II－11 内部境界条件を有する開水路不定流計算のオブジェクト指向プログラミング

Object Oriented Programming of Unsteady Open Channel Flow with Internal Boundary Conditions

池田裕一 [1]

Hirokazu Ikeda

堰や横越流など、さまざまな内部境界条件を有する1次元開水路不定流を陽解法で計算するシステムを、Java 言語によるオブジェト指向プログラミングで構築した。システムの中核となる「River」オブジェクトは、外部や内部の境界点の作用を表す「RiverNode」オブジェクトと、各点間を結ぶ開水路を示す「RiverChannel」オブジェクトの集合体として構成した。これら2種類のオブジェクトについては、基本クラスを継承した派生クラスによって、さまざまな境界条件、計算方法、断面形状に対応する多様なものを用意できるようにした。プログラムコードは、オブジェト指向プログラミングの多態性を利用した可読性のよいものが作成可能となり、個々のクラスの再利用も容易となった。

キーワード : 開水路不定流, 内部境界条件, 陽解法, Java, オブジェクト指向, MVC アーキテクチャ

Keywords : unsteady open channel flow, internal boundary condition, explicit method, Java, Object Oriented Programming, MVC architecture

1．はじめに

一昔前までは、土木工学分野で計算機の利用というと、数値計算が唯一の用途であった。ところが最近の情報関連分野の発展と社会需要の多様化により、コンピュータの用途は数値計算はもとより、表計算、データベース、Web アプリケーションなど、ずいぶんと多彩になってきた。

そのような情勢の中で、Java は注目すべきコンピュータ言語のひとつである。インターネットで結ばれた大規模な分散処理系の構築から、パソコン単位のアプリケーション、そして携帯端末での小規模アプリケーションまで、多様なプラットフォームでの多彩なアプリケーションの構築をサポートし得る。そして、最近になって広く普及してきたオブジェクト指向プログラミング(OOP)もサポートしている。

Fortran など従来の言語は構造化(あるいは機能詳細化)プログラミングと呼ばれる。これはプログラムに要求されている「機能」全体をまず大まかに分解し(たとえば入力, 計算, 出力など), それをさらに細かく分解していく手法である. これに対して, OOP は, プログラムを動作させたい世界がどのような「もの(オブジェクト)」で構成されるか検討していく手法で, 人間の認識により近いアプローチで, 複雑なシステムを扱うことができる. また、プログラムコードの再利用が容易ともいわれている。

ここで取り上げる1次元開水路不定流は、支配方程式自体はさほど複雑なものではない。ただし、開水路全体を計算していくなかで、上下流端の(外部)境界条件はもとより、堰やゲート、逆サイフォン、横越流など、さまざまな内部境界条件を処理しなければならない。また、河川の合流や分流も扱って河川網を計算するなどと考えると、システム全体はかなり複雑なものになる。

そこで本研究では、こうした1次元開水路不定流の計算に OOP を適用していく手始めとして、1つの河川にさまざまな内部境界条件が設定されている場合に OOP を適用し、Java 言語による実装を試みた。

2．問題の定式化

(1) 支配方程式

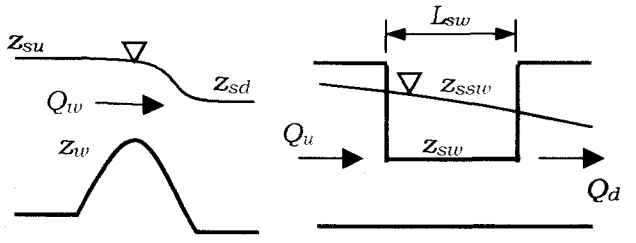
1次元開水路における不定流の連続式および運動方程式は次の2式である.

$$\frac{\partial A}{\partial t}+\frac{\partial Q}{\partial x}=0 \qquad \cdots(1)$$

$$\frac{\partial Q}{\partial t}+\frac{\partial vQ}{\partial x}+gA\frac{\partial z_s}{\partial x}+gA\frac{n^2|v|v}{R^{4/3}}=0 \qquad \cdots(2)$$

ここに, x:流下方向座標, t:時間, A:断面積, Q:流量, v:流速, z_s:水位, R:径深, n:粗度係数である。

1 : 正会員 博士(工学) 宇都宮大学大学院 助教授 工学研究科エネルギー環境科学専攻
(〒321-8585 栃木県宇都宮市陽東7－1－2、Tel :028-689-6215, E-mail : ikeda@cc.utsunomiya-u.ac.jp)



(a)堰　　(b)横越流

図—1　内部境界条件の例

これらの方程式の数値計算方法は枚挙に暇がないが、ここでは参考文献1)に説明されているスタッガード格子上の陽解法を採用する。これは常流・射流が混在している流れにも適用可能なものといわれている。ただし文献に示されていた人工粘性は(計算が複雑になるため)付加しないことにした。

（2）　境界条件

上流端では流量時系列を次のように式で与える[2)]。

$$Q_0(t) = Q_b + (Q_p - Q_b)\left\{\frac{t}{t_p}\exp\left(1-\frac{t}{t_p}\right)\right\}^C \quad \cdots(4)$$

ここに, Q_b :基底流量, Q_p :ピーク流量, t_p :ピーク時刻、C :洪水波形の形状を示す定数である.

下流端では特に境界条件は求めず、文献1)を参考に支配方程式と同様な方法で下流端の値を計算する。もちろん、下流端での水位を設定してもよいが、ここでは最も単純なものとした。

内部境界条件として。今回は堰と横越流を組み込むことにする。前者は流量を、後者は水深を決めるものなので、この2種類を検討しておけば、他の場合にも容易に応用できるだろう。堰については、堰の高さ z_w、上下流の水位 z_{su}、z_{sd} の大小関係から、そこを通過する流量 Q_w の大きさ、向きを決めるものとする[3)]（**図—1 (a)** 参照)。横越流の場合は、そこでの水位 z_{ssw} と越流堤の高さ z_{sw} との差ならびに越流長さ L_{sw} によって越流量量 Q_{sw} を計算し[4)]、それと上下流での流量 Q_u、Q_w との収支から次の時刻の水深を求めるものとする(**図—1 (b)** 参照)。

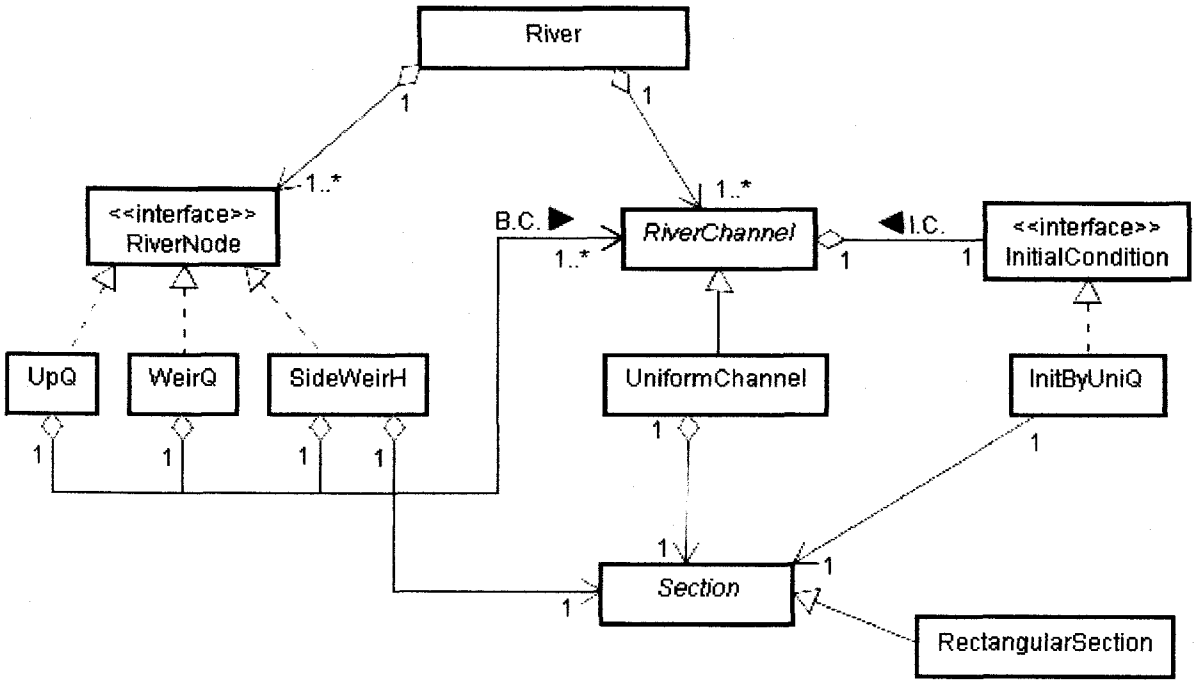
（3）　初期条件

初期条件の設定にもさまざまな方法がある。ここでは最も単純なもののひとつとして、流量は上流端で与える初期流量に、水深はそれに対応する等流水深にする。ただし、プログラムコードでは、他の方法による設定方法にも容易に切り替えられるよう工夫することにする。

3．クラスの設計

オブジェクトは実際世界の「もの」や「こと」を表すものであり, さまざまなデータを保有し、さまざまな動作をさせることが可能である。Java 言語ではそれぞれをフィールドおよびメソッドと呼ぶ。オブジェクトが有するフィールドとメソッドをまとめて記述したプログラム単位を「クラス」と呼び、オブジェクトの設計図のようなものである。

対象モデルにおけるオブジェクトあるいはクラスの構成を検討することは、この場のプログラミングを容易にするだけでなく、モデルの再利用性を高めるうえでも重要である。ここで取り上げた1次元開水路不定流の計算モデルのクラス図[5)]は**図－2**のようになる。以下順に説明していくことにする。



図—2　1次元開水路不定流の計算モデルのクラス構成

（1）　River クラス

これは、1つながりの開水路のデータと計算をまとめたクラスである。境界点を表す RiverNode オブジェクトと、それにより分割された個々の開水路を表す RiverChannel オブジェクトの集合体をフィールドとして保有する。またそれらから各時刻での水深や流量を取り出すメソッドを有する。

（2）　RiverNode インターフェースとその実装クラス

境界点を現すオブジェクトには、境界点の種類に応じてさまざまな機能が求められるが、これらを境界点を現す「同じ種類」のオブジェクトとしてくくってしまえるとプログラミングが楽になる。そこで、境界点に関するメソッドの呼び出し方だけを RiverNode インターフェースにより定めておき、そのメソッドの実装クラスとして、今回は上流端、堰、横越流の機能を表す UpQ、WeirQ、SideWeirH の3クラスを用意した。

（3）　RiverChannel クラスとその継承クラス

これは、一つの開水路部分を表すクラスで、格子点での水深や流量などの値やそれを計算するためのメソッドを有するものである。ただし、これには格子形成方法や計算方法、水路の特徴などにより、さまざまなバリエーションが考えられる。そこで本研究ではその最も上位に RiverChannel クラスを置き、そのデータやメソッドを継承して勾配および断面形状が一様な開水路における計算を実施するものとして、UniformChannel クラスをコーディングした。

（4）　InitialCondition インターフェースとその実装クラス

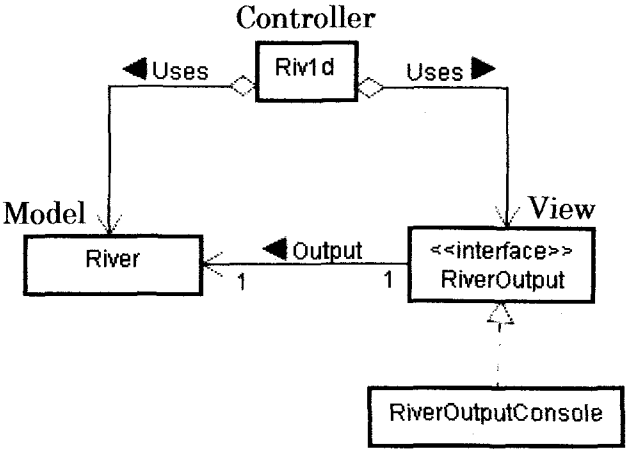
初期条件の設定は、開水路の計算を任されている RiverChannel クラスの仕事の一部であるが、さまざまな方法を用意して必要に応じて使い分けられるようにしておくと便利である。そこで、インターフェース初期条件を設定するメソッドの呼び出し方だけを InitialCondition インターフェースにより定めておき、そのメソッドの実装クラスとして、今回は流量とそれに応じた等流水深を与える InitByUniQ クラスを用意し、RiverChannl クラスのバリエーションによらず計算できるようにコーディングした。

（5）　Section クラスとその継承クラス

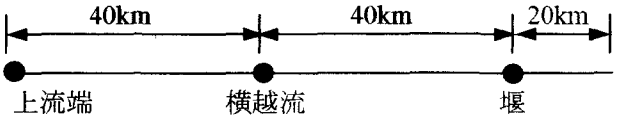
Section クラスは断面の性状にかかわるクラスである。今回の計算では、水深に応じて断面積や径深を計算したり、逆に断面積から水深や径深を計算したりすることがある。その際に必要な断面形状や摩擦係数のデータや計算方法を記述することにする。等流状態での水深や流量を求めるメソッドもこのクラスに含ませる。ただし、断面形状にはさまざまなものがあるので、今回は Section クラスを継承して長方形断面を扱う RectangularSection クラスを作成した。

4．アプリケーションの構築

次の段階として、準備された計算モデルのオブジェクトを生成して計算を行い、計算結果を出力するアプリケーションを構築することになる。アプリケーションの構成として典型的なものに MVC アーキテクチャがある。これは計算の中核的な役割を果たす Model 部分と、入出力の動作を行う View 部分、それらを制御する Controller 部分とに分けて設計する方式である（**図—3** 参照）。



図—3　コンソール出力によるアプリケーションの構成



図—4　開水路の構成例

表—1　計算例の計算条件

Q_b	200m^3/s、
Q_p	2000m^3/s
t_p	15 時間
C	20
横越流堤高さ	水路床より上 2m
横越流堤長さ	500m
堰高さ	水路床より上 1.5m
断面幅	200m
粗度係数	0.03
河床勾配	1/2000

今回の場合、Model には先に説明した River クラスが対応する。View には、結果を出力するための RiverOutput インタフェースを定め、その実装クラスとして単純端末に出力する RiverOutputConsole クラスをコーディングした。

Controller には、具体的なオブジェクトを生成して処理を行うコードを書く。これはもちろん、開水路の構成や利用するクラスによって異なってくる。ここでは具体例として、**図—4** に示すような3つの部分開水路と3つの境界点からなるからなる開水路について計算するものとする。その他の条件は**表—1** に示す通りとする。

リスト—1 は、そのプログラムコードのうち開水路を形成する部分を抜粋したものである。ここではまず3つの部

```
UniformChannel ch1
        = new UniformChannel(X11, Z11, X12, Z12, NX1);
UniformChannel ch2
        = new UniformChannel(X21, Z21, X22, Z22, NX2);
UniformChannel ch3
        = new UniformChannel(X31, Z31, X32, Z32, NX3);

RectangularSection sec = new RectangularSection(B, Rough);
ch1.setSection(sec);
ch2.setSection(sec);
ch3.setSection(sec);

InitByUniQ iniC = new InitByUniQ(Qb);
ch1.setInitialCondition(iniC);
ch2.setInitialCondition(iniC);
ch3.setInitialCondition(iniC);

UpQ up = new UpQ(ch1, Qb, Tb, Qp, Tp, Cp);
SideWeirH sweir
        = new SideWeirH(ch1, ch2, sec, Zsw, Dsw, Lsw);
WeirQ weir = new WeirQ(ch2, ch3, Zw, Dw, Bw);

River rv = new River();
rv.addChannel(ch1);
rv.addChannel(ch2);
rv.addChannel(ch3);
rv.addNode(up);
rv.addNode(sweir);
rv.addNode(weir);
```

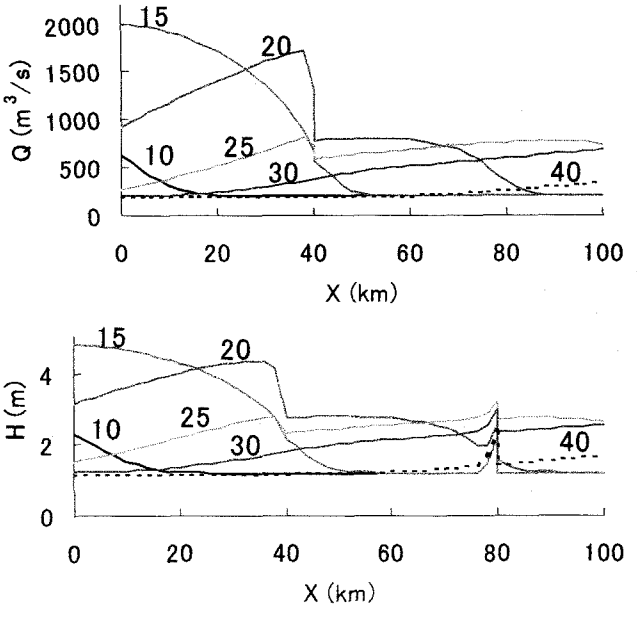
リスト－1 開水路を生成する部分のコード

```
RiverOutput rvOut
        = new RiverOutputConsole (rv, 41);

rv.init(0);
rvOut.output(rv, 0);

double t;
for (int i = 1; i <= Nt; i++){
        t = i * dt;
        rv.calc(t, dt);
        if (i % Nint ==0) rvOut.output(rv, t / 3600);
}
```

リスト－2 開水路不定流を計算する部分のコード



図—5 計算例(グラフ中の数字は t 時間を示す)

分開水路 ch1、ch2、ch2 を生成している。ついで長方形断面オブジェクト sec を生成して、ch1、ch2、ch2 に組み込んでいる。さらに初期条件を設定するために InitByUniQ オブジェクト iniC を生成させて ch1、ch2、ch2 に組み込む。以上で部分開水路3つが出来上がる。次に3種類の境界点オブジェクト up、seweir、weir を生成する。そして、River クラスのオブジェクト(変数名rv)を生成させて、これに先に生成させた3つの部分開水路と3つの境界点を組み込むものである。

リスト－2は不定流計算を実行している部分である。まず計算出力に必要な RiverOutput オブジェクト rvOut を生成させる。そして、先に生成した River オブジェクト rv について、init メソッドで初期条件を設定し、calc メソッドで時間刻み dt で1ステップずつ計算していく。その間、適当な間隔で計算結果を出力させている。

これらのリストを見て分かるように、コードの可読性は良好である。今後は、RiverNode や InitialCondition の実装クラスや、RiverChannel や Section の派生クラスを必要に応じてコーディングし、**リスト－1**と同様に開水路を生成させてやればよい。個々のクラスの再利用も容易であろう。

計算結果を**図－5**に示す。横越流による流量のカットや堰による堰上げ様子がうまく計算できている。

参考文献

1) 細田尚:常流・射流混在流れ、水理公式集プログラム例題集【例題 2-9】, 土木学会水理委員会, 2002.
2) 池田裕一:不定流計算, 水理公式集プログラム例題集【例題 2-2】, 土木学会水理委員会, 2002.
3) 伊藤良栄:Preissmann 型陰差分法における内部境界条件の実用的・安定的計算法、農業土木学会論文集 168、pp9-18、1993.
4) 水理公式集[昭和 60 年版]、土木学会水理委員会、1985.
5) 浅海智晴:UML & Java オブジェクト指向開発 入門編, ピアソンエデュケーション, 240p., 2002.