3C-2

コンシューマ向け組込み機器のアプリケーション起動高速化検討
(An examination of fast application boot for consumer device)

出原　章雄†　　攝津　敦†　　伊藤　孝之†　　落合　真一†

三菱電機（株）情報技術総合研究所†

1.　はじめに

複雑な機能を必要とするコンシューマ向け組込み機器では、単純なリアルタイム OS(RTOS)から、メモリ保護やインターネット接続可能な高機能 OS への移行が進んでいる。このような背景から、組込み Linux が注目されている。RTOS を利用したコンシューマ機器では、ユーザからの入力（リモコンボタンの入力等）に対し、即座に反応するように設計されている。ユーザビリティの観点から、ユーザの入力に対しては 1 秒以下で起動するべきであり[1]、Linux を用いた場合にも、応答時間に対する要求を満たす必要がある。

昨年、我々はこの点に着目し、GUI の一候補である X サーバベースでの GUI アプリケーション（以下 AP）起動性能測定を行った[2]。この測定結果では、GUI AP 起動までに 2.3 秒程度掛かり、目標性能である 1 秒以下での起動を満足できず、1 秒以下での起動を行う場合は OS/GUI とも改善する必要性を報告した。今回、我々はもう 1 つの GUI 候補である DirectFB について前回検討した改善案を実施し、これを GUI AP 全般に適用した場合について調査した。本稿では、その測定結果と調査内容を述べる。

2.　GUI AP 起動時間測定

今回使用した H/W は SH ベース CPU(240MHz)、Linux カーネルは 2.4.20 ベースである。測定用の AP は前回と同様、メニュー画面処理 AP とした。本 AP は、19 個のビットマップ（合計 45KB）を表示し、ユーザからの入力を受け付ける GUI AP である。

本メニューAP は、AP 起動マネージャによって起動される。AP 起動マネージャは fork/execve を行い、メニューAP を起動する。カーネルも前回同様、システムコール開始/終了およびプロセススイッチのトレースが可能である。このカーネルを用いて、測定用のメニューAP 内に目印となるシステムコールを発行し、AP の動作を追跡している。ただし、今回の測定では、キー入力からメニューAP 起動までは測定していない。

図 1 に測定結果を示す。各測定項目のうち、GTK 初期化処理について、DirectFB/GTK では GTK 初期化処理内で DirectFB 本体の初期化が行われるため、より時間が掛かっている。また、ロケール処理については英語のみのサポートのため、処理時間が短い。

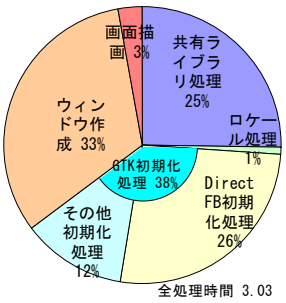

図 1　測定結果

An examination of fast application boot for consumer device
†Akio Idehara, Atsushi Settsu, Takayuki Ito, Shinichi Ochiai
Mitsubishi Electric Corporation Infomation Technology R&D Center

3.　起動処理改善

3.1.　改善策検討

前回我々が行った改善案検討「OS レベルでの改善」と「GUI ライブラリレベルでの改善」に基づき再検討を行った。「OS レベルでの改善」では、ロード処理の高速化の検討を行った。また、「GUI ライブラリレベルの改善」に関しては、GUI ライブラリが使用するデータの共有化の検討を行った。これを実現するために、GTK 初期化と、ウィンドウ作成の一部（GTK 画像関数呼び出し）を予め実行する方式について検討した。これにより期待される効果は以下の通りである。

■ GUI ライブラリレベルの改善効果（データの共有化）

メニューAP 起動前に GTK の初期化を行うことで、DirectFB 初期化の処理が実行される。メニューAP は先に初期化された DirectFB 環境を使用するため、メニューAP 実行時の DirectFB 初期化に掛かる時間の短縮が期待される。

■ OS レベルの改善効果（共有ライブラリのロード）

予め GTK 初期処理を実行することにより、DirectFB/GTK ライブラリが予めメモリ上にロードされる。これにより、2 回目の起動であるメニューAP による DirectFB/GTK ライブラリロードの時間の短縮が期待される。同様に、ウィンドウ作成処理についても、予め GTK 画像関数を呼び出すことで、DirectFB の画像ライブラリ用ラッパおよび、png 等の画像ライブラリが初期化され、メモリにロードされるため、2 回目以降の初期化およびライブラリロード時間を省く効果が期待される。

3.2.　改善後の測定結果

改善を施した結果を図 2 に示す。期待通り、GTK 初期化処理、ウィンドウ作成処理、共有ライブラリリンク処理に掛かる時間が短縮され、本 AP の起動時間は 1.14 秒程度まで減少することが確認できた。これにより、前回我々が行った検討を実証できた。

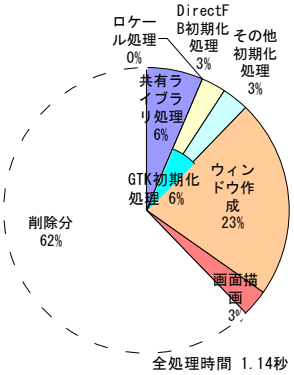

図 2　改善後の測定結果

4.　GUI AP 起動高速化の起動時間変動要因調査

今回の改善で、GUI AP 起動時間を大幅に削減することが可能となった。そこで、今回の起動高速化について、種々の画面構成でどのように変化するか調査を行った。

4.1.　画像部品数

起動処理の改善を行った場合でも、ウィンドウ作成処

理が全体の 6 割を占めている。この原因を探るため、ウィンドウ作成処理時間を調査した。調査結果を図 3 に示す。

結果から、全体の 4 割が画面部品作成であり、この中でも画像部品作成に時間が掛かっていることが判明した。今回使用した GUI AP は全部で 40 個の画面部品から構成され、この中で画像部品(image)は 19 個存在する(図 4 網掛け部分)。

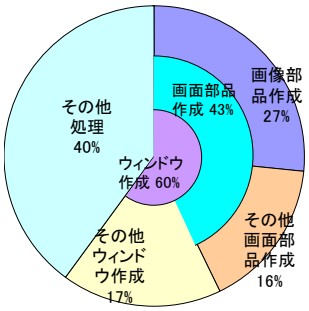

図 3　処理内訳

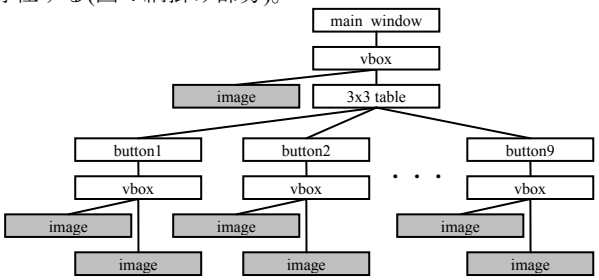

図 4　画面部品構成

画面構成を変更し、画面部品数を 12 個（うち画像部品 5 個）にしたところ起動時間は図 5 のようになった。図 3 と図 5 を比較すると、画像部品作成の時間が半分に減少した。また画面部品作成時間も画面構成変更前から 2 割減少し、全体として 3 割程度起動時間が削減されている。このことから、GUI AP 起動時間は画面部品数、特に画像部品に依存していることが分かる。

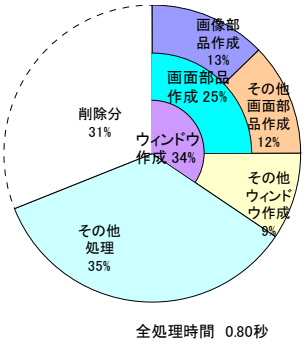

図 5　画面構成変更後

今回の場合、画像部品が 1 個増加するごとに 10 ミリ秒から 20 ミリ秒ほど起動時間が長くなる。画像部品が少ない場合はあまり問題とならないが、画像部品が 100 個程度増加した場合、起動時間は 1 秒以上延びる可能性がある。つまり、画像部品の増加により起動時間が延びるという問題があることが判明した。

ほとんどの場合、GUI AP 作成では GUI 作成ツールを使用するが、GUI 作成ツールを用いた場合、画面部品を簡単に記述できるため、不要な画像部品を作成してしまう可能性がある。これを防止するには、画面設計の方針（不要な画像は使用しない等）を作成し、これに従って AP を作成するといった工夫が必要がある。しかし、これでは自由な GUI AP 作成の妨げとなるため、抜本的な改善が必要である。

4.2.　画像形式

今回の GUI AP では、画像形式に png を使用した。起動時間に対する画像形式ごとの差を比較するため、gif、bmp、jpg を用いて再度測定を行った。png を 100%とした場合の各形式の起動時間全体、画像部品作成時間およびファイルサイズを表 1 に示す。

表 1　画像形式による比較

	gif	bmp	jpg
起動時間全体	98.7	88.9	92.2
画像部品作成時間	92.1	38.0	71.0
ファイルサイズ	83.4	123.6	30.6

図 3 と表 1 の比較から、画像形式に png を用いた場合、全体の 3 割程度を占める画像部品作成処理が、bmp の場合には 1 割程度で済む。しかし、ファイルサイズの増加によるロード時間の増加等により、bmp による起動時間全体は png の 9 割程度となる。このことから、png を用いたことによる起動時間の増加分は高々10%程度であるため、画像形式が起動時間に与える影響は小さいと考えられる。

4.3.　調査結果

以上から、今回の起動高速化を一般に適用した場合について、以下の結論となった。

1. 画像部品数の増加により、起動時間が延びるという問題がある
2. 画像形式による、起動時間への影響は小さい

1. について、現状の起動方式で更なる起動高速化を図る場合、GUI AP 作成時に、以下のような制限を設ける必要がある。

・ 不要な画面部品は作成しない。特に画像部品については必要最低限にする。

しかし、このような制限は、自由な GUI AP の作成を阻害するものである。この改善策として、画面描画まで終了したメモリイメージを保持しておき、GUI AP 起動の際にはこのメモリイメージを使用することで起動の高速化を図るといったことが考えられる。この検討は今後の課題である。

5.　おわりに

本稿では、コンシューマ向け組込み機器のアプリケーション起動の高速化検討として、DirectFB/GTK を用いた GUI AP 起動時間の測定を行い、この改善を施した。また、この改善を一般に適用した場合について調査を行った。

測定の結果、前回我々が提案した改善（OS レベルでの改善と、GUI ライブラリレベルでの改善）を行えば、起動時間を 62%削減し、1.1 秒程度で起動することを確認し、前回行った検討を実証することができた。

また、これらの改善を一般に適用した場合について調査を行い、画像部品数の増加によりに、起動時間が延びるという問題点が発見された。これの改善方法として、画面描画まで終了したメモリイメージを保持しておき、GUI AP 起動の際にはこのメモリイメージを使用するといった改善を図る必要があることを報告した。

今後は、これら問題の改善を図り、更なる起動高速化に向けた検討を行う予定である。

参考文献

[1] Miller, R. B. "Response time in man-computer conversational transactions.", *Proc. AFIPS Fall Joint Computer Conference* Vol. 33, 267-277, 1968

[2] 攝津、伊藤、落合：「コンシューマ向け組込み機器に対する組込み Linux の評価」 FIT2004 2004.9