.678	8968.678	8633.209	8633.209
107d	107d Con1+Reos+CFRP(2m) N=20 CFRP slipped	1299.805	1299.810	3794.909	3794.909	5645.737	5645.737
108	108 Con1 + Reo + PC(Inner)	7553.552	7553.552	9201.810	9201.810	9988.562	9988.562
109	109 Con1 + Reo + PC(Outer) Delt_Sigma_Pe=0	4587.144	4587.144	7090.631	7090.631	8586.587	8586.587
109x	109x Con1 + Reo + PC(Outer) Delt_Sigma_Pe=400	5504.787	5504.787	7828.530	7828.530	9144.743	9144.743

図5. 結果比較シート

## 5. テストメンテナンス

本事例では Framework の使用に加え, GUIMap を極力使用しないことにより, テストメンテナンスに要する工数を軽減することに留意した. これは自動テストのメンテナンスにおいて, GUIMap の修正に大きな工数がかかることを過去に経験していたためである. 具体的な手法としては, AUT の各コントロールに対する制御は可能な限り物理記述を使用するようデザインした. また物理記述に必要な変数(ボタン名称等)は Framework の機能を生かし, ExcelWorkSheet から取得している.

物理記述の例を図 6 に示す.

これにより, メンテナンス発生時の作業のほとんどは Framework で使用する ExcelWorkSheet の修正をすれば良いことになる.

本事例で発生した実際のメンテナンス作業としては, AUT の日本語版インターフェースがリリースされた時にテストセットを新たに作成したが, 英語版で使用した Worksheet を日本語に置き換えるだけで済んだ為, ほぼ一人日で完了した事例がある.

日英各インターフェース用の Worksheet を図 7 に示す.

## 6. むすび

本事例で紹介するようなテストセットを構築する場合, 一つの障害として初期投資の大きさがあげられよう. 物理記述を多用して共通に使用出来る関数を作成しておく手法は, GUIMap に依存する方法と比較すると初期工数が明らかに増大する. ある意味, 後々のメンテナンス工数と初期工数とのトレードオフと考えることも出来る.

しかしながら構築したテストの Primitive な部分(共通関数部分)は他のまったく異なるテストにも使用できる可能性が高いこと, テストスクリプトのロジックを追いかけながらメンテナンスする必要がないこと, 更には自動テスト自体がメンテナンスを行いながらも長期間に渡って使用しなければ元の取りにくい存在であることを考えれば, 検討に値する手法ではないかと考える.

## 文 献

- [1] EMOS Framwwork for WinRunner  
[http://groups.yahoo.com/group/EMOS\\_frame/](http://groups.yahoo.com/group/EMOS_frame/)
- [2] EMOS Framework Basics 1.ppt  
<http://www.emos.de/emos/homepage/content.nsf/downloadFRMslides?OpenPage>

TEST TABLE

IDX	Name	Description	Testsequence
1			push_finish
2	x	Description	
x	Testsequence		EVAL~AUT_CM_push_button("Finish")
4	x	SecSpdBtn	
5		tbar	
6		btn	
7		expval	
x	Popup		
9		menu1	
10		menu2	
11		menu3	
12		expr	
13		expval	

TSL

```

131 public function AUT_CM_push_button(in label)
132   #Active window内の引数で指定されたラベルを持つボタンを押下する
133   {
134
135     auto rc;
136     auto winlabel;
137     auto handle;
138     auto func_name = "AUT_CM_push_button";
139
140
141     rc = win_get_info("{class:window,active:1}", "handle", handle);
142     if ( rc != E_OK ) {
143       tl_step(func_name, FAIL, "Window情報取得失敗");
144       return rc;
145     }
146
147     rc=_set_window("{class: window,handle: " & handle & "}",5);
148     if ( rc != E_OK ) {
149       tl_step(func_name, FAIL, "WindowSet失敗");
150       return rc;
151     }
152     rc = button_press("{class: push_button, label: " & label & "}");
153
154     return rc;
155   }
156
157 public function AUT_CM_push_button_chk(in label)
158   #Active window内の引数で指定されたラベルを持つボタンがEnableか調べる
159   #return 0 - disabled 1 - enabled
160   {

```

図6. 物理記述例

A	B	C	D	E	F
1	IDX Name	Description	Section5	Section6	Copy/AndRen
2	x Testsequence	this row defines test case descriptions execution of test blocks in a test case	新規断面(Section5) MainTab MainSpdBtn CreateNew LNK""JP_N_T2_SectionD1.xls"Section5New	新規断面(Section6) MainTab MainSpdBtn CreateNew LNK""JP_N_T2_SectionD1.xls"Section6New	Copy/AndRen MainTab EVAL "list_select_item" TF8ListView"."<\$\$SEC_ORG\$\$" EVAL "list_select_item" TF8ListView"."<\$\$SEC_ORG\$\$" Popup MainSpdBtn EVAL "type("<\$\$SEC_DEST\$\$><RReturn>") EVAL "list_activate_item" TF8ListView"."<\$\$SEC_DEST\$
3					
4	x Main Tab	Tab選択 Tab文字列	Section <<Clear>>	Section <<Clear>>	Section <<Clear>>
5	expval	期待値 OI X (check mode)			
6	x Popup	メニュー選択 メニュー文字列			Copy To.. <<Clear>> <<Clear>>
7	expval	メニュー文字列			
8	menu1	sub menu 文字列			
9	menu2	sub sub menu 文字列			
10	menu3	メニューを有効にする為のStatement			click("Right".1), <<Clear>>
11	expval	期待値 OI X (check mode)			
12	x MainSpdBtn	処理選択のHK=チェックモード) ツールバー分欄名 期待値 OI X (check mode)	Section CREATE <<Clear>>	Section CREATE <<Clear>>	Section RENAME <<Clear>>
13	tabar	タブ名(ボタン機能名)			
14	x CreateNew	新規名称 ダイアログキヤンセルボタン 名称 comboBoxエントリ 押下するボタン名	New Section Section 5 SFHB-III OK	New Section Section 6 SFHB-III OK	
15	label	ダイアログキヤンセルボタン 名称			
16	name	ダイアログキヤンセルボタン 名称			
17	ddbb	comboBoxエントリ 押下するボタン名			
18	button	押下するボタン名			
19					
20					
21					
22					
23					
24					
25					
26					
27					
28					
29					
30					
31					
32					
33					
34					
35					
36					
37					
38					
39					
40					
41					
42					
43					
44					
45					
46					
47					
48					
49					
50					
51					
52					
53					
54					
55					
56					
57					
58					
59					
60					
61					
62					
63					
64					
65					
66					
67					
68					
69					
70					
71					
72					
73					
74					
75					
76					
77					
78					
79					
80					
81					
82					
83					
84					
85					
86					
87					
88					
89					
90					
91					
92					
93					
94					
95					
96					
97					
98					
99					
100					

図7. 日英両インタフェースの比較