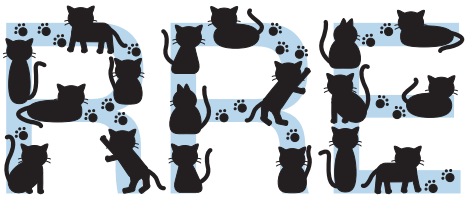

Round Robin Engineering

構造モデルと振舞モデル

Communication
Art
Technology
Systems

１．はじめに

ここ数年、車載ソフトウェアの領域ではAUTOSAR[1]という名前を聞くことが増えた．AUTOSAR は車載ソフトウェアのアーキテクチャのことであり、その標準化団体でもある．日本国内に目を向けると、TOPPERS プロジェクトから TECS（TOPPERS Embedded Component System）が発表された [2]．名前からも分かるとおり TECS は組込みシステム向けコンポーネントシステムである．どちらもソフトウェアの構造モデルに着目している．

一般にソフトウェアの仕様は、振舞モデルを表現していることが多く、構造モデルとしての仕様に言及されることは非常に少ない．「ボタンが押されたら、ライトが点灯すること」というような仕様はまさしく振舞いに関しての定義であり、その振舞がどのような構造で実装されねばならないかについては言及されていない．

本稿では、既存の構造モデルを紹介した後、構造モデルを設計することの意義と振舞モデルとの協調設計の概要を紹介する．

２．既存の構造モデル図

既存の構造モデル表現について整理する．

２.１　UML における構造モデル図

UML には１３図が存在するが、そのうち構造を記述するために以下の図を使用できる（図１）[3]．

- クラス図
- 複合構造図
- 配置図
- パッケージ図
- オブジェクト図
- コンポーネント図

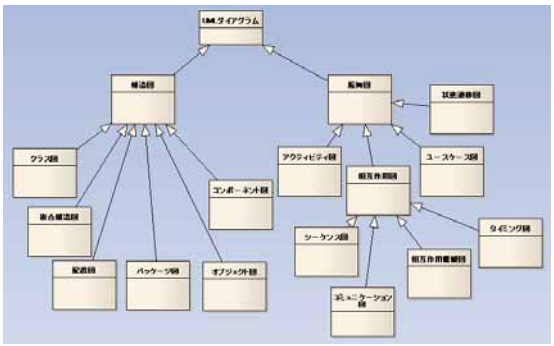

図1 UML 図の関係(Wikipedia 図を日本語化)

２.２　AUTOSAR における構造モデル図

AUTOSAR では車載ソフトウェアプラットフォームとなる大きなモデルを定義しているがここでは、ソフトウェアの観点から構造を記述しているモデルを取り上げる．

コンポジション図

AUTOSAR では、ソフトウェアコンポーネント(SWC)間の接続関係を記述する図としてコンポジション図(図 2)を定義している．コンポーネントは SWC として表現され、他のコンポーネントとのやり取りのためのポートが定義される．ポートにはやり取りの手段やデータタイプ等が定義される．

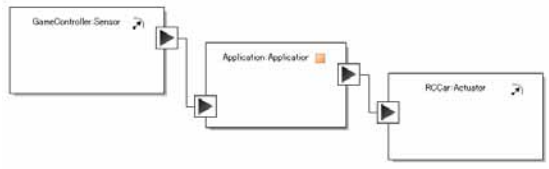

図2　コンポジション図

ランナブル図

SWC 内部の構造設計として RTOS のタスクを想定したランナブルという単位で構造を記述する図をランナブル図として定義している．ランナブルだけを記述するのではなく、ランナブル間で共有される変数や SWC の持つポートへのアクセスについても記

述することが出来る．

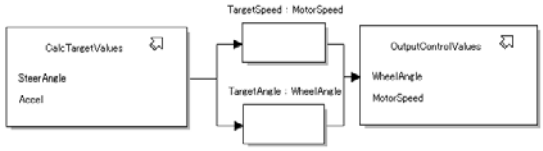

図3 ランナブル図

2.3　TECS における構造モデル図

TECS では、コンポーネントモデル、コンポーネント図、コンポーネント記述言語（TECS CDL（Component Description Language)）、コンポーネント実装モデルを仕様として規定しているが、ここでは、静的なコンポーネント結合という観点から TECS を取り上げる．コンポーネント図（図４）では、コンポーネント間の関係として、型をはじめ、その入出力まで明示的に記述するようになっている．

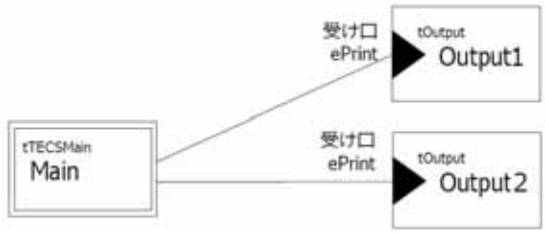

図4　TECS コンポーネント図

３．構造設計の必要性

UML にコンポーネント図があることからもわかるように構造設計という考え方自体は、古くから存在する．ここでは近年構造設計という考え方に注目が集まり、その必要性が増加している理由を考察する．

3.1　複雑化し続けるソフトウェアの見える化

組込みソフトウェアの複雑化はここ数年いわれ続けていることである．単機能それぞれの検討も必要であるが、複数の機能が相互に連携した複雑な機能を検討することが重要となっている．つまり、全体を俯瞰し、以下のような項目を設計/検討すること必要とされている．

- 機能/責務の分割
- コンポーネント間の依存関係

このような俯瞰のためには、構造設計図が適している．

3.2　非機能要件への強い要求

ソフトウェア開発で常に言われ続けているのが「再利用」である．近年では多品種少量開発が進んでいるためさらに「差分開発」への要求も強い．
つまり、以下のような項目を設計/検討できることが必要とされている．

- 機能としての再利用単位
- 再利用するための依存関係
- 製品 A と製品 B 間での差分

このような視点で俯瞰することにも構造図が適している．

４．構造設計と振舞設計の相互作用

構造設計と振舞設計の間での制約について考察した後、設計手順を説明する．

4.1　構造モデルと振舞モデル間の制約

構造モデルと振舞設計モデルは、１つのシステムの静的側面と動的側面を記述しているため、２つのモデル間には整合性が必要とされる．UML で振舞を記述するために使用される図の１つであるシーケンス図(図５)と構造図の１つであるクラス図を例にとりその制約を説明する．
シーケンス図は、オブジェクト間のメッセージの流れを時系列に沿って記述することで、振舞を記述する図である．従って、シーケンス図に登場するオブジェクトは、構造図に登場しなければならない．このように振舞設計は構造設計を受けたものでなければならない．逆のことも言える．構造図の１つであるクラス図で定義されたメソッドは、先のシーケンス図中での使用方法（受け渡し引数等）と整合性が取れていなければいけない．

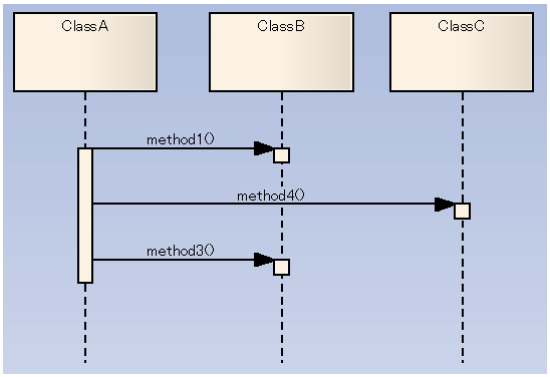

図5 シーケンス図

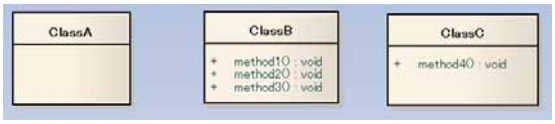

図6 クラス図

このことは、構造設計・振舞設計それぞれを単独で進めるのではなく、協調しつつ設計を進める必要があることを示している．

4.2 構造設計と振舞設計の協調設計

ここでは、自動販売機を例として構造設計と振舞設計を仮想的に進めその設計手順を説明する．

ここでは、ユーザーが硬貨を投入し商品を選択して購入するような自動販売機を対象とする．投入した金額に応じたボタンが点灯する等の要件は設定されているものとし、要件自体を検討対象とはしない．

システム全体の振舞

最初に、システムと外界とのやり取りを記述する．このことでシステムとして対応すべき事象や保持すべき状態を抽出する．

自動販売機と利用者とのやり取り（の１例）は以下の通りとなる（図７）．

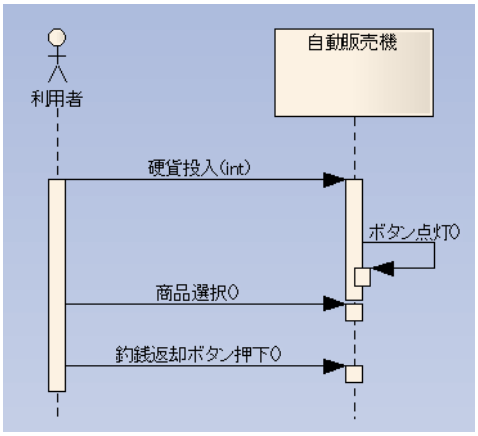

図7 自動販売機シーケンス図１

このようなやり取りを十分に考えシステム（この場合は自動販売機）で対応すべきイベントや保持すべき状態を抽出する．

システム内部構造の詳細化

次に、先の図では自動販売機となっているシステム全体の構造を詳細化していく．構造を詳細化し全体の要求仕様から導出される要件についてシステム内部のどのコンポーネントで責務を持つのかを決定し詳細化していく．（図８）

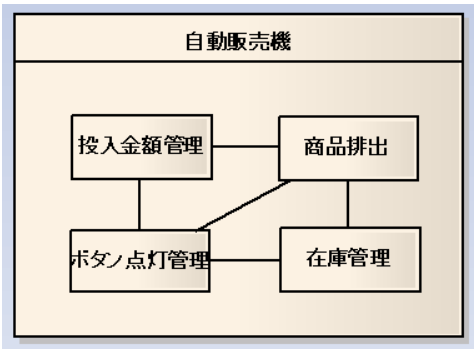

図8 複合構造図

ここでは、複合構造図で記述し、責務を各パートに分割している．これらのパート間の関係は、振舞設計を詳細化する段階でシーケンス図等を記述することで詳細に設計される．

一部のコンポーネントを外部調達する必要があるならば、この段階で、責務分割・インターフェース等

の不整合を調整する必要がある．
また、コンポーネントとして再利用を考慮する必要があるならば、想定する再利用時のインターフェースや再利用単位について調整する必要がある．

詳細化した構造の振舞詳細化

詳細化された構造モデルの各コンポーネントについて振舞を設計する．詳細化された構造モデルに対してシーケンス図を作成することで、対応すべきイベント、保持すべき状態を抽出することができる．抽出したイベント・状態から状態遷移表を作成することでモレヌケを検出することも可能である．
明確にモデリングされた構造情報を元に詳細な振舞設計をすることで、入出力情報の過不足や、保持すべき状態の妥当性を確認することが出来る．
内部の振舞を詳細に設計することで、入出力の情報として過不足ないかも確認することができる．
コンポーネントを外部調達した際には、この段階で、構造モデルから期待されるインターフェースや振舞と調達したコンポーネントのインターフェースや振舞との差異について確認/調整することが可能である．

構造設計・振舞設計のイテレーション

構造設計・振舞設計ともに一度行えば終了ということではなく、繰り返し行うことで、設計を洗練させることができる．検討を進めることで構造設計・振舞設計ともに不足な点/洗練させるべき点が見つかりお互いにフィードバックしていく関係となる．時にはシステムと外界とのやり取りも洗練する対象となる．

５．今後の課題

５.１　非機能要件項目の評価手法の不在

構造設計は、拡張性やメンテナンス性といった数値化することが難しい非機能要件の洗練もその目的とすることが多い．さらに、可変性への対応や交換可能性といった、設計時点での評価が難しい非機能要件も存在する．このような評価軸についての評価手法ついてもなんらかの指針等が求められる．

５.２　アーキテクチャパターン/構造設計の再利用

MVC（Model-View-Controller）コントローラが有名であるが、アーキテクチャパターンというアプローチが存在する．再利用対象としては、コンポーネントそのものだけが対象となりがちであるが、構造設計そのものを再利用することで、さらなる品質の向上が見込まれる．

５.３　プラットフォームへの依存

構造モデルやコンポーネントという概念が存在するためには、そのためのプラットフォームの存在が欠かせない．AUTOSAR はプラットフォームそのものまでを標準化の対象とし、TECS では TOPPERS がプラットフォームとして存在している．構造設計は、下層にあたるプラットフォームを意識しなければならないため、どの程度プラットフォームに依存する構造とするかも検討が必要である．

６．むすび

構造モデルについて既存のモデルから説明し、振舞モデルとの協調設計について説明した．
今後もさらに大規模化・複雑化していくと予想される組込みソフトウェアの設計いう行為について、これまで以上に構造設計の重要度が増してくる．
これまで構造モデルを重要視してこなかった設計者に本稿をきっかけに構造モデルについて再考いただくことを期待する．

参考文献

[1] AUTOMOTIVE OPEN SYSTEM ARCHITECUTURE
http://www.autosar.org/
[2] TOPPERS プロジェクト/TECS

http://www.toppers.jp/tecs.html

[3] ObjectManagementGroup,UML2.0 仕様書、オーム社,平成 18 年

[4] 統一モデリング言語 Wikipedia http://ja.wikipedia.org/wiki/%E7%B5%B1%E4%B8%80%E3%83%A2%E3%83%87%E3%83%AA%E3%83%B3%E3%82%B0%E8%A8%80%E8%AA%9E#UML.E3.81.AE.E3.83.80.E3.82.A4.E3.82.A2.E3.82.B0.E3.83.A9.E3.83.A0

[5] テクノロジックアート、UML2 ハンドブック 、翔泳社、2004 年 4 月

[6] 渡辺、組込みソフトウェア開発課題への挑戦 ～統合化～,http://www.zipc.com/cesl/info/07.pdf

[7] 今井、制御システムモデルと組み込みソフトモデル(1) モデルベース開発, http://www.zipc.com/cesl/info/08.pdf

柳下 知昭