ソフトウェア工学 140－16
（２００３．３．７）

多層化による組込みソフトウェアの移植性の向上

安部田　章 †

本稿では，家電組込みマイコンのソフトウェアの移植性や再利用性を向上させるソフトウェア構成法としてコンポーネントベースの多層アーキテクチャを提案する．まず，実際の開発現場における移植や再利用の形態を分析し，ソフトウェアに必要とされる要件を示す．次に，本方式が各々のケースで移植や再利用を効果的に実現できることを示し，その実装方法を示す．最後に，本アーキテクチャに基づいたコンポーネントの設計例を紹介する．

The Multiple Layer Architecture with High Portability in Embedded Software

AKIRA ABETA

This paper proposes the multiple layer architecture that improves the portability of embedded software on electric household appliances. We analyze case of the portability or reuse in actual development. We show that software based on our architecture has high portability in each case. This paper reports about an example of design of software based on it.

1. はじめに

近年，家電製品が高度化し組込みソフトウェアの規模が増大する [1) 3)]一方で，市場の要求に即応することが求められており，開発の効率化と信頼性確保が急務となっている．そこで組込みソフト開発において，従来のパフォーマンス重視の開発を改め，再利用性や保守性に優れたオブジェクト指向開発が注目されている．

オブジェクト指向を導入すると部品性が向上し再利用が容易となることが期待される．しかし，一言で移植や再利用といっても，ケースバイケースでありその形態も様々である．また，各々のケースでソフトウェアに求められる要件も異なってくるものと考えられる．

本稿では，まず，家電組込みソフトウェアの開発現場における移植や再利用のケース分析を行い，ソフトウェアに必要とされる要件を整理する．次に，各々のケースで効果的に再利用が行えるソフトウェア構成法として多層アーキテクチャを示し，その実践例を紹介する．

2. 組込みソフトウェア開発における移植や再利用のケース分析

2.1. マイコン間の移植

マイコンベンダー毎に納期対応に差が生じる，また，同じベンダーでも時期，あるいはマイコン機種によって，その納期対応にばらつきがある．納期厳守するため，マイコンの機種変更や他のベンダーへ転注するなどの対応を迫られる場合がある．また，突発的な仕様変更からマイコン選定を見直す必要が生じることもある．したがって，マイコンのソフトウェアは，マイコン HD に依存する部分を他の部分から容易に分離でき，交換可能なように構成する必要がある．

また，移植が容易になれば，マイコン相見積りが可能となり，マイコン価格交渉力が高まる．

2.2. ハードウェア間の移植

家電製品の場合，原価低減のため，ハードウェア部品を低コスト品に変更することがよく行われる．これは製品の VEC活動の一貫として，製品出荷開始後も継続して検討される．また，要求仕様が変更され，使用するハードウェア部品およびその制御方法が変更される場合も多い．したがって，ソフトウェアはハードウェア部品に固有の制御構造に依存した部分が容易に分離でき交換可能なように構成され，ハードウェア変更時の移植作業が容易に行える必要がある．

†九州日立マクセル株式会社 新分野開発 PT, 福岡県田川郡方城町大字伊方 4680, abeta@kyuma.co.jp

2.3. 同種製品間の再利用

家電製品ではラインアップやモデルチェンジなどを行うため，同種製品間で共通に利用可能な部分をできる限り共用，あるいは再利用し，製品展開を効率化する必要がある．ラインアップでは，基本アーキテクチャを共通化して制作採用すると共に，共通に利用できる機能部品を制作し，製品ごとに異なる部分を組み入れてカスタマイズしていく．モデルチェンジでは，その対象となる現行のソフトウェアを再利用ベースとして，機能削除，変更および追加の作業を進める．

したがって，共通に利用できる範囲を最大化するように，製品固有の部分を局所化して分離できる，追加機能部品を容易に組み込めるように，ソフトウェアが構成されることが望まれる．

2.4. 異種製品間の再利用

同一企業内で生産する製品では，ハードウェア部品の信頼性保守や購買効率化のため,異種の製品間でも，使用するハードウェアを可能な限り共用化する．したがって，それらの制御ソフトは，異種製品間を容易に再利用できるように部品化されていなければならない．

また，異種製品を開発する場合でも，すべてを一から作っていくことは稀であり，常は過去に作成した自社のソフトウェアで最も類似したものを選び出し，それを基に作成することになる．この場合，現状ではハードウェア制御部分や機能部分で部品として再利用できるようになっておらず，数値演算などの汎用サブルーチンなどに限定されているのが現状である．

ここで，基本アーキテクチャが製品によらず共通化され，その上で実行可能なソフトウェアが部品単位で分離でき，組合せ可能に作られていれば，ソフトウェアの再利用性が向上し異種製品の開発も大きく効率化できるはずである．

2.5. 移植性や再利用性向上のための要件

以上の分析から，家電組込みソフトウェアに求められる要件として,①OS を含めた基本アーキテクチャの分離，共用化，②マイコン間の移植を容易にするためのマイコン HD ドライバの整備（パッケージ化，呼び出しインターフェースの標準化），③機構および回路などハードウェア部品の制御ソフトの部品化（コンポーネント化），④アプリケーション（製品）の機能単位の部品化があげられる．

3. 移植性を考慮した多層アーキテクチャ

以上の要件を満たすソフトウェア構成法として，多層アーキテクチャ [5]を提案する．

3.1. 3層アーキテクチャ

組込みソフトウェアの移植性を向上させる多層アーキテクチャでは，まず，ソフトウェアの構成を①マイコンハードウェアドライバ層，②メカニズム層，③アプリケーション層の３つの階層に分けて構築する．

①は，マイコンのポート配置やアクティブレベル，通信や周期駆動用タイマなど，マイコンのハードウェアを抽象化する．この部分には uITRON などの OS を使用する場合もある．②は制御対象のハードウェアの制御メカニズムを抽象化する．③はメカニズム層のソフトウェア部品を使用してアプリケーションを実現する．

マイコン内蔵のハードウェアを抽象化するため，例えば，周期駆動タイマの時間設定および起動，プログラマブルパルスジェネレータの周波数設定および出力制御などの利用インターフェースを標準化する．これにより，マイコンを移植する場合，①マイコン HD ドライバをマイコン機種毎に準備しておき，交換するだけで容易に移植が行えるようになる．

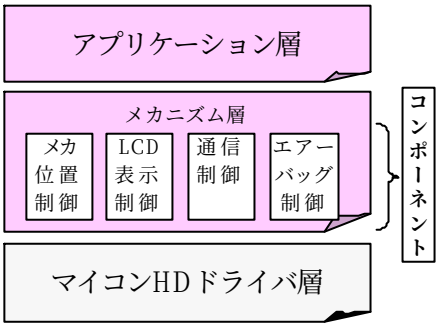

図 1 多層アーキテクチャ

マイコンのハードウェアに依存する部分と，機構および回路の制御メカニズムに依存している部分を別レイヤとすることにより，マイコン機種変更時は，マイコン HD ドライバ部分のみを交換すればよい上に，機構や回路部品の変更時にも，マイコン HD ドライバ部分の修正がなく，対象となる機構部品に対応するメカニズム層のみを交換すればよい．これにより，マイコン機種変更，ハードウェア機種変更時による影響の範囲を局在化させ，移植作業の効率化と信頼性の向上を実現することができる．また，アプリケーション層が

分離されているため，これらの変更時には，アプリケーション層の変更は基本的には生じない．

3.2. メカニズム層のコンポーネント化

組込み制御ソフトウェアでは，前述したメカニズム層の構築が大きなウエイトを占める．しかしながら，この部分は当然対象ハードウェアの制御メカニズムに依存するため，ハードウェア部品の機種や構成の変更により移植作業が発生する．このため，再利用が最も進んでこなかった部分であるといえる[7]．したがって，この部分の移植性を向上させることが組込みソフトウェア全体の移植性の向上をすすめる上で非常に重要なこととなる．そこで我々は，メカニズム層を制御対象ごとにコンポーネント化して構築する方式を導入する．

これにより，ハードウェア部品の交換時に，その影響範囲をコンポーネント内に局在化できるので，メカニズム層の移植性が向上する．

3.3. メカニズム層の多層化による移植性の向上

しかしながら，組込み制御ソフトウェアでは，このメカニズムコンポーネントが複雑化し大規模化する傾向がある．このような粒度の粗いコンポーネントによる再利用では，再利用の単位が非常に粒度の粗いものとなってしまうという問題がある．例えば，モータの回転を制御することによって機構メカ位置を制御するコンポーネントにおいて，制御対象のモータの機種が変更された場合，モータの回転を制御する部分だけでなく，特定のモータ機種に依存しないメカの位置制御処理の部分も含めて，コンポーネント全体が移植対象となってしまう．加えて，メカニズム層のコンポーネントが製品機能に依存する場合，異種製品への移植も困難なものになる．

そこで我々は，図 1に示すように，メカニズム層のコンポーネントをさらに，ア）ハードウェア依存層，イ）アプリケーションおよびハードウェア非依存層，ウ）アプリケーション依存層，に分割構築するメカニズム層の多層化を提案する．

ア）の例としては，LEDダイナミック点灯やチャタリング防止付入力検出，モータ PWM 制御など具体的なハードウェアを隠蔽する抽象度の低いメカニズムを提供する．イ）は，ア）を利用して具体的なハードウェアの制御に非依存な抽象度の高いハードウェアメカニズムを提供し，アプリケーションの具体的な仕様にも非依存で利用できる．例としては，LCD文字表示や7セグ LED数字表示，モータ位置制御などが挙げられる．ウ）は，ア)やイ）を使用してアプリケーションの具体的な仕様を実現するメカニズムである．

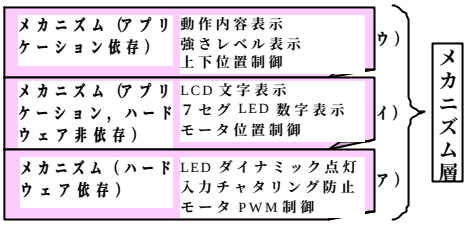

図 2 メカニズムの多層化

メカニズム層を上記のように３層構造に分割構築することにより，制御対象のハードウェアが変更された場合でも，ア）を入れ替えるだけでイ）ウ）層のプログラムを再利用可能にした．また，ア）イ)層は，アプリケーションに非依存であるため，他のアプリケーションでもそのまま利用可能となる．これにより，従来困難であったメカニズム層の部品性が向上し，ハードウェア間の移植や製品間の再利用が容易となった．

3.4. コンポーネントの多層化

メカニズム部分の多層化は，図 3に示すように，コンポーネントの多層化により実現する．コンポーネントの多層化により，制御メカニズムを各々のケースで効果的に再利用できるようになる．例えば，モータの回転制御によるメカ位置制御コンポーネントは，図に示すようにア）モータ PWM 制御，イ）モータ位置制御，ウ）動作位置制御と多層化して構築することにより，３つの抽象レベルで再利用できるようにする．

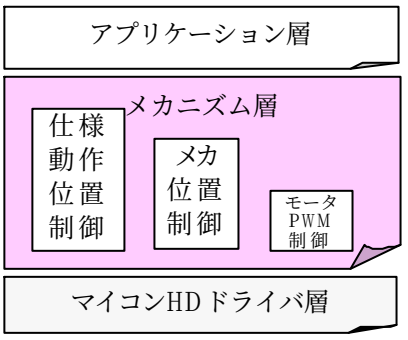

図 3 メカニズムの多層化

例えば，マイナーチェンジなどでは、製品仕様に合わせた位置制御といったアプリケーション依存層の高

い機能レベルでコンポーネントを再利用できる.また，フルモデルチェンジでは，メカ位置制御といったＡＰおよびHW非依存層の抽象レベルで再利用可能であることが期待できる．そして，異種製品においても少なくとも HW 依存層といった低い抽象レベルでは，コンポーネントを再利用可能になることが期待できる.

4. 実装メカニズム

4.1. 基本アーキテクチャと通常タスク

組込み制御システムにおける処理の多くは，周期的に実行する周期駆動タスクとして実装される．この理由として，組込み制御システムにおけるディジタル制御が一定サンプル周期による処理を基本としていること，反応時間の見積りがしやすく時間保証が容易なこと，などが挙げられている [4]．そこで本方式における基本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ている．したがって，本アーキテクチャにおけるコンポーネントの実装は，一定の時間で周期的に駆動される周期駆動タスクとして動作することを前提としている．この周期駆動タスクを通常タスクと呼ぶことにする．この通常タスクとして実装されるコンポーネントは，uITRON などの RTOSにおいても，RTOS 上に周期駆動タスクとして実装可能であり，容易に移植可能である.

4.2. 通常タスクと割込み処理

コンポーネントは基本的に，通常タスクとして実装することにより，再利用性に優れた部品化が可能となる．しかしながら，組込み制御システムでは，実行速度等の制限などから，外部割込みやタイマ割込みを利用した処理プログラムが必要となる場合も生じる．例えば，比較的高速な出力信号のタイミング作成処理などでは，割込み処理が必須となる．このような場合，割込み処理内で必要となる最低限の信号出力処理を記述し，それ以外の例えば信号パラメータ設定等の部分は，通常タスクとして構築し，通常タスクと割込みタスクを一つのコンポーネントとして実装する.

例えば，図 4に示すように，通常タスクと 2 つのタイマ割込み処理を一つのコンポーネントとして作成する．このコンポーネントを再利用する場合は，通常タスクを基本アーキテクチャに組込み，割込みタスクを割込みリソースに割り当てることによって行う.

家電組込み分野では，ヒューマンインターフェースの処理や時刻・時間計算，表示制御など多くの入出力処理が基本アーキテクチャで処理可能である．それ以外の割込みを利用する処理は，通常タスクと割込みタスクの組合せにより実現できる．このような例としては，信号出力処理などにおけるパルス列の生成などがあげられる．また，通信用のドライバのように，汎用的な信号出力処理は，インターフェースを規定してドライバ化する．これを通常タスクから呼び出すことによりドライバを利用する.

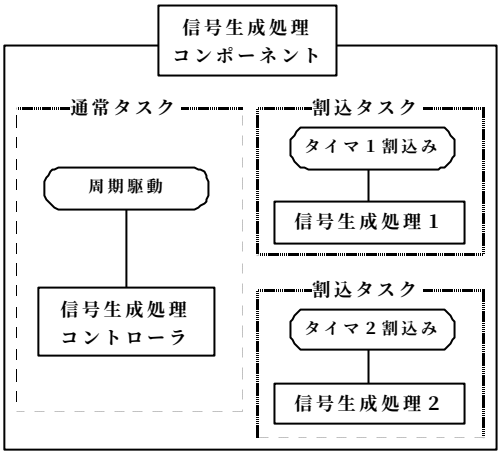

図 4　割込みを含むコンポーネント

4.3. コンポーネントの内部構造

コンポーネントの内部構造は，図 5に示すように，まず、コンポーネントの名称を記述しインターフェースを記述する。コンポーネントを利用する場合、このインターフェースを呼び出す.

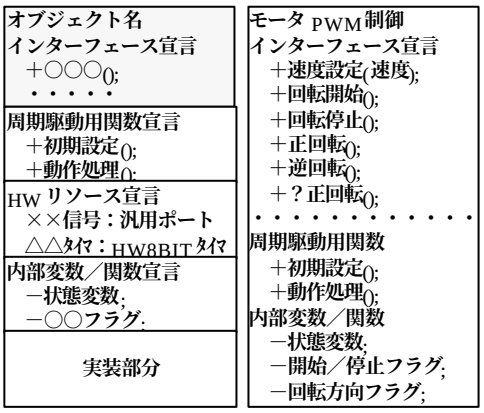

図 5　コンポーネントの内部構造

次にこのコンポーネントを動作させるための周期駆動用の関数を定義し，このコンポーネントで利用する

マイコンリソースを宣言している．この部分は開発ツールが，手順にしたがって基本アーキテクチャに周期駆動用関数を組み込み，リソースの割り当てを行うことによって自動化することができる．最後に，このコンポーネントの内部で利用する変数を宣言し，実装部分を記述する．

図 6に多層コンポーネントの内部構造を示す．図に示すように，コンポーネントが他のオブジェクトを必要とする場合、そのオブジェクトをインクルードし、周期駆動用の初期設定と動作処理を呼び出す.例えば、モータ制御コンポーネントは、初期設定でブレーキ用タイマの初期設定を呼び出し、動作処理でブレーキ用タイマの動作処理を呼び出すことによって実現する．また、ＡＰ，HW非依存層のコンポーネントであるメカ位置制御オブジェクトは、下位層のモータＰWM制御オブジェクトをインクルードし、周期駆動用の初期設定や動作処理を階層的に呼び出だすことで実現する．

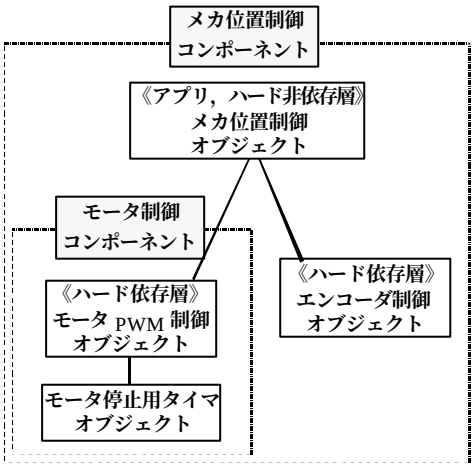

図 6 多層コンポーネントの内部構造

5. 実践例

メカニズム層を多層化した設計例として，モータによるメカ位置制御を UML 記述したものを図2 に示す．図のように，ハードウェア依存層はモータの具体的な制御方法(PWM 制御)を隠蔽し，モータ制御インターフェースを提供している．中間層は、最下位層のモータＰWM制御のインターフェースを利用してモータを動作させ，エンコーダのクロック信号を検出することによりメカの位置をカウンタ値で制御するメカ位置制御ユニットを実現している。そして，最上位層はメカ位置制御ユニットを利用して，具体的な上下位置にメカを移動させるアプリケーションに依存したメカニズムを提供している．

このように構成することにより，モータの制御方法が変更された場合でも，ハードウェア依存層以外の部分は再利用可能となった．また，中間層以下のメカ位置制御ユニットは，具体的なアプリケーションとは非依存であり, 他のアプリケーションに移植可能である．

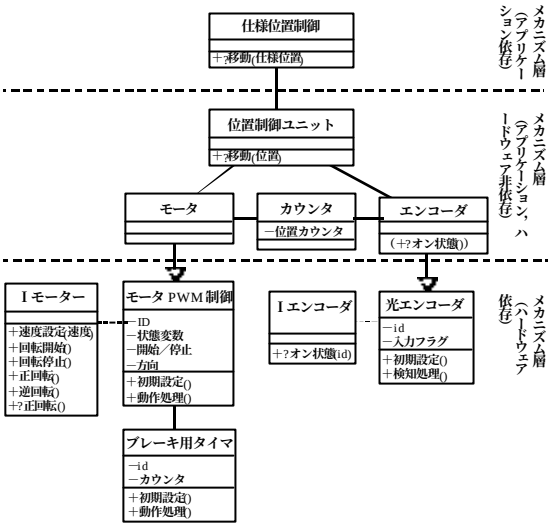

図 7 メカニズム層の多層化の実例

6. おわりに

弊社では,マイコン転注の自在化, 相見積りの実現, および短納期実現のために，組込みマイコンにおける RAD 開発システムの実現をすすめている．移植や再利用向上への取り組みはそのベース技術の一つに位置付けられる．この開発システムは，パソコン上で制御対象のハードウェアの挙動をシミュレートし，パソコン上で家電組込みソフトを開発，デバッグできるシステムである．このシステムでは，RADフレームワークにより，手軽にコンパイル＋実行できる高速試作環境を提供している．また，パソコン上で作成デバッグしたプログラムは，そのままマイコン用にコンパイルできる．

本システムでは，図 1に示したマイコン HD ドライバ層をマイコン機種ごとに準備しており，マイコンの移植を容易にしている．また，メカニズム層をハードウェアごとにファイル単位でフレームワークに取り込んで利用可能である．今後はこの部分をコンポーネントとして登録・利用できるようにしたい.

今後の課題としては，アプリケーション層の部品化による移植性の向上を進めることが挙げられる．アプ

リケーション層は，メカニズム層のコンポーネントを利用して製品の具体的な機能を実現する部分である．この部分を機能単位にコンポーネント化を図る方法がまず考えられる．しかしながら，製品の機能単位での再利用の場合でも，効果的な再利用を実現するためには，メカニズム部分と同様に粒度の大きさが問題となる．2)で河野らが論じているように，組込み制御ソフトは，状態構造をもつ複数の入出力変換特性があるという特長がある．したがって，状態遷移マシン(FSM)としての部品化およびモード拡張性（モードの追加，削除など）を実現することが重要であると考えている．

開発体制についてであるが，我々のような健康機器家電分野では，従来は一人一製品の開発を担当することも珍しいことではなかった．しかし近年では製品の高機能化が進み，もはや一人で一製品のすべてを開発することが困難になってきている．開発サイクルの短縮が急務となっている中では，人を投入してでも製品を早期に立ち上げ製品投入タイミングを逃してはならない．そこで，プロダクトラインの考え方である，ドメインエンジニアリング(再利用の対象となるコア資産の開発)とアプリケーションエンジニアリング(コア資産の統合による製品の開発)を分業化する考え方[6)]を家電製品開発にアレンジした開発体制を検討している．

この開発体制では，アプリケーションエンジニアリングはユーザサイドから商品を熟知し要件定義，要求分析，分析などを行い，コンポーネントを組み立ててプログラムを作成する業務を担う．これに対して，ドメインエンジニアリングでは，マイコンの知識，ハードウェアの知識，OSの知識，回路設計の知識といった組み込みシステムで必要となる実装技術を熟知し，ドメインで必要となるコンポーネントの抽出，設計および実装を行う．これらのすべての領域をカバーし，すべての面でプロフェッショナルになることは，非常に難しい．現在の組み込みソフト技術者がカバーするべき範囲が広がりすぎていると考えており，その点からもこの分業体制の中で各々のプロフェッショナル育成を進めることが人材の育成の面からも好ましいと考えている．

参 考 文 献

1) 高田広章:組込みシステム開発技術の現状と展望，情報処理論文誌，Vol.42，No.4，pp.930-938 (2001).
2) 河野善彌，陳慧，B.H.Far:組込システム事業の為のソフトウェアシステム，ソフトウェア工学研究会資料. 02-SE-139-4，pp.1-8.情報処理学会 (2002).
3) 渡辺政彦:状態遷移ベースのソフトウェア開発環境の現状と動向,計測と制御. Vol.41，No.2，pp.117-121(2002).
4) 横山孝典，納谷英光，他 5 名:組込み制御システムのための時間駆動オブジェクト指向ソフトウェア開発法，電子情報通信学会論文誌，D-Ⅰ，Vol.J84-D-Ⅰ，No.4，pp.338-349 (2001).
5) 安部田章:組込みソフトウェアの移植性を向上する多層アーキテクチャ，ウィンターワークショップイン神戸論文集，情報処理学会シンポジウムシリーズ，Vol.2003，No.5，pp.9-10 (2003).
6) 今関剛，亀田裕香，川口晃:組み込みシステム開発へのプロダクトラインの適用について,オブジェクト指向最前線 2002，OO2002 シンポジウム，pp.129-132.
7) 渡辺博之，渡辺政彦他２名:組み込み UML,翔泳社，pp.129-132.