The Invisible Modelling Framework (TIME)


Rob. Vertessy | Director, Cooperative Research Center for Catchment Hydrology, Canberra, ACT, Australia, rob.vertessy@csiro.au

김 상 현 | 부교수, 부산대학교 환경공학과, kimsangh@pusan.ac.kr

1. 서론

과학적인 유역관리를 위한 모형개발 환경의 구현은 구체적인 방법론의 제시와 더불어 중요한 연구과제이다. 모형개발자들은 개발과정이나 표현과정, 모형의 보정 혹은 구동과정에서 빈번히 반복적이고 시간 소모적인 소프트웨어의 작업을 수행해왔다. 이 과정에서 상당한 시간과 자원의 낭비가 발생되었고, 많은 경우 개발된 모형이 일반화되는데 상당한 장애요인이 되었다. 또한 개발자들 상호간의 상이한 개발환경은 모형의 통합이나 상호검증에 긍정적으로 작용하지 못했다. 이와 관련하여 Cooperative Research Center for Catchment Hydrology(CRCCH)의 Catchment Modelling Toolkit Project는 결집력 있는 수문 및 환경모형의 체계를 구축하기 위한 중요한 움직임이다. 이와 같은 과정의 가장 핵심적인 부분은 보다 신속하게 개발되거나 결집될 수 있는 모형환경을 개발하는 것이다.

모형 환경은 새로운 수문 및 환경 관련 모의모형들의 개발을 지원하는 소프트웨어 체제를 의미한다. 이와 같은 모형환경은 운영특성에서 상당한 가변성을 가지고 있음과 동시에 공통적인 요소도 내포하고 있다. 이는 공통적인 자료양식에 대한 지원, 그리고 시각화와 함께 새로운 모형 요소를 위한 보기판(template)을 제공함을 의미한다.

환경모형 개발자들의 다양한 필요성을 우수하게 충족하기 위해 새롭고 획기적인 체계가 만들어졌는데, 이 과정에서 상당수의 기존의 환경모형 체계들이 재평가되고 폭넓게 활용되었다.

2. 모형화 체제(Modelling Framework)

다양한 원리들에 대한 연구와 함께 특정한 영역에 대한 소프트웨어 체계를 구축하는 것은 중요한 연구분야였고 수문 및 환경모형 분야도 활발히 이루어져 왔다(Gamma 등, 1994; Argent 등, 2001). 수문 및 환경 모의를 위한 각종 체제들은 모형의 신속한 개발과 자료처리의 반복적 요소의 통합이나 시각화를 지원하고 있다. 이들 중 일부체계는 모형의 언어들(Reed 등, 1999)이나 시각화 도구들(Maxwell과 Costanza, 1997)에 대한 습관적인(custom) 개발도구를 제공하는 부분까지 발달되었다. Tarsier 같은 체제는 제3자의 개발도구나 언어를 사용하고 있다(Rahman 등, 2003).

이들 다양한 체제는 각기의 장점과 단점을 가지고 있는데, 예를 들면, 특정영역을 위한 언어를 개발하는 것보다 시장에서 구매 가능한 컴파일러를 사용하는 것이 보다 좋은 구동시간을 확보하기가 용이하다. 그러나 습관적인 모형 언어 환경을 사용하면, 메모리의 관리 같은 비교적 중요하지 않는 작업으로부터 모형개발

자들이 자유로울 수 있고 훨씬 더 습관화된 모의경험을 제공한다. 더욱이 습관적인 언어의 인터프리터나, 컴파일러를 개발해서 사용하면 모형코드에 존재하는 변수나 기능들에 기반을 둔 사용자 환경을 고려한 체제의 요소를 구현할 수 있다는 것이다(Rahman 등, 2003). 또한 시장에서 구매 가능한 컴파일러에 대한 경험이 부족한 사용자도 습관화된 모형언어의 체제에 적합한 반면에, 더욱 습관화된 사용자 환경을 개발할 필요성이 있는 개발자들은 상용적으로 개발된 도구들의 체제에 적응하려 하는 경향이 있다.

The Invisible Modelling Environment(TIME)은 상용화된 언어와 습관적인 모형언어 환경을 구동시간에 대한 능동적인 구축이 지원되는 시장에서 구매 가능한 개발환경인 .NET를 사용하여 적절히 절충한 체제이다. .NET은 Visual Basic, Fortran, C++(Meyer, 2001) 같은 다른 언어체제의 기본적인 통합을 지원한다. 각각의 언어체계에 대해서는 .NET의 특정한 컴파일러가 필요하다. 또한 .NET은 구동시간에 대해 요소를 구체화하거나 요소의 특성을 언어와 독립적으로 제공하는 기작을 가지고 있다. 이와 같은 특성은 적용사례 혹은 체제 개발자에 의해 사용되는 관습화된 metadata의 기호(Tag)와 함께 등급구조, 영역, 등급방법 등으로 구성되는 고유특성들을 포함한다. TIME은 사용자 환경의 발생 같은 일련의 다수 작업을 자동화하는 .NET의 metadata 활용능력을 광범위하게 도입하였는데, 이와 같은 기능은 다른 상용의 개발도구에서는 불가능한 작업이다. 공통적인 작업을 자동화함으로써 이들 작업을 직접적으로 모형 코드에 결합시키지 않고, 체제의 진화를 작업을 수행함에 있어서 모형 개발자의 코드관리 작업을 경감시켜준다.

3. TIME

TIME은 모형의 개발에 있어서 몇 가지 핵심적인 단계를 지원하는 새로운 환경모형 체제이다. TIME은 여러 가지 언어 중 하나를 사용하여 높은 정도의 자료가공과 분석, 시각화를 제공하는 일반적인 시험기반에서 모형의 요소를 시험함으로써 새로운 모형 요소를 개발하는 것을 지원한다.

TIME은 다수의 자료처리와 시각화를 위한 재사용 가능한 요소들을 사용하여 시각적으로 화려한 사용자 환경과 높은 습관화 적용모듈의 통합을 지원한다. 또한 래스터(Raster), 시계열(Time Series), 점(Points), 선과 도형(Lines and Polygons), 그리고 절점 연결 네트워크(Node Link Networks) 등의 다수의 중요한 환경모형 자료를 지원한다.

3.1 얼개(Architecture)

개념적으로 TIME은 일련의 성층화된 체계(System)로 나타나며 각각의 층들은 하위층들과 상호작용을 할 수 있도록 구성되어 있다(그림 1). 각각의 층들은 요소들의 군집으로 채워지고, 개개의 군집은 모형개발의 특정한 면을 지원하도록 되어있다. 이들 개개의 군집은 시각화, 입출력, 사용자 선택 관리 등을 다룬다.

핵심적인 구조는 Kernel 층에 포함되어 있는데, 여기에는 여러 가지의 metadata 기호들과 모형들과 자료의 표현을 위한 함축된 상위등급들이 포함된다. Data 층은 입출력 작용과 관련되는 여러 가지 기작과 함께 자료의 표현에 대한 핵심적인 부분을 포함하고 있다. Models 층은 모형 구성과정의 모든 요소를 포함하고 있고 대부분의 개발자들이 요소들을 만들어내는 부분이다. Tools 층은 자료와 모형의 일반적 공정과 자료의 통계나 최적화 공정을 포함하고 있는 부분이다. Visualization과 User Interface 층은 시각화 틀과 각각의 자료 형태를 표현하고 높은 정도의 사용자 편의 환경을 구성하는 부분이다.

3.2 핵심 체제(Core Framework)

Kernel 층은 상당수의 간단한 모형을 개발하기에는 충분한 환경이나, 래스터나 시계열 자료 같은 형태를 직접적으로 처리하지 않는다. 표 1에 나타난 metadata 기호는 요소들의 범위를 구분하는 기호뿐

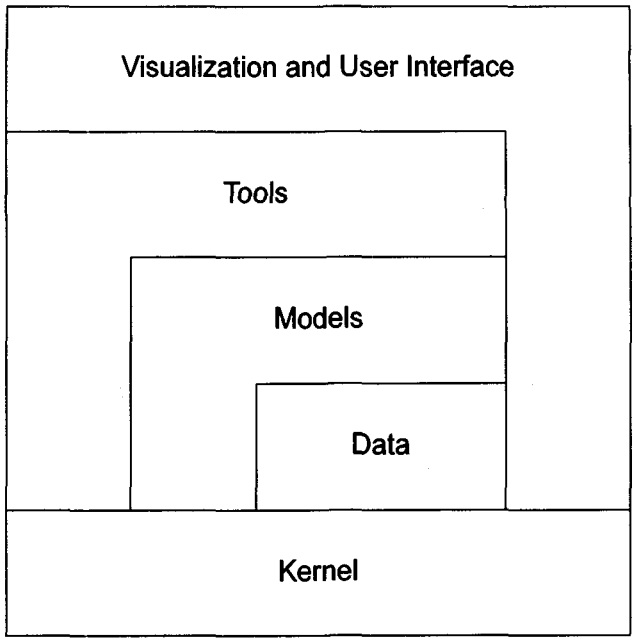

그림 1. TIME의 주요 얼개(Architectural) 층

만 아니라 모형들의 특성을 서술하고 구분하는 기호로 쓰인다.

핵심 체제는 상위 등급인 Model과 Data, 경계 Geometry, 그리고 지원 등급인 Subject와 이에 부응하는 Observer로 구성된다(그림 2). Model은 모든 모형요소의 상위 등급과 모든 하위 등급이 구현되고 함축된 방법인 runTimestep을 포함한다.

Data는 모든 자료 양식의 상위 등급이고, 모든 자료양식을 일차원적으로 접근하고 일반화시키는 방법들을 포함하고 있다. Data의 모든 하위 등급은 공동

표 1. TIME의 Kernel 층에 의해 정의된 Metadata의 기호

Tag	Used To	Applied To
Input	Classify model fields	Fields
Output	Classify model fields	Fields
Parameter	Classify model fields	Fields
State	Classify model fields	Fields
Minimum	Minimum allowable value	Fields
Maximum	Maximum allowable value	Fields
WorksWith	A component works with a particular type of data	Classes
Ignore	Exclude component or field from generic tools	Classes, Fields
Summary	Provide free text description of fields or components	Classes, Fields
CalculationUnits	Specifies the Units that component uses internally	Fields
DisplayUnits	Specifies the Units that should be used to display inputs and results	Fields
UserOption	Flags a field as a default the user can change and maintain across sessions	Fields

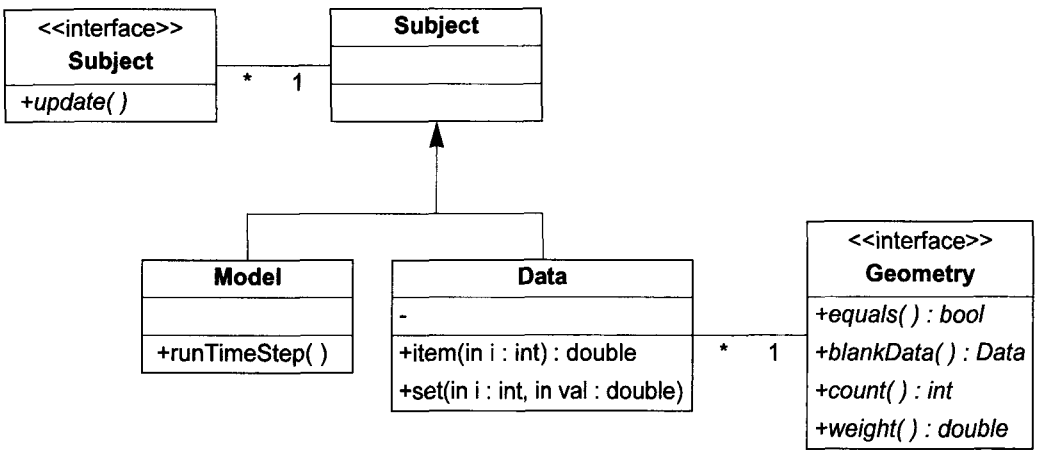

그림 2. Data/Geometry 틀을 포함하는 핵심 체제

적인 사용공간의 확보를 위해 item(in i: int) 부호나 setItem(in I: int, in val: double)을 사용해서 구축되어야 한다. 자료의 사용과 관련된 일차원적인 방법은 통계처리 같은 일반적인 도구들에 의해 처리되며 공간자료나 자료의 천이적인 내용에 관련되어 있지 않다. 하위 등급들에서는 이차원 자료의 색인이나 시계열 자료의 날짜와 시간의 접근 등의 부가적인 접근 방법을 추가하는 것이 자유롭게 이루어질 수 있다.

자료의 목적지에 대한 내용은 Geometry에 의해 정의되며, 가감과 같은 수학적 기능과 상관도 분석 같은 습관적인 방법 등에 적합하도록 공통된 Geometry를 공유한다. 이와 같은 구조는 속성자료가 도형과 같은 공간자료를 효율적으로 표현할 수 있도록 하여준다. 도형들의 형상과 위치는 공통된 Geometry에 의해 나타나고, 속성표의 각각의 열들은 자료 형태로 목적화되어 처리되거나 포괄적으로 다루어진다. 다음은 Data와 Geometry의 구분을 보여 주며, 대부분의 자료 형태들은 다음의 두 개 등급으로 나타난다.

- Geometry 사용자 공간으로 구축되는 등급 : 시간 공간 자료의 내용을 포함한다.
- 자료의 형태에 따라 저장되는 값들을 표현하는 등급 : 일차원적인 접근 방법이나 일반적인 방법 혹은 특정한 방법에 대한 부가적인 접근법을 사용한다.

Model과 Data의 상위 등급인 Subject는 Observers가 모형과 자료의 목적지들이 변화의 통지를 받을 수 있도록 구성되어 있다.

3.3 시각화 구조

그림 3에서 볼 수 있는 시각화 구조는 자료의 시각화에 해당되는 모든 요소의 상위 등급을 포함한다. 이들은 Layer와 ViewDecorator를 포함하고 있는데, 시계열 자료나 분산도안 혹은 공간 지도들도 동일한 기본체계에 의해서 다루어진다.

하위 등급인 Layer는 일정양식의 자료를 Canvas에 표현하고 끌어올 수 있는 논리를 포함하고 있다. Canvas는 하나의 그림판에 복수의 Layer를 중첩시킬 수 있고, 각 층들의 도안을 관리하며, 세계적으로 상용화된 좌표체계를 화상의 좌표체계로 전환을 지원한다. Canvas는 하나 혹은 복수의 ViewDecorator에 의해 꾸며진다. Canvas나 ViewDecorator의 구성은 꾸밈자 설계 양식(Gamma 등, 1994)을 참조하여 만들어졌다.

출력 요소, bitmap 포괄자, 웹 기반 지도 제작도구들 등의 여러 가지 요소가 View의 목적함수들에 사용될 수 있다. 이와 같은 요소들 중 가장 공통적으로 사용되는 요소는 ViewControl인데, 사용자 환경에서 화상을 window에 나타내도록 하는 것이다.

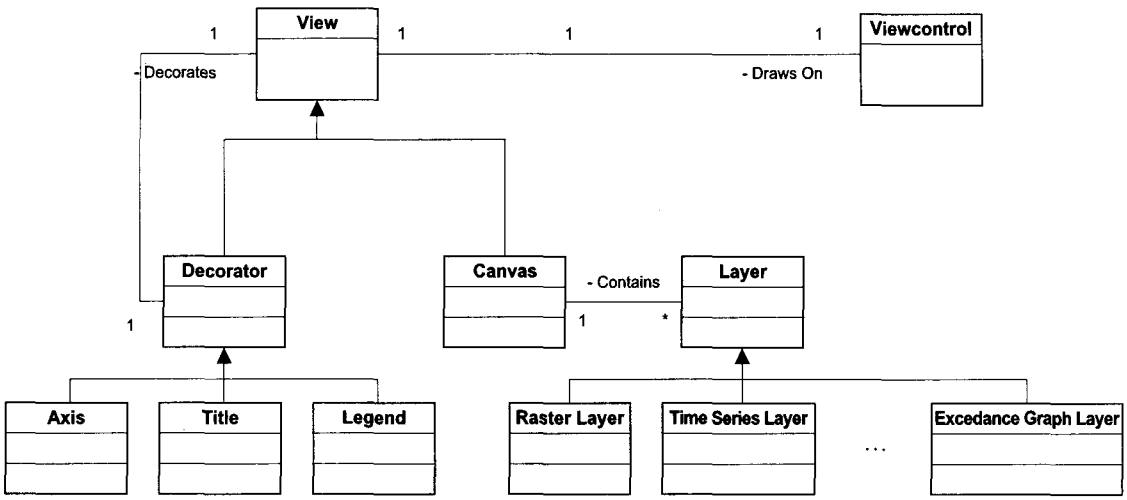

그림 3. TIME 시각화 구조, ViewControl은 단수의 View를 포함하고 있고 Decorator들이나 Canvas에 연결될 수 있다. 단수의 Canvas는 복수의 Layer를 포함할 수 있는데, 각자는 자료를 특정한 방식으로 표현할 수 있다.

보완적인 요소들은 시각화의 사용자 습관화를 위하거나 사용자의 그림과 상호작용(확대, 응답반응, 펼침) 등과 같은 일반적 기능을 제공한다.

3.4 기타 기능들

TIME 코드는 모형의 적용과 관리에 관련되는 몇 가지 조합되어 있는 구조들이 있다. 입출력 보조체계는 자료를 다른 형식의 읽고 쓰는 작용과 관련된 작용을 가지고 있다. User Option 요소들은 Window 주소에서의 선택사항이나 metadata의 기호를 사용하여 사용자에 의한 적용 습관화를 관리하기 위한 일반적 기작을 제공한다. Units 요소들은 통상적인 단위들(예: mm, mgL^{-1})을 표현하고 상호간의 단위에 대한 전환을 수행한다(그림 4). 단위들은 단수의 단위(길이, 질량, 시간, 통화) 혹은 승제연산을 사용한 복합적 단위를 나타낸다.

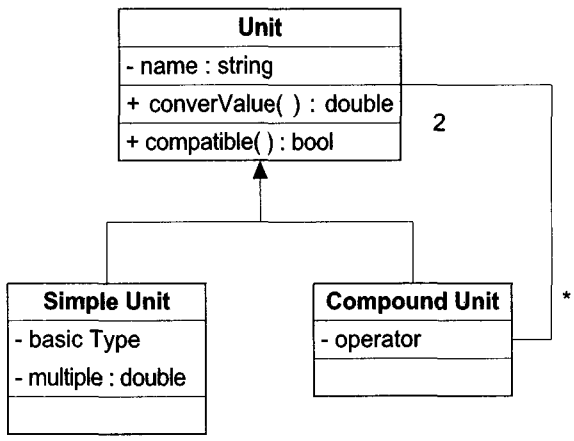

그림 4. Unit들에 대한 TIME의 표현, 단수단위와 복수단위

4. 일반적인 요소들과 도구들

Data, Tools, Visualization, 그리고 User Interface 층은 핵심 구조나 기타 기능들과 함께 다수의 일반적인 요소를 제공한다. 이들 요소들은 적용 개발자들이 모형 요소들을 통합하는데 이용하는 것과 더불어, TIME 개발자들이 모형 요소를 쓰거나 시험하는데 집중적으로 사용된다.

4.1 자료양식과 자료처리과정

TIME은 IO를 포함한 다양한 자료의 형식을 지원하고 포함한다(표 2). 지형분석의 래스터와 같은 특정한 자료처리 공정도구와 더불어, TIME은 어떤 형태의 자료도 다룰 수 있는 다수의 기능을 포함하고 있다. 이들은 수학적 함수, 통계적 기능 및 규칙기반 자료처리엔진을 포함하고 있다.

4.2 가시화

Visualization과 User Interface 층은 가시화 구조 안에 다수의 요소를 포함하고 있는데, 이들은 배열구조의 자료를 제외한 주요 표준자료 표현양식을 변화하면서 표현할 수 있는 요소로 구성되어 있다. 예들들면, 분산도안, 동일 양식의 복수자료 표현(시계열, 래스터 자료), 누가빈도곡선과 흐름지속기간곡선, 그리고 확률밀도그림 등이다. 또한 다양한 종류의 꾸밈자가 사용되는데, 축, 타이틀, legend, 그리고 라벨 등이다. 모든 층과 꾸밈자는 개발자나 사용자가 시각화 과정을 구성하고 장식하는 과정에서 필요한 다수의 선택사항을 제공하고 있다.

4.3 모형처리공정

metadata 환경기반인 TIME의 주요 장점 중 하나는 '모형공정도구'들에 있다(Rahman 등, 2003). 모형공정도구들은 구동시간 중에 새로운 모형에 실시간으로 적응하는 일반적 기능을 제공한다. 이들 도구들은 모형의 입력과 출력, 매개변수를 조사하고 변수의 유효수치 범위 등의 추가적인 정보를 발견한다.

TIME은 몇 가지 최적화 방법과 모형의 사용자 공간을 자동적으로 발생시키는 metadata를 사용하는 광범위한 도구를 포함하고 있다. 사용자 공간 발생자는 사용자들이 입력과 변수, 상태변수, 출력들을 선택하거나 가시화할 수 있는 꾸밈 요소를 제공해 준다.

시간진행모형의 경우 사용자 공간 발생자는 입력 시계열자료를 자동적으로 접목시키거나 출력 시계열자료나 상태변수를 자동적으로 기록하게 해준다. 이 기능은 모형의 개발자들이 시계열 목적함수를 구성하는 노력을 대신 해주는 동시에, 새로운 모형의 요소를 유연하게 유지하면서, 구동시간 동안에 이해와 결정을 수행하게 해준다. 이 부분은 거대한 통합모형의 적용시 광범위하게 사용할 수 있는 특별히 중요한 요소이다. 간단한 모형의 시험에는 각각의 출력과 상태변수

표 2. TIME의 표준자료 양식

Data Type	Summary
Raster	Two-dimensional grid of data, located within a geospatial context
Time Series	Temporal arrangement of data on one of several fixed time steps
Node Link Network	Abstract representation of physical networks, such as river systems
Sites	Collections of points in space
Poly Lines	Linked collection of multi segment lines
Polygons	Collection of closed polygonal regions
Cross Sections	Surveyed, or generated river cross sections
Arrayed Data	Ordered list of values with no spatial or temporal context

를 기록하는 것이 바람직하나, 통합 모형의 경우는 각각 요소에 대한 많은 시계열자료를 저장하는 것은 불필요하며 메모리의 제한성으로 인하여 비현실적이다.

5. TIME의 사용

개발자들은 TIME을 모형 적용사례의 구성과 함께 모형의 요소를 개발하는데도 쓸 수 있다. 모형의 가장 중요한 과학적인 알고리즘은 독자적으로 보호될 수 있으며, 자료처리나 가시화 같은 행정적인 부분은 다른 요소로 분리될 수 있다. 모형 개발자들은 TIME의 일반적인 요소를 모형을 사용하거나 보정하는 시험대로도 사용할 수 있다.

모형 적용사례 또한 단수 혹은 복수의 요소들을 여러 가지 TIME의 요소들을 사용하여 보호할 수 있다. 모형의 요소는 몇 가지 .NET 언어 중 하나를 선택하여 작성될 수 있으며, 모형의 적용사례는 다른 언어로부터 조합될 수 있다.

5.1 모형의 개발

모든 모형은 .NET 언어에서 하나의 등급으로 구성된다. 모형의 등급들은 상위등급인 Model로부터 권한을 이어 받게 되고, runTimeStep()의 함축된 방법으로 설치된다. 부가적으로 모형들은 그들의 입력, 출력, 매개변수들, 상태변수들과 문서를 metadata 기호로 정의한다.

그림 5는 C#으로 만들어진 간단한 ToyModel의 완성된 코드를 보여주고 있다. ToyModel은 두개의 입력자료인 rainfall과 actualET; 상태변수인 netRainfall; 매개변수인 coefficient, 그리고 출력인 runoff로 구성되어 있다. 각각의 변수선언은 적절한 metadata의 기호로 표시되어 있으며, TIME 안에서 모형공정도구에 의해서 사용되고 있다.

모형의 요소를 위한 소스코드는 일반적으로 TIME의 Kernel이나 Data 층을 참조하게 되어있다. 사용자들은 모형을 시험하거나 사용할 때만 일반적인 도구나 시각화 층을 접근하게 된다. 이는 모형의 사용자

```
using System;
using TIME.Core;

public class ToyModel : Model {
 [Input,Minimum(0.0)] double rainfall;
 [Input,Minimum(0.0)] double actualET;
 [State] double netRainfall;
 [Parameter,Minimum(0.0),Maximum(1.0)]
  double coefficient;
 [Output] double runoff;

public override void runTimeStep( ) {
 netRainfall =
  Math.Min(0.0, rainfallactualET );
 runoff = coefficient * netRainfall;
 }
}
```

그림 5. C#으로 구성된 TIME 예제

환경의 발생이나 최적화 과정을 의미한다. 발생된 도구들은 입력자료, 강우자료, 출력자료를 접목시키는 과정을 지원한다.

5.2 모형적용사례의 개발

모형적용 개발자들은 TIME 기반에서의 적용사례를 .NET 언어를 사용한 자료와 모형, 그리고 시각화 도구를 사용하여 개발하였다. Rainfall Runoff Library(Perraud, 2003)는 TIME 환경에서 구성된 주요한 적용사례로서 모형들과 자료, 시각화, 그리고 최적화 도구들을 보다 정교한 사용자 환경에서 구현하였다. 적용사례의 개발은 TIME에 대한 보다 폭넓고 깊은 이해를 필요로 하지만, 수많은 재사용 가능한 요소들은 관련 작업을 상당히 경감시켜준다.

6. 결론

모형구조들은 모형의 개발에 필요한 여러 가지 측

면과 모형의 통합, 자료의 시각화 등을 일반화하거나 단순화한다. 기존의 구조체제들에서 주목할 부분은 시장에서 구매가능한 도구들의 성능과 습관적으로 개발하고 사용해온 언어의 용이성으로 특성화된다. TIME은 시장에서 개발된 환경에 습관적인 언어와 유사한 사용성의 장점을 혼합하여 '두 세계의 최고'를 시도한 새로운 체제이다. 이와 같은 시도는 기존의 분야에서와 동등한 기능성과 함께, 보다 신속한 학습능률과 효과적인 모형개발이 가능하면서 습관적인 모형사례 개발과 견고한 기초를 동시에 만족하게 해준다.

참/고/문/헌

Argent, R.M., Vertessy, R.A., Rahman, J. and Seaton, S.(2001), From components to decisions: The role of software engineering and frameworks in catchment decision support. In: Proceedings of MODSIM 2001, International Congress on Modelling and Simulation, Canberra, December 2001, Volume 4, pp.1589-1594

Gamma, E., Helm, R., Johnson, R. and Vlissides, J.(1994), Design Patterns: elements of reusable object oriented software, Addison Wesley

Maxwell, T. and Costanza, R.(1997), A language for modular spatio-temporal simulation, Ecological Modelling, 103, pp.105-113

Meyer, B.(2001), .NET is Coming. Computer, pp.92-97

Perraud, J.M., Podger, G., Rahman, J. and Vertessy, R.A.(2003), A New Rainfall Runoff Modelling Software Library, Proceedings of MODSIM 2003, Townsville, July

Pree, W. and Koskimies, K.(2000), Framelets Small and Loosely Coupled Frameworks, ACM Computing Surveys, 32, 1es.

Rahman, J.M., Seaton, S.P., Perraud, J-M., Hotham, H., Verrelli, D.I. and Coleman, J.R.(2003), It's TIME for a New Environmental Modelling Framework, Proceedings of MODSIM 2003, Townsville, July

Reed, M., Cuddy, S.M. and Rizzoli, A.E.(1999), A framework for modelling multiple resource management issues - an open modelling approach, Environmental Modelling and Software, 14, pp.503-509