# 組み込みソフトウェアのアスペクト指向ソフトウェアアーキテクチャ
# ～自動販売機のカップ機構制御ソフトウェアアーキテクチャの構築～

熊崎 敦司
南山大学経営学部情報管理学科
TEL: (052)832-3111
d00bb002@nanzan-u.ac.jp

後藤 修平
南山大学経営学部情報管理学科
TEL: (052)832-3111
m03bb004@nanzan-u.ac.jp

野呂 昌満
南山大学数理情報学部情報通信学科
TEL: (0561)89-2000
yoshie@nanzan-u.ac.jp

張 漢明
南山大学数理情報学部情報通信学科
TEL: (0561)89-2000
chang@nanzan-u.ac.jp

**Abstract**

ソフトウェアの構造は，注目する視点ごとに最適な構造が異なる．この解決策の一つとしてアスペクト指向が注目されている．本研究に関連する我々の研究の最終目的は，組み込みソフトウェアのアスペクト指向ソフトウェアアーキテクチャを提案することである．組み込みソフトウェアという領域は広すぎるので，事例毎に構築したソフトウェアアーキテクチャを抽象度を鍵に階層化，整理する．本研究では，その第一段階として自動販売機のカップ機構制御のソフトウェアアーキテクチャを構築する．このさいに，複数の視点から見ても整理された構造を得るためにアスペクト指向を適用する．ハードウェア制御と時間制御をアスペクトとして分離することで，組み込みソフトウェアの開発に有効なアーキテクチャが提案できた．

## 1 はじめに

組み込みソフトウェアは開発対象が拡大し，要求される機能が高度化・複雑化してきている．これまで組み込みソフトウェア開発は，ハードウェア主導でソフトウェアは付随するものという位置付けでおこなわれてきたが，昨今，組み込みソフトウェアの構造を整理するためにオブジェクト指向が導入されつつある．

ソフトウェアの構造は，注目する視点ごとに最適な構造が異なる．組み込みソフトウェアには，ハードウェアの同期処理，時間制御などの特徴がある．組み込みソフトウェアをオブジェクト指向開発するとき，ハードウェア制御を中心に設計すると，時間制御が複数のクラスに散在した構造になってしまう．

本研究の目的は，組み込みソフトウェアのソフトウェアアーキテクチャを提案することである．組み込みソフトウェアという領域は広すぎるので，抽象度の高いソフトウェアアーキテクチャを提案し，事例を適用する．事例毎に構築したソフトウェアアーキテクチャを抽象度を鍵に階層化，整理することが研究の最終目標である．本研究では，その第一段階として，自動販売機のカップ機構制御ソフトウェアを適用し，そのソフトウェアアーキテクチャを構築する．

カップ機構制御のソフトウェアアーキテクチャを構築するさいに，複数の視点から見ても整理された構造を得るためにアスペクト指向 [3] を適用する．構築したアスペクト指向アーキテクチャ[4] を用いて，カップ機構制御のシミュレータを試作し，組み込みソフトウェア開発にソフトウェアアーキテクチャを用いることが有効であることを確認する．

アスペクト指向を用いることで，ハードウェア制御と時間制御，それぞれをアスペクトとしてきれいに分離したソフトウェアアーキテクチャを得ることができた．カップ機構制御のソフトウェアアーキテクチャにもとづきシミュレータを試作することにより，組み込みソフトウェアにアスペクト指向ソフトウェアアーキテクチャを適用することが有効であることが確認できた．ハードウェア

制御と時間制御をもつ組み込みソフトウェアの開発に有効なソフトウェアアーキテクチャが提案できたと考えられる．

## 2 アスペクト指向ソフトウェアアーキテクチャ

我々はこれまでにアスペクト指向ソフトウェアアーキテクチャ[4] を提案した．MDSOC[5] の考えに基づいてソフトウェアを多次元分離し，ソフトウェアをアスペクトの集合と考える（図 1参照）．

図 1. アスペクト指向ソフトウェアアーキテクチャ

一般にアスペクト指向計算では，アスペクト間の関係は合流点上にアドバイスとして記述される．同様に，我々が考えるアスペクト指向ソフトウェアアーキテクチャでもアスペクトは合流点によって結合される．一つのアスペクトは構成要素のコンテナと考えることができ，合流点はコネクタと考えることができる．

## 3 組み込みソフトウェアのソフトウェアアーキテクチャ

組み込みソフトウェアは，ハードウェアの同期処理や時間制御を行うという特徴から，ハードウェア制御・時間制御と，それらをとりまとめるアプリケーションロジックで構成でき，これらをアスペクトとして分離することが考えられる．アスペクトを分離することで，変更点がわかりやすくなり，変更に強い構造になる．

### 3.1 組み込みソフトウェア設計の指針

組み込みソフトウェアの特徴としてハードウェア制御があげられる．ハードウェア制御を行うさいには，ハードウェアの変更容易性，ハードウェアはそれぞれ独立に稼動している点，ハードウェア間の同期を考慮する必要がある．我々は次の 3 つをハードウェア制御の設計の指針とした．

- 仮想ハードウェア
- 並行オブジェクト
- ハードウェア間の協調関係

#### 仮想ハードウェア

ハードウェアを制御する仮想ハードウェアオブジェクトを用意することで，ハードウェアの変更に柔軟に対応できる．仮想ハードウェアを置けば，ハードウェアの変更のさいに，仮想ハードウェアオブジェクトを変更するだけで対応することが可能である．

一つのハードウェアが複数の用途に使用される場合には，ハードウェアの用途の数だけ仮想ハードウェアを用意する．ハードウェアを制御するアプリケーションロジックからみると，これらは別々に実現するほうがハードウェアの役割が明確になり，変更箇所の特定，局所化に有効である．図 2は，一つのハードウェアに対して複数の仮想ハードウェアが存在する例である．これは自動販売機のカップ制御を実現するさいに必要なハードウェア，モータの例である．モータはカップの排出，供給の両方で使用されることからそれぞれに対して仮想ハードウェアを用意している．

仮想ハードウェアを置くことで，ハードウェアを扱うさいには，仮想ハードウェアにメッセージを送信する．ハードウェアに直接メッセージを送信することはない．図 2の例では，仮想ハードウェアに排出開始というメッセージが送信されたら，モータが正回転を行う．各ハードウェアの仮想ハードウェアオブジェクトは独立であり，それぞれがメッセージ通信を行うことはない．

#### 並行オブジェクト

組み込みソフトウェアの特徴として，ハードウェアはそれぞれ別々に動作するものであることがあげられる．組み込みソフトウェアを開発する上で，ソフトウェア上でもハードウェアの扱いをもともとのハードウェアに近づけるために，仮想ハードウェアを並行オブジェクトとして設計する．

図 2. ハードウェアと仮想ハードウェアの例

**ハードウェア間の協調関係**

組み込みソフトウェアの開発において，各ハードウェア間の協調関係を管理する方法としてつぎの 2 つがあげられる．

- 各ハードウェアごとに状態遷移機械を置きハードウェア間の協調関係を管理する方法
- ハードウェア全体を管理する状態遷移機械を置く方法

本研究では，ソフトウェアの制御ごとに仮想ハードウェアを置き，ハードウェア間の関係を複雑にならないようにするので，各ハードウェアごとに状態遷移機械を置き，ハードウェア間の協調関係を管理する．

### 3.2 アスペクトの分離

アスペクト指向技術を用いて組み込みソフトウェアの構造をいくつかのアスペクトに分離することで，組み込みソフトウェアの構造を整理する．本研究では，ハードウェア制御と時間制御を特徴とする組み込みソフトウェアを対象としている．ハードウェア制御を中心に構造を整理すると時間制御がきれいに整理できない．また，時間制御を中心に構造を整理するとハードウェア制御がきれいに整理できないので，これらを分離する．さらに，これらを残ったアプリケーションロジックから分離する．想定するアスペクト指向ソフトウェアアーキテクチャ[4]は，以下の 3 つのアスペクトから構成される．

- アプリケーションロジックアスペクト
- ハードウェアアスペクト
- 時間アスペクト

**アプリケーションロジックアスペクト**

アプリケーションロジックアスペクトは，ハードウェア全体を制御するので，複数の仮想ハードウェアで構成される複合ハードウェアオブジェクトで構成される．

**ハードウェアアスペクト**

ハードウェアアスペクトは，仮想ハードウェアオブジェクトとハードウェアオブジェクトで構成され，仮想ハードウェアがハードウェアの制御を行う（3 節参照）．

**時間アスペクト**

時間アスペクトは，仮想タイマオブジェクトとタイマオブジェクトで構成され，仮想タイマがタイマの制御を行う（3 節参照）．

### 3.3 組み込みソフトウェアのアスペクト指向アーキテクチャ

組み込みソフトウェア設計の指針と 3.2節で分離したアスペクトをもとに，我々が想定したアスペクト指向アーキテクチャを図 3に示す．図 3の太線は，一つのアスペクトが別のアスペクトと横断的に関連（クロスカット）することを表す．

## 4 自動販売機のカップ機構制御ソフトウェアの設計と試作

本研究で想定した組み込みソフトウェアのアスペクト指向アーキテクチャが，組み込みソフトウェア開発に有効であることを確かめるために，自動販売機のカップ機構制御のカップ排出制御のシミュレータの分析・設計・実現を行う．カップ機構制御は，以下にあげる 3 つの制御に分類できるが，どの制御も同じ構造で実現できると考え，カップ排出制御だけの試作をおこなった．自動販売機のカップ機構制御ソフトウェアは状態遷移機械を保持した並行オブジェクトの集まりであると考えた．並行オブジェクト間のメッセージ通信の実現には非同期処理の実現に有効なアクティブオブジェクトパターン [6] を用いた．

### 4.1 自動販売機のカップ機構制御

自動販売機のカップ機構制御は複雑であるので機能を分類し，単純化してわかりやすくした．自動販売機のカップ機構制御は，以下の 3 つの制御からなる．

図 3. 組み込みソフトウェアのアスペクト指向アーキテクチャ

- カップ検索制御
- カップ供給制御
- カップ排出制御

カップ検索制御とは，カップをためておく収納筒から一致するカップの種類を検索する制御である．カップ供給制御とは，検索制御のあと収納筒からカップ供給位置へカップを移動させる制御である．カップ排出制御とは，排出要求によって一つのカップを排出する制御である．カップ検索制御・カップ供給制御・カップ排出制御は同じ構造になると考え，本研究では，カップ排出制御のシミュレータの試作を行う．

### 4.2 カップ排出制御の分析

分析段階では，カップ排出制御を実現するための仮想ハードウェアを分析し，仮想ハードウェアを制御する複合ハードウェアを用意する．複合ハードウェアがカップ排出制御を実現するので，複合ハードウェアの制御を分析し，メッセージの順序を状態遷移で表す．

#### クラス構成

カップ排出制御は，以下のハードウェアを使用する．ここで，タイマもハードウェアと同様に扱う．

- カップモータ
- カップモータセンサ
- タイマ

カップ排出制御に使用するハードウェアを分析した結果，つぎの 3 つの仮想ハードウェアを用意することにした．

- 仮想カップモータ（ドロップリングモータ）
- 仮想カップモータセンサ（ドロップリングセンサ）
- 仮想タイマ

用意した仮想ハードウェアは，ドロップリングの構成要素であり，ドロップリング専用の仮想ハードウェアとなる．ドロップリングは，上で述べた仮想ハードウェアを制御する複合ハードウェアオブジェクトである．

#### カップ排出制御の状態遷移

カップ排出制御にはメッセージ通信の順序があるので，カップ排出制御を分析し，ドロップリングオブジェクトのメッセージ通信の順序を状態遷移であらわす．仕様書をもとに状態遷移を分析すると，状態遷移するさいのア

クションが複数になる．複数のアクションを抽象化するとカップ排出制御がわかりやすくなると考え，分析した状態遷移を抽象化した（図 4）．抽象化した状態遷移は複合ハードウェアの状態遷移になる．状態遷移を抽象化することにより，ドロップリングから各ハードウェアの制御を隠蔽できる．

### 4.3 カップ排出制御の設計

ハードウェアはそれぞれ別々に稼動しているものであるので，ソフトウェア上でもハードウェアを別々に稼動するものと考え，各仮想ハードウェアを並行オブジェクトとして設計した．設計したカップ排出制御のクラス図を図 5に示す．図 5の<<active>>ステレオタイプは，並行オブジェクトであることを示す．図 5の太線の矢印はアスペクト間の関係を表す．各並行オブジェクトが保持する状態遷移機械の振る舞いをコアコンサーン，アスペクトを構成する各並行オブジェクトへのメッセージ通信をアドバイス [3] として設計する．

#### 4.3.1 並行オブジェクト間のメッセージ通信

設計段階で，複合ハードウェアを含む各ハードウェアを並行オブジェクトとする．ドロップリングとドロップリングセンサを例にとり並行オブジェクト間のメッセージ通信を説明する．

**ドロップリングの状態遷移**

ドロップリング並行オブジェクトのメソッドの実行順序を決定するためにドロップリングの状態遷移を設計した．ドロップリングオブジェクトが状態を持つことによって呼び出されたメソッドが実行可能か判断できる．

**ドロップリングのシーケンス**

ドロップリングオブジェクトは，メソッドが呼ばれ，実行可能かどうかを現在の状態によって判断する．実行可能であればドロップリングオブジェクトの状態を遷移させる．状態遷移を行うさいに，ハードウェアコンサーン・タイマコンサーンへメッセージを送ることになっている．アスペクト間のメッセージ通信による返り値は，未来型（Future）オブジェクト [7] を使用する．複合ハードウェアの設計段階では具体的なジョインポイント [3] は決定していない．実現段階でジョインポイントを決定する．

**ドロップリングセンサのシーケンス**

ドロップリングと同様に，ドロップリングセンサオブジェクトが状態を持つことによって，呼び出されたメソッドが実行可能か判断できる．ドロップリングセンサは，ドロップリングコンサーンからのメッセージ通信によって動く．処理結果をドロップリングオブジェクトに返す場合，返り値の受け渡しには未来型オブジェクトを使っているので，未来型オブジェクトに返り値を渡す．

### 4.4 カップ排出制御の実現

4.3節で設計した並行オブジェクトを実現するために，アクティブオブジェクトパターンを用いた．アクティブオブジェクトパターンではメッセージ通信の順序を決められないのでアクティブオブジェクトパターンに状態遷移機械を追加した．アスペクト間の関係の記述には，既存のプログラミング言語で記述可能なアスペクトモデレータフレームワーク [1] を用いた．

アクティブオブジェクトパターンでは，並行オブジェクトに送られたメッセージがキューに積まれ，先着順にメッセージに対応する処理が行われる．拡張したアクティブオブジェクトでは，キューに積まれたメッセージを状態遷移機械によって，状態遷移可能であるかを判断する．状態遷移可能であれば，メッセージに対応する処理を行い，状態遷移できなければメッセージを破棄する．

アスペクトモデレータフレームワークを用いることでアスペクト間の関係の記述を分離することができる．あらたにアスペクトを追加するさいもアスペクト間の関係だけを追加すればよい．

**アクティブオブジェクトパターン**

設計段階で，各仮想ハードウェアを並行オブジェクトとした．ドロップリングをアクティブオブジェクトパターンを使用して実現したさいのクラス図を図 6に示す．

- Proxy クラス

  アクティブオブジェクトへのメッセージを Request オブジェクトに変換し，ActivationQueue に積む．

- Scheduler クラス

  ActivationQueue に積まれたメッセージを取り出し，実行する．

- State クラス

  ActivationQueue から取り出されたメッセージが実行可能であるかを，現在の状態によって判断する．

- ActivationQueue クラス

  アクティブオブジェクトへのメッセージを Request オブジェクトとしてためておく．

- ServantClass クラス

図 4. カップ排出制御の状態遷移図

図 5. カップ排出制御のクラス図（設計段階）

ActivationQueue から取り出されたメッセージを実行する．

– Result クラス

実行結果の返り値を扱うクラス．サブクラスに FutureResult と RealResult を持つ．

#### ドロップリングアクティブオブジェクト

ドロップリングにメッセージが送られると，ActivationQueue にメッセージが積まれる．Scheduler が ActivationQueue からメッセージを取り出し，state クラスで実行可能かどうかを判断し，実行可能であれば Servant がメッセージを受け取り，実行する．Servant のメソッド呼び出しをジョインポイントとし，メソッド呼び出し後にハードウェアコンサーン・タイマコンサーンにメッセージが送られる．実行結果の返り値は，Future パターン [7] を用いて実現する．ハードウェアコンサーン・タイマコンサーンからの実行結果が Future オブジェクトに設定される．

#### ドロップリングセンサアクティブオブジェクト

ドロップからのメッセージによってドロップリングセンサが動作する．ドロップリングセンサアクティブオブジェクトの挙動は，ドロップリングアクティブオブジェクトとほぼ同様である．Servant が CupMotorSensor に read メッセージを送った結果を，ドロップリングアクティブオブジェクトの Future に送る．Future に送る動作は Servant クラスには記述されず，read メソッドがジョインポイントとなり，read メソッド呼び出しのアドバイス（after）に記述される．

## 5 考察

本研究で提案した組み込みソフトウェアのアーキテクチャを用いることで，ソフトウェアの変更点を局所化することができた．コンサーンの分離に関してと，状態遷移機械を用いたハードウェア間の協調関係の管理によるソフトウェアの変更点の局所化について考察する．

### 5.1 ハードウェアの追加・変更に関する考察

組み込みソフトウェアを 3 つのコンサーンに分類したことで，ソフトウェアに変更が生じても，変更するコンサーン以外のコンサーンには影響がないという利点がある．ハードウェアコンサーンが変更される場合を例にとり，変更点の局所化について考察する．

組み込みソフトウェア開発において典型的なケースであるハードウェアの追加を考える．ハードウェアの追加の例として，故障を検知するハードウェアを追加するケースを想定する．故障検出用のハードウェアが追加されるので，仮想ハードウェアも追加する．オブジェクト指向実現だと，ドロップリングクラス内の記述を変更することになる．アスペクト指向実現の場合は，ドロップリングのメソッドのアドバイスに故障検出処理を追加することで，ハードウェアの追加に対応できる．ドロップリングクラスを変更する必要はないので，ドロップリングクラスはハードウェアの変更によって変更されることはない．

ハードウェアの変更には，変更するハードウェアのインターフェースに変更が有る場合と無い場合がある．前者は仮想ハードウェアのメソッドの中身を変更することで対応できる．後者は仮想ハードウェアを変更することはない．どちらもアスペクト間の記述には全く影響を与えないので，変更に強い構造である．

仮想ハードウェアのメソッドが変更になるケースを考察する．このケースはアスペクト間のアドバイスを変更することで対応することができ，アプリケーションロジックコンサーン側には影響を与えない．

オブジェクト指向設計・実現したカップ排出制御ソフトウェアとアスペクト指向設計・実現を比較した結果，オブジェクト指向実現では，多数のオブジェクトに変更点が及ぶが，アスペクト指向実現では，変更点が他のコンサーンに及ぶことはない．

### 5.2 ハードウェア間の協調関係に関する考察

組み込みソフトウェアを開発するさいに，各ハードウェア間の協調関係を管理する方法は，ハードウェア全体を管理する状態遷移機械を置く方法 (図 7) と，各ハードウェアごとに状態遷移機械を置きハードウェア間の協調関係を管理するという方法 (図 8) がある．

前者の利点は，ハードウェア間の協調関係の管理が 1 か所でできることである．しかし，ソフトウェアが大規模化すると，ハードウェアを管理するオブジェクトも大きくなり，ソフトウェアの変更に柔軟に対応できなくなるという欠点がある．

後者の利点は，あるハードウェアが制御に必要な他のハードウェアだけを知っていればよいので，ハードウェア間の関係がわかりやすいことである．欠点は，一つのハードウェアが多くのハードウェアと関わっている場合，協調関係が複雑になってしまうことである．

本研究では，ソフトウェアの制御ごとに仮想ハードウェアを置き，仮想ハードウェアの並行動作を制御する複合ハードウェアを用意した．仮想ハードウェアと複合ハードウェアに状態遷移機械を置き，コンポーネント間の協調関係を管理した．その結果，ハードウェアの追加・変更に柔軟に対応し，協調関係が複雑にならない構造にできた．

図 6. ドロップリングアクティブオブジェクトのクラス図

図 7. ハードウェア全体を管理する状態遷移機械を置く例

図 8. ハードウェアごとに状態遷移機械を置く例

### 5.3 組み込みソフトウェアのソフトウェアアーキテクチャの評価

本研究では，組み込みソフトウェアの設計上の留意点と分離したコンサーンをもとにアスペクト指向ソフトウェアアーキテクチャを提案した．提案したソフトウェアアーキテクチャにもとづき，自動販売機のカップ機構制御を分析・設計・実現を行い，カップ排出制御のシミュレータを試作した．

自動販売機のカップ機構制御を設計，シュミレータを試作した結果，ハードウェア制御と時間に関するコンサーン，それらを制御するコンサーンを分離することができ，複数の視点からみても最適な構造を得ることができた．提案した組み込みソフトウェアのソフトウェアアーキテクチャの有用性を確認することができた．

提案したアスペクト指向ソフトウェアアーキテクチャを用いることで，以下の利点がある．

- アプリケーションロジック・ハードウェア・タイマのコンサーンの分離によるソフトウェアの構造の整理
- ソフトウェアの変更点の局所化

## 6 おわりに

本研究では，我々が考えた組み込みソフトウェア設計の指針と組み込みソフトウェアのコンサーンをもとに，組み込みソフトウェアのアスペクト指向ソフトウェアアーキテクチャを提案した．提案したソフトウェアアーキテクチャにもとづいて，カップ機構制御の機能の一部であるカップ排出制御のシミュレータを Java 言語で試作し，組み込みソフトウェアの開発に，我々が提案したアーキテクチャを用いることが有効であることを確認した．提案したソフトウェアアーキテクチャによって，ハードウェア制御と時間制御をもつ組み込みソフトウェアの開発支援になると考える．

今後の課題を以下に述べる．

- 正常・異常終了の問題点

  タイミングコンサーンの時間制御の問題点について述べる．異常終了のあとに正常終了のイベントがきたケースと同時イベントを考える．今回のカップ機構制御では，Future オブジェクトを用意し，早いほうの返信を受け取って，以後は受けつけないようにしていた．異常終了イベントが送られた後の正常終了イベントをどう扱うかが問題である．時間制御の問題点は自動販売機をはじめとする組み込みソフトウェアでは特に考慮すべき問題である．

- Java 以外のプログラミング言語による実現

  本研究では，オブジェクト指向開発・アスペクト指向適用という観点から Java 言語を用いてカップ排出制御の実現をおこなった．しかし，開発現場では C や C++ などの言語で開発されていることがある．開発現場にあたらしい言語を導入することは，時間・コストなどにマイナスの影響を与える．したがって，開発現場で用いられている C や C++に対する実行環境を作成することで，開発の効率化を高めていく．

  また，Java 言語以外のプログラミング言語で実現することは Java のマルチスレッド機能を利用できなくなる．昨年提案した自動販売機の設計と試作 [2] では全体のスケジューリングをおこなうモニタを導入してこの問題に対応した．C や C++ で実現し，比較・検討をおこなう．

- 他の事例への適用と階層化した組み込みソフトウェアのソフトウェアアーキテクチャ群の構築

  本研究では，ハードウェア制御と時間制御を組み込みソフトウェアの 2 大特徴として扱った．今後，他の組み込みソフトウェアのソフトウェアアーキテクチャとの共通部分を抜き出すことで，階層化，整

理したソフトウェアアーキテクチャ群が構築できる．組み込みソフトウェアに広く通用するソフトウェアアーキテクチャの構築につながる．

## 参考文献


[1] C. Constantinides, A. Bader, T. Elrad and M. Fayad “Designing an Aspect-Oriented Framework in an Object-Oriented Environment,” *Computing Surveys 32, 41*, 2000.

[2] 森貴彦, 服部和真, “自動販売機制御ソフトウェアのアーキテクチャスタイルに関する研究”, 南山大学経営学部情報管理学科卒業論文, 2001.

[3] 野呂昌満: “プログラミング言語から見たアスペクト指向開発技術”, ソフトウェアシンポジウム論文集, pp.169-178, 2002.

[4] M. Noro and A. Kumazaki, “On Aspect-Oriented Software Architecture : It Implies a Process as Well as a Product,” *Proceedings of APSEC*, 2002.

[5] H. Ossher and P. Tarr, “Multi-Dimensional Separation of Concerns and the Hyperspace approach”, in *Proceedings of the Symposium on Software Architectures and Component Technology: The State of the Art in Software Development.* Kluwer, 2001.

[6] D. Schmidt, M. Stal, H. Rohnert, F. Buschman, *Pattern-Oritented Software Architecture , Volume 2: Patterns for Concurrent and Networked Objects,* John Wiley & Sons, 2000

[7] D. Lea, *Concurrent Programming in Java, Design Principles and Patterns, Second Edittion,* Addison-Wesley, 1999.