ZIPC+Drawrialによる 家電組込みソフトウェア開発の効率化

九州日立マクセル株式会社
新分野開発部 ソフトアドバンスGr グループ長
安部田　章

１．はじめに

家電製品の多機能化に伴い、組込みソフトウェアが大規模化、複雑化し、その信頼性と生産性の両立が困難となっています。また、市場における競争の激化に伴い開発期間の短縮が要求されており、組込みソフトウェア開発の効率化が急務となっています。

ソフトウェア開発の効率化というとライブラリやコンポーネントなどコードの再利用ばかりにとらわれてしまいがちです。しかしながら、開発全体の効率化を考えると、コードの再利用のみならず仕様から設計／実装を含めた体系的な再利用を実現することが望まれます。この場合、仕様をベースに関連する設計／実装を再利用する『仕様ベースの再利用』が不可欠となります。

また、短納期化の観点からは、仕様決定の遅延が大きな障害になっていることは周知のことと思います。弊社においても例外ではなく、OEM先との『仕様決定の遅延』が、短納期化を実現するために最も大きな課題として挙がっています。

そこで本稿では、以下の２点についてZIPC＋Drawrialによる弊社の取組みをご紹介します。

（１）仕様の早期決定
（２）仕様ベースの再利用

まず、（１）に関しては、マッサージチェアのリモコン部のヒューマンインターフェース（HMI）設計にDrawrialを導入することで早期試作開発を行い、OEM先とのHMI仕様の早期決定に取り組んだ内容について報告します。また、マッサージチェアの状態遷移設計にZIPCを導入することで、Drawrialで作成したリモコンの試作画面を操作して、ZIPCで作成したマッサージチェアの状態遷移仕様を動作させることにより、HMI仕様との連携も含めたマッサージチェアの状態遷移『仕様の早期検証』に取り組んだ内容もご紹介します。

（２）に関しては、Drawrial のHMI仕様の再利用や、ZIPCによる状態遷移設計を適用することにより、『仕様ベースの再利用』に取り組んだ内容について報告します。また、マッサージチェアのメカ駆動部のソフトウェアにZIPCを適用することにより、マッサージチェア全体のモデルベース開発への取組みについて報告します。

２．家電組込みソフトウェアの開発プロセスと効率化の課題

家電組込みソフトウェアの開発は、それを組込む製品の開発プロセスに影響を受けます。そこでまず、製品の開発プロセスと対応付けて、開発の効率化を考える必要があります。

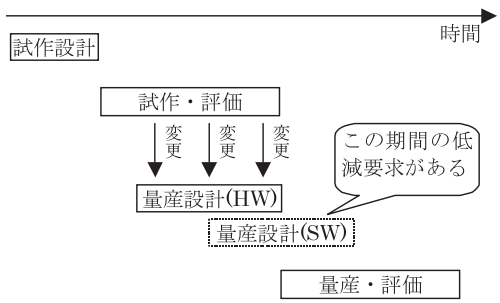

図１　製品開発プロセスとの対応付け

図１は家電製品開発の典型的な開発プロセスを示しています。図に示すようにまず、「試作設計」で試作時の仕様策定、試作設計を行います。次に「試作・評価」では試作実装を行いその評価を行います。ここで明らかになった問題点を試作で動作確認するとともに「量産設計」に反映させます。

ハードウェアの設計は試作・評価の終了とほぼ同時に終了できますが、ソフトウェアの設計

はハードウェアの仕様が確定してからでないと最終的な設計は出来ないことが多いため、どうしてもハードウェアの量産設計から遅れることとなります。この遅れの低減要求の圧力がソフトウェア開発者に大きくのしかかってきています。この要求に応えるためには、「試作・評価」から「量産設計（SW）」への仕様変更の要求を低減するとともに、仕様変更の発生しそうな個所をあらかじめ予測しソフトウェアを変更容易な構造にしておく必要があります。すなわち以下の２点に取り組むことが必要となります。
（ア）仕様変更の発生を少なくする
（イ）変更への対応を容易にする

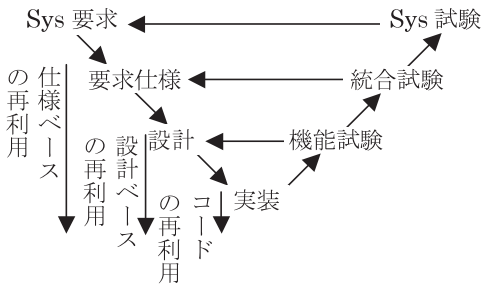

図２　ソフトウェアのV字型の開発プロセス

図２は一般的な組込みソフトウェアのV字開発プロセスを示しています。V字の右辺から左辺への矢印に示すように、要求仕様の検証は、ソフトウェアの統合試験によって行います。ここで誤解があるのは、この段階でソフトウェアの要求仕様が妥当なものかどうかを検証するわけではありません。この段階では要求仕様に瑕疵はないことを前提として、仕様どおりに動作するかどうかを検証するものです。すなわち、この段階で要求仕様の妥当性に問題が見つかれば、大きな手戻りを起こしプロジェクトの遅延を引き起こします。したがって、ソフトウェア開発の効率化を考えたとき、『仕様の早期決定』と、要求仕様の決定時における『仕様の早期検証』を可能な限り行っておくことが非常に重要となります。

また、V字開発プロセスを見ると明らかなように、実装ベースで再利用するよりも設計ベースで再利用するほうが、設計ベースで再利用するよりも要求仕様ベースで再利用する方が、再利用の効率は良くなります。すなわち仕様ベースで関連する設計・実装を再利用することが出来れば、ソフトウェアの開発効率は格段に向上します。

以下では、『仕様の早期決定』と『仕様の早期検証』、および『仕様ベースの再利用』への取組みについて説明します。

３．DrawrialによるHMI設計と仕様の早期決定

マッサージチェアでは、2005年にリモコン部に、LCDの画面サイズ拡大と十字キーの導入といった大きな仕様変更と機能追加が発生しました。この画面設計期間の短縮と仕様決定の早期化を計るため、Drawrialを導入しました。また、2006年にはマッサージチェアのマイナーチェンジが行われ、DrawrialによるHMI設計の仕様ベースの再利用を行いました。

3.1．リモコンHMI設計期間の短縮

リモコンの仕様変更と機能追加に伴い、画面設計はまったくの新規開発となったが、Drawrialを導入する事により、画面設計と画像部品の管理が容易になりました。

例えば、Drawrialの「画面レイアウト設計機能」により、複数の画面にレイアウトする画像部品を「リンク」によって一元管理できるので、画像部品に変更があった場合、元の画像部品のみを変更すればリンク先のすべての画面に反映されます。これにより画像部品の一元管理が可能になり、画面変更時の作業を効率化することが出来ました。

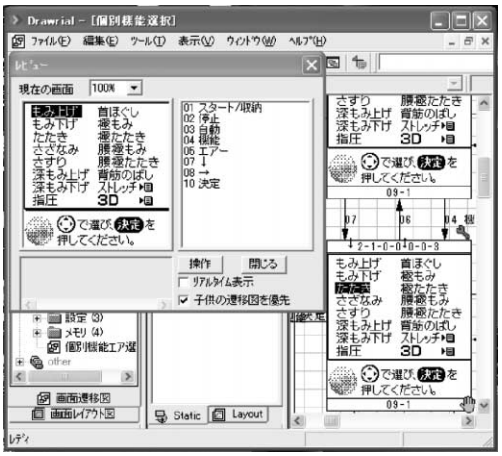

図３　レビュー機能による画面設計の様子

また、図３に示すように、Drawrialの「画面遷移設計機能」により、画面遷移仕様の設計と遷移図などの画面仕様書の作成が容易になりました。さらに、Drawrialの持つレビュー機能により、リモコンのデバッグボードなど、実機の作成を待たずに動作確認することが出来るようになり、画面設計をハードウェアの設計や試作によらず前倒しで行うことが出来るようになりました。

３.２．シミュレータによる早期仕様決定

弊社ではOEM先との製品仕様の擦り合わせに時間を要し、ソフトウェアの仕様決定が遅延するという問題がありました。また、十字キーなどの操作と連動した画面遷移仕様の実機による確認が困難なため、仕様の未決定部分を残したまま、試作設計・実装に入らざるを得ず、試作設計・実装中に仕様が確定したり、仕様変更が頻発したりすることも珍しくありませんでした。リモコンの仕様変更と機能追加の要求仕様を早期に決定するため、Drawrialのレビュー機能と連携したモックアップ（バーチャル・ターゲット）を作成し、リモコンの操作／表示のシミュレーションを実現しました。これにより、実機に近い形でボタン操作と画面遷移のデモンストレーションを行えるようになり、OEM先との仕様決定を早期に行うことが出来るようになりました。これにより、仕様決定の質も向上し、仕様決定後の仕様変更も少なくなりました。

図４　バーチャル・ターゲットを用いた リモコン画面操作のシミュレーション

３.３．リモコンHMI検証の早期実現

以上のようにDrawrialを導入することにより、HMI設計を効率化することができるとともに、実機の作成を待たずに、実機に近い環境でHMI動作検証を行うことが出来るようになり、試作・評価時に仕様変更などの手戻りの発生を低減することが出来ました。

３.４．DrawrialによるHMIの仕様ベースの再利用

2006年のマッサージチェアでは、リモコンに大きな仕様変更や機能追加がありませんでした。そのため、Drawrialで作成した画像部品や画面レイアウト、画面遷移などの設計情報をほとんどそのまま再利用することが出来ました。

画面の詳細な変更に関しては、画像部品を変更することにより、関連するすべての画面レイアウト図に反映されるため、変更のコストを最小限に抑えることが出来ました。

４．ZIPCによる状態遷移設計と仕様の早期検証

マッサージチェアのリモコンのソフトウェア設計において、ZIPCを導入して状態遷移設計を効率化しました。また、DrawrialとZIPCとの連携により、HMI設計と状態遷移設計の動作検証を早期に行うことができました。

４.１．状態遷移設計の早期検証の実現

ZIPCで作成したマッサージチェアの状態遷移設計の動作確認をZIPCのシミュレーション機能により行い、状態遷移設計とその検証を同時進行で行うことができました。これにより、マッサージチェアの状態遷移仕様は確認できますが、リモコンの操作や画面表示と連携してマッサージチェアの状態遷移が適切に行われているかどうかを検証する必要があります。

そこでZIPCのVIP機能を利用して、Drawrialで作成したHMIを連携させたZIPCの状態遷移シミュレーションを実現することにより、マッサージチェアの状態遷移仕様の早期検証を行うことができるようになりました。図５に、Drawrialのレビュー機能によりリモコンのバーチャル・ターゲットを動作させ、リモコンの操作によりZIPCの状態遷移が適切に行われているかどうか確認している様子を示しています。

図５　DrawrialとZIPC連携による状態遷移設計検証の様子

４.２．エミュレーション機能によるモデルベース検証

通常、ZIPCで作成した状態遷移仕様は、ジェネレーション機能によりC言語などの実装コードに変換して、各々のマイコンの開発環境でコンパイル、デバッグを行います。この場合、マイコンの開発環境でバグを発見すると実装コードで修正を行い、それをZIPC上の状態遷移モデルに反映させるといった、２段階の修正を行わなければなりません。

ここで、ZIPCで作成した状態遷移仕様を、ZIPC上で状態遷移モデルとして動作デバッグを行うことが出来れば実装コードを意識せず、モデルベースによる検証が実現します。この場合、ZIPC上のモデルのみでバグの検出・修正が完結します。

そこで図６に示すように、ZIPCのエミュレーション機能を利用して、ZIPCで作成したマッサージチェアの状態遷移仕様のモデルによるデバッグを行い、ZIPC上の状態遷移仕様のモデルベース検証を実現しました。

さらに図６に示すように、リモコンのバーチャル・ターゲット上での操作を、VIP経由でZIPCに引渡しリモコンの操作によるデバッグも試みました。

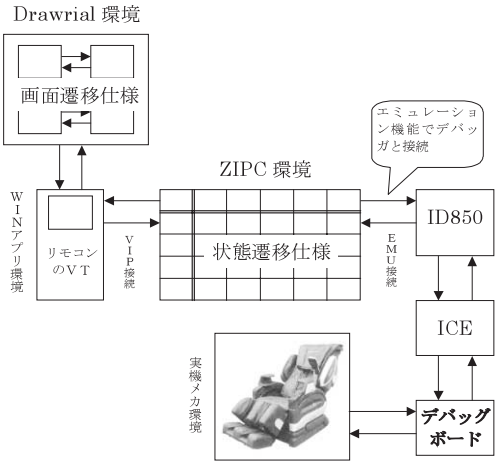

図６　エミュレーション環境

今後のモデルベース開発、検証へ発展させる上で大変有意義な取組みになりました。

５．フレームワーク構築とモデルベース開発への取組み

２章で述べたように、ソフトウェア開発者には、ソフトウェア量産設計の終了の、ハードウェア量産設計の終了からの遅延を少なくする要求があり、それを実現するためには、仕様変更の発生を少なくするとともに、変更時のコストを低減する必要があります。

３章および４章で、Drawrialによる仕様の早期決定と実機に近い環境での仕様検証に取り組むことで、（ア）仕様変更の発生を少なくする、ことができました。（イ）に関してもHMIの仕様ベースの再利用を実現するとともに、ZIPCのモデルベース検証を試行する上で、仕様変更に強い仕様ベースの再利用の実現に一歩近づくことができました。以上の取組みによって、弊社における家電組込みソフトウェア開発の効率化がある程度実現できたものと考えています。

本章では、DrawrialやZIPCを初めとした開発ツールによるモデルベース開発と、仕様ベースの再利用の実現で、（イ）変更への対応を容易にするためのさらなる取組みについて述べます。

５.１．開発フレームワークの構築

仕様ベースの体系的な再利用を実現するためには、再利用資産の開発を行うための枠組みであるフレームワークの構築が不可欠です。フレ

ームワークの決定により、その上で動作する再利用資産の設計ルールが決定し部品の標準化が図られます。その設計ルールにのっとって、フレームワーク上で動作するコンポーネントやドライバ、ライブラリなどの再利用資産を開発していきます。

5.2. フレームワークの要求仕様

弊社のマッサージチェア開発に適合するフレームワークに要求される仕様を分析します。まず、マッサージチェアの性能要求とそれに基づくアーキテクチャを規定します。ここでアーキテクチャの記述方法として、一般的によく使用される論理、実行、開発、配置の４つのビューで記述することとします。

マッサージチェアでは、マッサージ動作を実現するメカ機構を一定時間内に制御するという性能要求があります。そのため、メカ機構の時間制御優先のアーキテクチャとして周期駆動型アーキテクチャを採用しました。RTOSは導入しておらず、複数の機能の並行処理を行うため、擬似マルチタスク機構を導入しています。

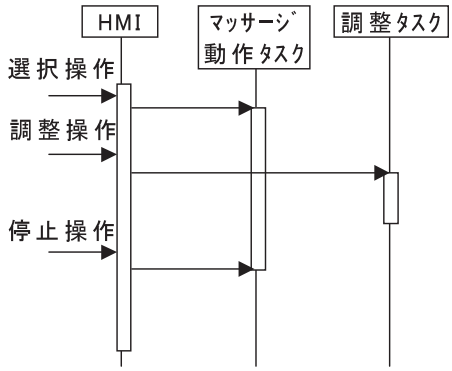

図7　フレームワークの実行ビュー

図７にマッサージチェアの実行ビューの例を示します。一定周期でHMIタスクがユーザ操作を監視します。HMIタスクがユーザのマッサージの選択操作を検出するとマッサージ動作タスクを起動します。マッサージ動作タスクはアクティブとなり一定周期でマッサージ動作処理を行います。マッサージ動作タスクがアクティブの状態で、HMIタスクがユーザの調整操作を検出すると調整タスクを起動し、マッサージ調整処理を行います。このようにしてHMI、マッサージ動作および調整タスクが並行に処理を行います。

次にマッサージチェアにおける変化要求を分析しますと、以下の３点となります。

（１）マイコン間の移植
（２）ハードウェア間の移植
（３）機能の選択による変更要求

まず、（１）については、マイコン価格低減のためマイコンの機種変更要求、仕様変更によるマイコンパフォーマンス変更要求が挙げられます。したがって、マイコンを容易に移植可能なようにフレームワークを設計する必要があります。

（２）については、仕様変更やコスト低減の要求からメカや回路のハードウェア部品の変更要求に対応する必要があります。したがって、モータやセンサなどのハードウェアを容易に変更できるような構造にする必要があります。

また、（３）に関しては、仕様変更やラインアップにおける機能変化点に対応できる構造への要求がありました。図８にマッサージチェア全体の概念的な論理ビューを示しました。

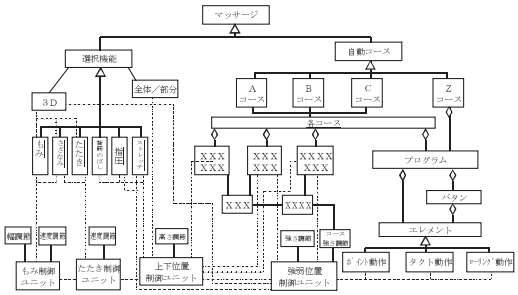

図８　マッサージチェア全体の概念的な論理ビュー

マッサージチェアではマッサージ機能として選択機能とコースの２つに大きく分けられます。図の論理ビューの分析から、マッサージチェアの選択機能では「もみ上げ」「たたき」「さざなみ」「ストレッチ」などといったマッサージ機能の追加、変更、削除要求が挙がりました。また、調整機能として高さ調節、強さ調節、幅調節、速度調節、部分／全体動作があり、各々のマッサージ機能別に、調整可能な機能を選択できる要求がありました。

また、コース数や内容を容易に変更できるように、マッサージ動作を「ポイント」、「タクト」、「ローリング」の３つの基本動作『エレメント』の組合せで『パタン』や『プログラム』を容易に構成できるようなアーキテクチャにしました。これによりコース内容の動的な変更も可能となります。図９にコースのプログラム仕様に関するアーキテクチャの詳細を示します。

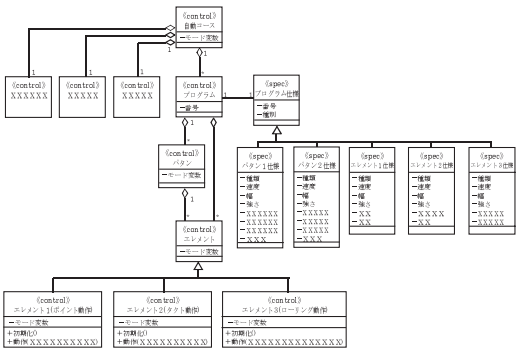

図９　コースの論理ビュー

このような要求に対応するフレームワークとして図10のような開発ビューを設計しました。図中の状態遷移層の設計にZIPCを導入しマッサージ機能に対応する状態の追加、削除、変更が可能になるようにし、状態遷移層から各々のマッサージの機能タスクを呼び出す構造にしました。

図10　フレームワークの開発ビュー

以上のフレームワークの設計によって、（１）の「マイコン間の移植」をデバイスファイルの交換により容易に行えるようにしました。また（２）の「ハードウェアの変更」をその制御コンポーネントの交換あるいは変更により容易に行えるようにし、他の部位に影響しない構造にしました。最後に（３）機能追加要求としては、選択機能のマッサージ動作、およびコースの追加、変更、削除はZIPCによるモデルベース開発により容易に対応できるようにしました。さらに、コースのプログラム仕様を容易に変更可能な構造にしました。

６．まとめ

本稿では、家電組込みソフトウェア開発を効率化するため、DrawrialとZIPCを導入することによる、『仕様の早期決定』と、『仕様ベースの再利用』に取り組んだ内容についてご紹介しました。また、DrawrialによるHMI設計とZIPCによる状態遷移設計によりマッサージチェアのモデルベース開発への取組みに関してもご報告しました。

しかしながら現在はZIPCによるソフトウェア仕様のモデルベース開発は状態遷移層に限定されています。今後は、ZIPCによるモデルベース開発を機能タスク層にまで拡げる取組みを進めていこうと考えています。

また、ZIPCの状態仕様と設計／実装のトレーサビリティを実現し、仕様ベースの再利用を実現したいと考えてます。仕様のトレーサビリティの実現には要求管理ツールの導入を進めており、さらなる仕様ベース開発を推進していこうと考えてます。