形式手法による組み込みソフトウェア開発

第4回 行列とLTSAを使った設計の検証

藤倉 俊幸

要求間の関係が複雑だと，一つの機能を実装するのに複数のクラスに影響があるなど，見通しが悪くなります．見通しの悪さを定量的に評価するために，これを行列で表してみます．また，前回紹介したLTSAで仕様をステート・マシンにして検証し，続々と問題が発覚するようすを見ていきます．（編集部）

1．おさらいと要求リストの見方

前回（第3回，2012年8月号）はカップラーメン・タイマの仕様からクラスを設計することを，従来手法で行いました．今回は本連載のテーマである形式手法によりクラス設計を行いましょう．これにより，従来手法でありがちな仕様の抜け・漏れを無くすことができます．

今回は演繹（えんえき）による（要求から仕様を導き出す）項目別の設計を完結させます．要求リストと進捗状況を**表1**に示します．前回も使用した表ですが要求間の関係も**表2**に示します．今回の作業は，**表1**の中でまだ形式化していない要求を形式化することです．

● 階層関係があることに注意

表1で，上位要求に対応する行の実装欄を，「実装済み」ではなく「－」としてあるのは，下位要求を実装することで実装済みになるからです．Spec07は第2回（2012年6月号）で実装および検証しました．しかし，実質的にはSpec04に完全に含まれているので今回の表では上位要求として扱うことにしました．何だか適当な感じですが，本来であれば要求分析の結果を踏まえて，Spec04から実装すればSpec07だけ取り出して実装する必要は無かった訳です．あえて，第2回でSpec07を実装したのは，形式化の例を示すためでした．実際の開発で，簡単なところから作り始めると不必要なスクラップ＆ビルドが多くなり，仕事をしているのに先に進まないことになるので要求分析の結果を踏まえて計画的に実装することが必要です．

● 実装しない上位要求にも意味がある

ところで，実装しない上位要求は何のためにあるのでしょうか．一つは，システムの目的を明確にするためです．二

表1　カップラーメン・タイマの要求リスト

要求IDのReqは要求を，SpecはReqを満たすための仕様を示す．仕様を実装すれば要求を満たすことができる

要求ID	内　容	実　装	形式化
Req01	タイマを起動してから，設定した時間が経過したことを知らせる	—	
Req02	設定時間は30秒刻みで設定できる	—	
Req03	LED1で電源がONであること（プログラムが動いていること）を表示する	—	
Req04	LED4でタイマが動いていることを表示する	—	
Spec01	電源がONの間（プログラムが動いている間），LED1の点灯・消灯を1秒間隔で繰り返す	実装済み	済み（第3回）
Spec02	SW1がONになると設定時間を30秒にして，タイマを起動する	実装済み	
Spec04	タイマが動いている間にSW8がONになると，ONの間LED2を点灯させ，設定時間を30秒延長する	実装済み	
Spec05	タイマが動いている間は，10秒間隔で，LED4を2回点滅させる．LED4の2回点滅は，点灯・消灯を0.25秒間隔で2回繰り返すことで行う	実装済み	
Spec06	設定時間が経過すると，LED4を15秒間点滅させた後，消灯する．LED4の点滅は，点灯・消灯を0.25秒間隔で繰り返すことで行う	実装済み	
Req05（Spec07）	LED2でSW8が押されたことを表示する	—	済み（第2回）
Spec09	SW8で設定時間を30秒延長する	実装済み	

表2　要求間の関係

「仕様6は要求1を満たす」，「仕様2と仕様9で要求2を満たす」のような関係がある

第1回式番号	要求間の関係
5	Spec06⇒Req01
6	Spec02∧Spec09⇒Req02
7	Spec05⇒Req04
8	Spec01⇒Req03
9	Spec04⇒Spec09
11	Spec04⇒Spec07