Review of Modelling Frameworks and Environments

Report for Project Task B2

Piotr G Kin, Joel Rahman, Fred G R Watson and R.M. Argent
November, 2000

Synopsis

Project 1.1 of the CRC for Catchment Hydrology has the goal of producing a modelling toolkit for catchment prediction. The “Modelling Toolkit” is one of the foci of the CRC for Catchment Hydrology, and its development is seen as one of the main achievables by the CRC Board. As a second step in the process of identifying and selecting candidate frameworks for testing and development into a pilot toolkit, this paper reviews a number of frameworks that are currently available or undergoing development, and identifies those most likely to be suitable for selection as candidate toolkits.

Cooperative Research Centre for Catchment Hydrology

Program 1: Predicting Catchment Behaviour

Project 1.1: Development of a Catchment Modelling Toolkit
# Table of Contents

TABLE OF CONTENTS .............................................................................................................. II

1 INTRODUCTION ....................................................................................................................... 1

2 FRAMEWORK FEATURES ........................................................................................................ 1

3 MODELS, FRAMEWORKS, TOOLS AND ENVIRONMENTS .................................................. 4

3.1 ITEMS COVERED IN THE REVIEW .................................................................................... 5

4 REVIEW OF MODELLING FRAMEWORKS AND ENVIRONMENTS ...................................... 6

4.1 INTRODUCTION .................................................................................................................. 6

4.2 CME/SME .......................................................................................................................... 7

4.3 DIAS ................................................................................................................................... 8

4.4 EPA MODELS-3 .................................................................................................................. 9

4.5 GENERIC FRAMEWORK WATER (GFW) ......................................................................... 10

4.6 ICMS BUILDER .................................................................................................................. 11

4.7 MMS - MODULAR MODELING SYSTEM ......................................................................... 12

4.8 SMS/WMS ....................................................................................................................... 13

4.9 TARSIER ........................................................................................................................... 14

4.10 OTHER FRAMEWORKS AND TOOLS ............................................................................. 15

4.10.1 BASINS ................................................................................................................... 15

4.10.2 ECOBAS System ...................................................................................................... 15

4.10.3 ISIS ............................................................................................................................ 16

4.10.4 FACET (DIAS FACET) ........................................................................................... 16

4.10.5 Facet (Facet Decision Systems, Inc.) ....................................................................... 17

4.10.6 SPIN ........................................................................................................................ 17

4.10.7 TimeStudio (formerly Hydrol) ................................................................................ 18

4.10.8 NELUP .................................................................................................................... 18

4.10.9 WET-SPA ............................................................................................................... 19

4.10.10 ATHYS .................................................................................................................. 19

4.10.11 OHyMOS .............................................................................................................. 19

5 GENERAL PURPOSE SIMULATION AND ANALYSIS .......................................................... 21

5.1 IDL INTERACTIVE DATA LANGUAGE ............................................................................ 21

5.2 GENERAL PURPOSE SIMULATION ENVIRONMENTS .................................................... 22

5.2.1 The Scientists' Intelligent Graphical Modelling Assistant (SIGMA) ......................... 22

5.2.2 Stella/iThink ............................................................................................................. 22

5.2.3 Extend ....................................................................................................................... 23

6 REFERENCES .......................................................................................................................... 24

7 APPENDIX A CME/SME DETAILS .................................................................................... 25

8 APPENDIX B DIAS DETAILS ............................................................................................... 27

8.1 DIAS BACKGROUND INFORMATION ............................................................................. 27

8.2 ROLE/PURPOSE .............................................................................................................. 27

8.3 AVAILABILITY ............................................................................................................... 28

8.4 MAJOR STRENGTHS AND WEAKNESSES ................................................................ 28

8.5 REASONS FOR FURTHER INVESTIGATION/TRIAL ..................................................... 29

8.6 SELECTED MODELS CREATED WITH DIAS ............................................................... 30

9 APPENDIX C EPA MODELS-3 ........................................................................................... 31

9.1 BACKGROUND INFORMATION ..................................................................................... 31

9.2 ROLE/PURPOSE .............................................................................................................. 32

9.3 AVAILABILITY ............................................................................................................... 32

9.4 MAJOR STRENGTHS AND WEAKNESSES ................................................................ 32
9.5. MODELS-3 CAPACITY TO MEET PROJECT 1.1 CORE REQUIREMENTS .................................................................32
9.6. REASONS FOR FURTHER INVESTIGATION/Trial .........................................................................................32
9.7. MODELS-3 ARCHITECTURE .........................................................................................................................32

10 APPENDIX D ICMS BUILDER DETAILS .................................................................................................35
10.1. ROLE/PURPOSE ..........................................................................................................................................35
10.2. AVAILABILITY ...........................................................................................................................................35
10.3. MAJOR STRENGTHS AND WEAKNESSES ...................................................................................... 35
10.4. REASONS FOR FURTHER INVESTIGATION/Trial........................................................................... 36
10.5. MODELS IMPLEMENTED WITH ICMS BUILDER ............................................................................... 37
10.6. ADVANCED INFORMATION ......................................................................................................................38

11 APPENDIX E MMS DETAILS ..............................................................................................................42
11.1. OVERVIEW ................................................................................................................................................ 42

12 APPENDIX F SMS/WMS DETAILS ........................................................................................................43
12.1. ROLE/PURPOSE ..........................................................................................................................................43
12.2. AVAILABILITY ...........................................................................................................................................44
12.3. MAJOR STRENGTHS AND WEAKNESSES ...................................................................................... 44
12.4. CAPACITY TO MEET PROJECT 1.1 CORE REQUIREMENTS .................................................................... 45
12.5. REASONS FOR FURTHER INVESTIGATION/Trial........................................................................... 45

13 APPENDIX G TARSIER DETAILS .........................................................................................................46
13.1. TARSIER ...................................................................................................................................................46
13.2. OVERVIEW ................................................................................................................................................ 46
13.3. MAJOR STRENGTHS ..................................................................................................................................47
13.4. MAJOR WEAKNESSES .............................................................................................................................. 47
13.5. MINOR ISSUES ........................................................................................................................................... 47
1 Introduction

Previous work under Project 1.1 of the CRC for Catchment Hydrology has highlighted the approach selected to investigate and develop a catchment prediction toolkit. This is a four step process, namely:
1. Develop an outline and specification of the features required and desired of the toolkit (reported in Draft Report B1);
2. Review the relevant modelling frameworks available or under development, to gauge the state and nature of the available options;
3. Develop assessment and selection criteria, and use these to select a number of candidate frameworks for development of a modelling toolkit, and
4. Develop modules within the candidate frameworks, and assess this development with the aim of finally selecting a pilot toolkit.

This report covers the second of the steps listed above.

2 Framework Features

A difficulty in reviewing frameworks and other tools of different designs and capabilities is selection of the features to be listed and reviewed. These features need to be detailed enough to allow a clear understanding of the system attributes, and to allow identification of those frameworks and tools that promise to meet some of the needs of Project 1.1.

For the purposes of specifying a framework for catchment prediction modelling, the following aspects of a modelling toolkit/environment have been identified as necessary (Argent et al., 2000):

- consistent documentation on model specification and model use, including program help files;
- ability to examine, and in some cases edit, data before, during, and after program execution;
- visualisation and statistical analysis of data;
- ability to incorporate stochastic and historical data;
- involvement of users in program and interface design;
- knowledge-based selection of models, when more than one model is available, and
- processes for undertaking and recording the peer reviewing of models.

This is largely a grab-bag of features, covering a range of software engineering, documentation and development processes, but are all features that must be addressed. Further to this, these features are to be considered in the context of supplying different services to meet the needs of different users types. Various users require that a modelling framework provide one or more of the following (Argent et al., 2000):

(a) the capacity to edit, generate, examine, analyse, and visualise data. The data may be primary input data such as spatial data, economic data, or time-series biophysical data. They may also be data obtained from processing of primary data, or they may be results obtained from the running of applications;
(b) the ability to construct new core and support modules;
(c) the ability to combine modules to construct new applications. This may involve combining various core and support modules, and the testing and validation of the application, and
(d) the capacity to run applications multiple times with different parameters, to test alternative management interventions.
To provide a basis for comparison of the frameworks and tools that are available, two levels of review will be undertaken. For all items, a brief overview will be provided, covering the following points:

- **Name**: The name of the tool/ framework/ application
- **Category**: Category or type of tool
- **Architecture**: Brief description of the system architecture
- **Status**: Development period and effort
- **Supported Domains**: Style of modelling supported (e.g., spatial, temporal)
- **Availability**: Type and cost
- **Platforms**
- **Popularity**: Numbers and type of users
- **Developer**
- **Available Model Example/s**
- **References/ sources**
- **P 1.1 Core Requirements**: Level of fulfilment
- **P 1.1 Suitability**: Ability to meet requirements
- **Preliminary Recommendation**

For the most promising tools, framework or applications, a detailed overview is provided in the appendices to this report. These cover some or all of the following, depending upon the availability of information:

- **Role/Purpose of the software**
- **Major Strengths and Weaknesses**
  - Availability
  - Architecture
  - Usability/Performance
  - Capacity to meet Project 1.1 core requirements
  - Ease of Adoption
- **Further Investigation Merits**
- **Valuable Concepts**
- **Framework as Computer Program (SE Perspective)**
  - **Architecture**
    - Syntactical Paradigm
    - Implementation Language
    - Transparency (Model Builder Perspective)
    - Flexibility/Modifiability
    - Longevity/Stability/Robustness
  - **Integration**
    - External Software Tools (viewers, solvers)
    - DBMS capacity
  - **Availability**
    - Source Code (Available, Licence, Proprietary, Other)
    - Collaboration with Authors
    - Object Libraries (Available, Licence, Proprietary, Other)
  - **Documentation**
    - Architecture
    - Source Code
- **Framework as Model Construction Environment (Model Builder Perspective)**
• Architecture
  • OO Paradigm
    • User Defined Classes
    • Inheritance Mechanism
    • Reusability of components
  • Separation of Data from Models
  • MMS (Model Management System)
  • Modelling Language (Extensibility, etc)
  • Model/Algorithm Library
  • “Recursivness” (connected set of models is also a model)
  • Import/Encapsulation Mechanism
  • GUI metaphor (“Drag and Drop”, etc.)
  • Transparency (from Model User Perspective)
• Implementation of Standards
  • Protocols
  • Interfaces e.g., IDL (Interface Definition Language)
  • Documentation
  • GUI
• Documentation
  • Capability: None, Limited, Satisfactory, Full
  • Automation: None, Partial, Full
• Framework as Modelling Environment (Model User Perspective)
  • Stand-Alone Application
  • Model “Views”
  • “Modelling Cycle” Support

The following section covers one conceptual view of the tools, framework and applications that are available to meet some or all the needs of Project 1.1.
3 Models, Frameworks, Tools and Environments

Developments and discussions within Project 1.1 (particularly at the May, 2000 meeting) led to creation of a conceptual framework for the models, frameworks, tools and environments that are available. Figure 1 shows a version of this diagram, with the four main categories being:

- Exploratory data analysis and visualisation tools
- General purpose simulation environments
- Semi-integrated modelling tools (that exchange information through file transfer and database exchange)
- Integrated modelling frameworks

NOTE that these definitions and the examples shown under each one in Figure 1 are based upon limited information and interpretation, and may be changed in light of further investigation.

Figure 1. Conceptual Framework of models, frameworks, tools and environments
3.1. **Items Covered in the Review**

This section provides a list of the frameworks reviewed in the following sections. As noted previously, some (denoted by the trailing asterisks) will be covered in some detail in the appendices.

These include

- GFW (Generic Framework Water) ([http://www.waterland.net/riza/aquest.old/](http://www.waterland.net/riza/aquest.old/))
- ICMS Integrated Catchment Management System *
- Tarsier *

Items covered in less detail will be those that appear to offer less promise, or which only cover particular aspects of the features required. These include:

- BASINS
- ECOBAS Model system
- ISIS
- FACET
- Facet (Decision Systems)
- SPIN
- TimeStudio (formerly Hydrol)
- NELUP
- WET-SPA
- ATHYS
- OHyMOS

An exploratory data analysis approach (IDL) will be presented, for completeness, and to contrast with the frameworks. Some of the general purpose simulation approaches (SIGMA, Stella, Extend) will also be presented, for completeness, and to contrast with the frameworks.
4 Review of Modelling Frameworks and Environments

4.1. Introduction

This review covers the basic information on the models, frameworks, tools and environments listed previously. As noted, the reviews will cover:

- **Name**: The name of the tool/ framework/ application
- **Category**: Category or type of tool
- **Architecture**: Brief description of the system architecture
- **Status**: Development period and effort
- **Supported Domains**: Style of modelling supported (e.g., spatial, temporal)
- **Availability**: Type and cost
- **Platforms**:
- **Popularity**: Numbers and type of users
- **Developer**:
- **Available Model Example/s**:
- **References/ sources**:  
- **P 1.1 Core Requirements**: Level of fulfilment
- **P 1.1 Suitability**: Ability to meet requirements
- **Preliminary Recommendation**:  

KIN ET AL., NOV., 2000
4.2. CME/SME

Basic Information

<table>
<thead>
<tr>
<th>Name</th>
<th>Collaborative Modelling Environment/ Spatial Modelling Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Semi-integrated/ integrated modelling framework</td>
</tr>
<tr>
<td>Architecture</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>Supported Domains</td>
<td></td>
</tr>
<tr>
<td>Availability</td>
<td></td>
</tr>
<tr>
<td>Platforms</td>
<td></td>
</tr>
<tr>
<td>Popularity</td>
<td></td>
</tr>
<tr>
<td>Developer</td>
<td></td>
</tr>
<tr>
<td>Available Model Example/s</td>
<td></td>
</tr>
<tr>
<td>References/ sources</td>
<td><a href="http://iee.umces.edu/~villa/cme/cme.html">http://iee.umces.edu/~villa/cme/cme.html</a></td>
</tr>
</tbody>
</table>

Further Detail to be Provided by Piotr Kin
4.3. DIAS

### Basic Information

<table>
<thead>
<tr>
<th>Name</th>
<th>DIAS (Dynamic Information Architecture System)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Integrated Modelling Framework</td>
</tr>
<tr>
<td>Architecture</td>
<td>Fully extensible OO</td>
</tr>
<tr>
<td>Status</td>
<td>Current, over 7 years of development</td>
</tr>
<tr>
<td>Supported Domains</td>
<td>Basically no limitations; domain is determined by (i) available objects and processes within DIAS, (ii) component models and external applications</td>
</tr>
<tr>
<td>Availability</td>
<td>Unknown</td>
</tr>
<tr>
<td>Platforms</td>
<td>UNIX (Sun Solaris™) and (possibly) Windows NT</td>
</tr>
<tr>
<td>Popularity</td>
<td>Very popular</td>
</tr>
<tr>
<td>Developer</td>
<td>Decision and Information Sciences Division, Argonne National Laboratory, USA</td>
</tr>
<tr>
<td>References/ sources</td>
<td>DIAS Web Pages <a href="http://www.dis.anl.gov/DIAS/index.html">http://www.dis.anl.gov/DIAS/index.html</a>; (Sydelko et al., 1999)</td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
<td>Completely fulfilled</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
<td>Very extensive and complex, requiring considerable effort</td>
</tr>
<tr>
<td>Preliminary Recommendation</td>
<td>Acquire, carefully research, utilise OO architecture concepts, use for implementation of P 1.1 benchmark models</td>
</tr>
</tbody>
</table>

The OO-IDLAMS comprehensive ecosystem-modelling framework is based on IDLAMS (Integrated Dynamic Landscape Analysis and Modelling System) and integrates ecological models and decision support techniques through a geographic information system (GIS)-based framework. It is intended for use in adaptive ecosystem management modelling, and employs DIAS (Dynamic Information Architecture System).
4.4. EPA Models-3

<table>
<thead>
<tr>
<th>Basic information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Category</td>
</tr>
<tr>
<td>Architecture</td>
</tr>
<tr>
<td>Status</td>
</tr>
<tr>
<td>Supported Domains</td>
</tr>
<tr>
<td>Availability</td>
</tr>
<tr>
<td>Platforms</td>
</tr>
<tr>
<td>Popularity</td>
</tr>
<tr>
<td>Developer</td>
</tr>
<tr>
<td>Available Model Example/s</td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
</tr>
<tr>
<td>Preliminary Recommendation</td>
</tr>
</tbody>
</table>
### 4.5. Generic Framework Water (GFW)

#### Basic Information

<table>
<thead>
<tr>
<th>Name</th>
<th>Generic Framework Water</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Integrated Modelling Framework</td>
</tr>
<tr>
<td>Architecture</td>
<td>Component based</td>
</tr>
<tr>
<td></td>
<td>Uses UML and OMG-IDL</td>
</tr>
<tr>
<td>Status</td>
<td>3 years of development; 21 organisational partners, and some 30 staff (at unknown levels of involvement)</td>
</tr>
<tr>
<td>Supported Domains</td>
<td>Primarily spatial and temporal hydrological modelling</td>
</tr>
<tr>
<td></td>
<td>eg 2-D and 3-D modelling of marine and lacustrine systems</td>
</tr>
<tr>
<td>Availability</td>
<td>Unknown</td>
</tr>
<tr>
<td>Platforms</td>
<td>Windows</td>
</tr>
<tr>
<td>Popularity</td>
<td>Unknown</td>
</tr>
<tr>
<td>Developer</td>
<td>Institute for Inland Water Management and Waste Water Treatment (RIZA) (and others), The Netherlands</td>
</tr>
<tr>
<td>Available Model Example/s</td>
<td>Unknown</td>
</tr>
<tr>
<td>References/ sources</td>
<td><a href="http://www.waterland.net/riza/aquest.old">http://www.waterland.net/riza/aquest.old</a></td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
<td>Appear to be fully met</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
<td>Appears suitable</td>
</tr>
<tr>
<td>Preliminary Recommendation</td>
<td>Investigate, and, if possible, adopt as a candidate framework</td>
</tr>
</tbody>
</table>

This framework appears to be what we envisage the greater toolkit project to produce in 3 or so years. The software/ architecture development has been supported by development of:

- Good modelling practice guidelines
- Umbrella agreements on model sharing between groups
- An Open Modelling System
- Software quality procedures
- Data dictionary and model library system
4.6.  ICMS Builder

### Basic Information

<table>
<thead>
<tr>
<th>Name</th>
<th>ICMS Builder (Integrated Catchment Management System)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Advanced Modelling Environment</td>
</tr>
<tr>
<td>Architecture</td>
<td>Object Oriented (OO)</td>
</tr>
<tr>
<td>Status</td>
<td>Current, advanced Beta Version released</td>
</tr>
<tr>
<td>Supported Domains</td>
<td>2D (grided), 1D (lumped) and Time Domain</td>
</tr>
<tr>
<td>Availability</td>
<td>Complete source code (Delphi) and documentation</td>
</tr>
<tr>
<td>Platforms</td>
<td>MS-Windows™</td>
</tr>
<tr>
<td>Popularity</td>
<td>Small user group including researchers and research students, ICAM at ANU</td>
</tr>
<tr>
<td>Developer</td>
<td>Michael Reed, CSIRO Land and Water</td>
</tr>
<tr>
<td>Available Model Example/s</td>
<td>Namoi river system</td>
</tr>
<tr>
<td>References/ sources</td>
<td>ICMS internal notes and LWRRDC fact sheet</td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
<td>Essentially satisfied</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
<td>Good starting point</td>
</tr>
<tr>
<td>Preliminary Recommendation</td>
<td>Merits further investigation/Trial</td>
</tr>
</tbody>
</table>
### 4.7. MMS - Modular Modeling System

**Basic Information**

<table>
<thead>
<tr>
<th>Name</th>
<th>MMS - Modular Modeling System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Semi-integrated modelling tool</td>
</tr>
<tr>
<td>Architecture</td>
<td>Database oriented</td>
</tr>
<tr>
<td>Status</td>
<td>Approx. 6 years of development; Current release: 1.1.7 (1.2 beta also available)</td>
</tr>
<tr>
<td>Supported Domains</td>
<td>Natural resources 1-D and 2-D</td>
</tr>
<tr>
<td>Availability</td>
<td>Source downloadable via internet from <a href="http://wwwbrr.cr.usgs.gov/mms/">http://wwwbrr.cr.usgs.gov/mms/</a></td>
</tr>
<tr>
<td>Platforms</td>
<td>Unix; Currently being ported to Windows</td>
</tr>
<tr>
<td>Popularity</td>
<td>Mainly US Government Agencies (USGS, BoR) and US authorities (TVA, Electrical Power Research Institute)</td>
</tr>
<tr>
<td>Developer</td>
<td>US Geological Survey and Center for Advanced Decision Support for Water and Environmental Systems (CADSWES)</td>
</tr>
<tr>
<td>Available Model Example/s</td>
<td>PRMS TOPMODEL</td>
</tr>
<tr>
<td>References/ sources</td>
<td><a href="http://wwwbrr.cr.usgs.gov/mms/">http://wwwbrr.cr.usgs.gov/mms/</a></td>
</tr>
<tr>
<td>(Leavesley et al., 1996)</td>
<td></td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
<td>Partly met</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
<td>Appears suitable</td>
</tr>
<tr>
<td>Preliminary Recommendation</td>
<td>Investigate further to ascertain: Nature of architecture Planned development</td>
</tr>
</tbody>
</table>

The advantages of this system are robustness and support. Further investigation needs to be undertaken to determine if the architecture supports the type and style of modelling that we plan to undertake.
4.8. SMS/WMS

**Basic Information**

<table>
<thead>
<tr>
<th>Name</th>
<th>SMS/WMS Surface-Water Modelling System/Watershed Modelling System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Integrated Modelling Suites</td>
</tr>
<tr>
<td>Architecture</td>
<td>Unknown; most likely following OO paradigm</td>
</tr>
<tr>
<td>Status</td>
<td>Current; SMS ver. 7.0, WMS ver. 6.0</td>
</tr>
<tr>
<td>Supported Domains</td>
<td>2D Raster (Grid), Time</td>
</tr>
<tr>
<td>Availability</td>
<td>No Source Code, or Libraries (Executables only)</td>
</tr>
<tr>
<td>Platforms</td>
<td>MS-Windows™ 95/98, NT, Unix (X-Windows)</td>
</tr>
<tr>
<td>Popularity</td>
<td>Very popular (a large number of Web references)</td>
</tr>
<tr>
<td>Developer</td>
<td>Environmental Modelling Research Laboratory of Brigham Young University; The Waterways Experiment Station, U.S. Army Corps of Engineers (USACE); FHWA: Federal Highways Administration, Office of Bridge Technology (USA)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Available Model Example/s</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P 1.1 Core Requirements</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
<td>Only as a source of scientific and SE ideas</td>
</tr>
<tr>
<td>Preliminary Recommendation</td>
<td>Further investigation/acquisition justified; both systems might be used as reference standards for P1.1</td>
</tr>
</tbody>
</table>
## 4.9. Tarsier

### Basic Information

<table>
<thead>
<tr>
<th>Name</th>
<th>Tarsier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Integrated Modelling Framework</td>
</tr>
<tr>
<td>Architecture</td>
<td>Data object based</td>
</tr>
<tr>
<td>Status</td>
<td>Under (continuing) development</td>
</tr>
<tr>
<td>Supported Domains</td>
<td>Unlimited: 1-D, 2-D, 3-D, temporeal, cellular automata</td>
</tr>
<tr>
<td>Availability</td>
<td>Source and derivatives probably available</td>
</tr>
<tr>
<td>Platforms</td>
<td>MS-Windows</td>
</tr>
<tr>
<td>Popularity</td>
<td>Small number of users</td>
</tr>
<tr>
<td>Developer</td>
<td>Dr Fred Watson, California State University Monterey Bay, California, USA</td>
</tr>
<tr>
<td>Available Model Example/s</td>
<td>Macaque (large scale hillslopes based water yield model) Uakari (smal scale grid-based water and nitrogen movement) Moo (cellular automaton)</td>
</tr>
<tr>
<td>References/ sources</td>
<td><a href="http://earthsystems.monterey.edu/~fwatson/tarsier.html">http://earthsystems.monterey.edu/~fwatson/tarsier.html</a> (Watson, 1998)</td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
<td>Mostly met</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
<td>Appears suitable, but may require higher user skills that envisioned</td>
</tr>
<tr>
<td>Preliminary Recommendation</td>
<td>Investigate further, adopt as candidate framework, undertake trial</td>
</tr>
</tbody>
</table>
4.10. Other Frameworks and Tools

4.10.1. BASINS

<table>
<thead>
<tr>
<th>Basic Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td><strong>Category</strong></td>
</tr>
<tr>
<td><strong>Architecture</strong></td>
</tr>
<tr>
<td><strong>Status</strong></td>
</tr>
<tr>
<td><strong>Supported Domains</strong></td>
</tr>
<tr>
<td><strong>Availability</strong></td>
</tr>
<tr>
<td><strong>Platforms</strong></td>
</tr>
<tr>
<td><strong>Popularity</strong></td>
</tr>
<tr>
<td><strong>Developer</strong></td>
</tr>
<tr>
<td><strong>Available Model Example/s</strong></td>
</tr>
<tr>
<td><strong>References/ sources</strong></td>
</tr>
<tr>
<td><strong>P 1.1 Core Requirements</strong></td>
</tr>
<tr>
<td><strong>P 1.1 Suitability</strong></td>
</tr>
<tr>
<td><strong>Preliminary Recommendation</strong></td>
</tr>
</tbody>
</table>

4.10.2. ECOBAS System

<table>
<thead>
<tr>
<th>Basic Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td><strong>Category</strong></td>
</tr>
<tr>
<td><strong>Architecture</strong></td>
</tr>
<tr>
<td><strong>Status</strong></td>
</tr>
<tr>
<td><strong>Supported Domains</strong></td>
</tr>
<tr>
<td><strong>Availability</strong></td>
</tr>
<tr>
<td><strong>Platforms</strong></td>
</tr>
<tr>
<td><strong>Popularity</strong></td>
</tr>
<tr>
<td><strong>Developer</strong></td>
</tr>
<tr>
<td><strong>Available Model Example/s</strong></td>
</tr>
<tr>
<td><strong>References/ sources</strong></td>
</tr>
<tr>
<td><strong>P 1.1 Core Requirements</strong></td>
</tr>
<tr>
<td><strong>P 1.1 Suitability</strong></td>
</tr>
<tr>
<td><strong>Preliminary Recommendation</strong></td>
</tr>
</tbody>
</table>
ECOBAS is a system to produce a model documentation which is easily accessible, complete, standardised, comparable and transferable to different applications. Documentation includes not only equations and declaration of used variables but also additional information about the ecological context and ranges of validity. This links all documentation of mathematics with a documentation of the environment, where the mathematical formulation has been found to be valid.

**Overview**

The ECOBAS system has three main parts:

- **ECOBAS_MIF (Model interchange format)** – format/language to document mathematical models. It is a language or syntax definition to formulate complete ecological models in a standardised and exchangeable format
- **ECOBAS-modeling-assistant-tool** – software to setup, check, manage and convert ECOBAS documents. Designed to provide an easy to use interface
- **ECOBAS-world-wide-web Database** – a well known market place to exchange documents

For complete documentation of an object, the characterisation of the ecological environment for which it was created and validated is needed. The ECOBAS_MIF format uses several international standards of classification to provide this information. Conversion of ECOBAS documents are possible in two directions – towards text processing format (at the moment TeX and HTML format) and into model description language (MDL). Some transformations only one-directional eg. STELLA. Transformation from any simulation system to ECOBAS_MIF will be in no case complete – it requires additional information supplied by the user, due to no present simulation system contains all of the information required by ECOBAS_MIF.

### 4.10.3. ISIS

**Basic Information**

<table>
<thead>
<tr>
<th>Name</th>
<th>ISIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Component tools and toolbox</td>
</tr>
<tr>
<td>Architecture</td>
<td>Assume database and DDE</td>
</tr>
<tr>
<td>Status</td>
<td>Fully developed</td>
</tr>
<tr>
<td>Supported Domains</td>
<td>1-D, link-node; possibly 2-D</td>
</tr>
<tr>
<td>Availability</td>
<td>Commercially available</td>
</tr>
<tr>
<td>Platforms</td>
<td>Unknown, assumed MS-Windows</td>
</tr>
<tr>
<td>Popularity</td>
<td>Water resources and flood agencies</td>
</tr>
<tr>
<td>Developer</td>
<td>Wallingford Software Ltd and Halcrow Group Ltd</td>
</tr>
<tr>
<td>Available Model Example/s</td>
<td>Unknown</td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
<td>Limited</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
<td>Limited</td>
</tr>
<tr>
<td>Preliminary</td>
<td>No further investigation</td>
</tr>
</tbody>
</table>

### 4.10.4. FACET (DIAS FACET)

http://www.dis.anl.gov/DIAS/facet.htm
The Framework for Addressing Cooperative Extended Transactions (FACET) is an addition to the DIAS system, which provides further modelling and simulation functionality.

The FACET object class suite provides a flexible architecture for implementing models of the dynamic behaviours of "agents" for which the behaviour tends to involve complex patterns of interaction of multiple entities over time. FACET is particularly well suited for simulation of social processes, either human or animal, and between humans and their organisations and systems.

Recommendation: No further investigation

4.10.5. Facet (Facet Decision Systems, Inc.)

http://www.facet.com

* This is one of many commercial decision support and modelling options that are available.

Facet produces decision support and modelling systems for natural resources management. They provide a range of products, training and services that include:

- Cause&Effect is a decision management portal which automates processes for data assembly, analysis, work-flow, data visualisation, and interaction.
- Frameworks is a management system that organises decision processes for specific technical domains
- The FastTracks methodology manages the process of building decision management systems.

They also offer training, support and consulting services. Facet products are client-server and web-ready, and the user interfaces are Java-based, using Internet browser technology to run over Intranets and the Internet.

Clients have included Golder Associates Ltd., Southern California Edison, Energy Power Research Institute (EPRI), Idaho National Engineering & Environmental Laboratory and University of Lethbridge - Water Resource Institute

Recommendation: No further investigation

4.10.6. SPIN

<table>
<thead>
<tr>
<th><strong>Basic Information</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Category</td>
</tr>
<tr>
<td>Architecture</td>
</tr>
<tr>
<td>Status</td>
</tr>
<tr>
<td>Supported Domains</td>
</tr>
<tr>
<td>Availability</td>
</tr>
<tr>
<td>Platforms</td>
</tr>
<tr>
<td>Popularity</td>
</tr>
<tr>
<td>Developer</td>
</tr>
<tr>
<td>Available Model Example/s</td>
</tr>
<tr>
<td>References/ sources</td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
</tr>
<tr>
<td>---------------------------</td>
</tr>
<tr>
<td>Preliminary Recommendation</td>
</tr>
</tbody>
</table>

### 4.10.7. TimeStudio (formerly Hydrol)

**Basic Information**

<table>
<thead>
<tr>
<th>Name</th>
<th>TimeStudio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Time series data management system and Semi-integrated model toolkit</td>
</tr>
<tr>
<td>Architecture</td>
<td>Component based plus DDE linking</td>
</tr>
<tr>
<td>Status</td>
<td>Fully developed</td>
</tr>
<tr>
<td>Supported Domains</td>
<td>1-D, temporal</td>
</tr>
<tr>
<td>Availability</td>
<td>Commercially available</td>
</tr>
<tr>
<td>Platforms</td>
<td>MS-Windows</td>
</tr>
<tr>
<td>Popularity</td>
<td>Consulting, government agency and authorities</td>
</tr>
<tr>
<td>Developer</td>
<td>Hydro Tasmania</td>
</tr>
<tr>
<td>Available Model Example/s</td>
<td>Catchment management in Italy, New Zealand, Australia and Scotland</td>
</tr>
<tr>
<td>References/ sources</td>
<td><a href="http://www.timestudio.com/">http://www.timestudio.com/</a></td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
<td>To a limited degree</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
<td>Limited (no or minor spatial capability)</td>
</tr>
<tr>
<td>Preliminary Recommendation</td>
<td>No further investigation</td>
</tr>
</tbody>
</table>

### 4.10.8. NELUP

**Basic Information**

<table>
<thead>
<tr>
<th>Name</th>
<th>NERC/ESRC Land Use Programme (NELUP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Semi-integrated model toolkit</td>
</tr>
<tr>
<td>Architecture</td>
<td>Database centric</td>
</tr>
<tr>
<td>Status</td>
<td>2-3 years; Prototype + 2 case study applications</td>
</tr>
<tr>
<td>Supported Domains</td>
<td>Economics, ecology and hydrology - 1-D and 2-D</td>
</tr>
<tr>
<td>Availability</td>
<td>Unknown</td>
</tr>
<tr>
<td>Platforms</td>
<td>Unix (assumed)</td>
</tr>
<tr>
<td>Popularity</td>
<td>UK river management</td>
</tr>
<tr>
<td>Developer</td>
<td>Water Resource Systems Research Laboratory and Centre for Land Use and Water Resources Research, Newcastle, UK</td>
</tr>
<tr>
<td>Available Model Example/s</td>
<td>Rivers Tyne and Cam, UK</td>
</tr>
<tr>
<td>References/ sources</td>
<td><a href="http://www.ncl.ac.uk/wrgi/wrsrl/projects/nelup/nelup.html">http://www.ncl.ac.uk/wrgi/wrsrl/projects/nelup/nelup.html</a></td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
<td>Limited</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
<td>Limited</td>
</tr>
<tr>
<td>Preliminary Recommendation</td>
<td>No further investigation</td>
</tr>
</tbody>
</table>
Further information to be provided by Roger Braddock

### 4.10.9. WET-SPA

<table>
<thead>
<tr>
<th><strong>Name</strong></th>
<th>WET-SPA (Water and Energy Transfers within and between the Soil, Plant and Atmosphere)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Category</strong></td>
<td>Toolbox</td>
</tr>
<tr>
<td><strong>Architecture</strong></td>
<td>Object-oriented modules</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>Unknown (possibly abandoned)</td>
</tr>
<tr>
<td><strong>Supported Domains</strong></td>
<td>Predominantly Soil, Plant and Atmosphere system</td>
</tr>
<tr>
<td><strong>Availability</strong></td>
<td>Unknown</td>
</tr>
<tr>
<td><strong>Platforms</strong></td>
<td>Unix</td>
</tr>
<tr>
<td><strong>Popularity</strong></td>
<td>Limited</td>
</tr>
<tr>
<td><strong>Developer</strong></td>
<td>Laboratory of Hydrology, Free University Brussels</td>
</tr>
<tr>
<td><strong>Available Model Example/s</strong></td>
<td>WET-SPA is mainly a purpose built system for soil, plant and atmosphere interactions using limited, defined modules</td>
</tr>
<tr>
<td><strong>References/ sources</strong></td>
<td>Batelaan et al. 1996</td>
</tr>
<tr>
<td><strong>P 1.1 Core Requirements</strong></td>
<td>Lacks support for identified toolkit uses</td>
</tr>
<tr>
<td><strong>P 1.1 Suitability</strong></td>
<td>Unsuitable</td>
</tr>
<tr>
<td><strong>Preliminary Recommendation</strong></td>
<td>No further investigation</td>
</tr>
</tbody>
</table>

### 4.10.10. ATHYS

| **Name** | ATHYS |
| **Category** | Semi-Integrated Modelling Toolkit |
| **Architecture** | File based exchange between applications |
| **Status** | Unknown (possibly abandoned) |
| **Supported Domains** | Specifically for hydrological applications |
| **Availability** | Unknown |
| **Platforms** | Unix |
| **Popularity** | Limited - nil |
| **Developer** | Laboratoire d'Hydrologie, Montpellier Cedex, France |
| **Available Model Example/s** | MERCEDES spatial gridded surface runoff model has been integrated with the ATHYS system |
| **References/ sources** | Bouvier and Deleaux |
| **P 1.1 Core Requirements** | Lacks support for identified toolkit uses |
| **P 1.1 Suitability** | Unsuitable |
| **Preliminary Recommendation** | No further investigation |

### 4.10.11. OHyMOS
### Basic Information

<table>
<thead>
<tr>
<th>Name</th>
<th>OHyMOS Object-oriented Hydrological Modeling System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Integrated Modelling Framework</td>
</tr>
<tr>
<td>Architecture</td>
<td>Object oriented</td>
</tr>
<tr>
<td>Status</td>
<td>Active. Current version: 3.1</td>
</tr>
<tr>
<td>Supported Domains</td>
<td>1-D and 2-D</td>
</tr>
<tr>
<td>Availability</td>
<td>Download <a href="http://hywr.kuciv.kyoto-u.ac.jp/ohymos.html">http://hywr.kuciv.kyoto-u.ac.jp/ohymos.html</a></td>
</tr>
<tr>
<td>Platforms</td>
<td>Unix; possibly MS-Windows</td>
</tr>
<tr>
<td>Popularity</td>
<td>Japanese agencies and authorities</td>
</tr>
<tr>
<td>Developer</td>
<td>Department of Civil Engineering, Kyoto University</td>
</tr>
<tr>
<td>Available Model Example/s</td>
<td>Unknown</td>
</tr>
<tr>
<td>References/ sources</td>
<td><a href="http://hywr.kuciv.kyoto-u.ac.jp/ohymos.html">http://hywr.kuciv.kyoto-u.ac.jp/ohymos.html</a></td>
</tr>
<tr>
<td></td>
<td>Mailing list: <a href="mailto:ohymos-ML-admin@rdp.dpri.kyoto-u.ac.jp">ohymos-ML-admin@rdp.dpri.kyoto-u.ac.jp</a></td>
</tr>
<tr>
<td>P 1.1 Core Requirements</td>
<td>Possibly</td>
</tr>
<tr>
<td>P 1.1 Suitability</td>
<td>Possibly/ unknown</td>
</tr>
<tr>
<td>Preliminary</td>
<td>Warrants limited further investigation</td>
</tr>
<tr>
<td>Recommendation</td>
<td></td>
</tr>
</tbody>
</table>
5 General Purpose Simulation and Analysis

Exploratory Data Analysis and Visualisation
Available software tools range from simple analysis and visualisation tools through extensive editing and exploration (e.g., spreadsheets, statistical packages) to complex, mathematically orientated (often employing high-level symbolic processing) software development frameworks (e.g., Mathematica, MatLab).

Although applications in this category are often functionally extensive and algorithmically complex, they do not provide native support for module construction. They maintain their essential properties of implementing data analysis, or more generally, data processing.

The most representative examples are collated in Table 1 and one example is briefly characterised in the following section.

Table 1 Exploratory Data Analysis and Visualisation Software Tools.

<table>
<thead>
<tr>
<th>Exploratory Data Analysis and Visualisation</th>
<th>Analysis, Visualisation Tools</th>
<th>Editing, Exploration Packages</th>
<th>Extensible Data Processing Environments</th>
<th>Generalised Data Processing Frameworks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Surfer Grapher</td>
<td></td>
<td>IDL</td>
<td>Mathematica MatLab LabView</td>
<td></td>
</tr>
</tbody>
</table>

5.1. IDL Interactive Data Language
The example is presented to contrast the framework examples presented in Section 4

<table>
<thead>
<tr>
<th>Category</th>
<th>Exploratory Data Analysis and Visualisation Toolkit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Popularity</td>
<td></td>
</tr>
<tr>
<td>Developer</td>
<td>Research Systems</td>
</tr>
<tr>
<td>Status</td>
<td>Fully developed</td>
</tr>
</tbody>
</table>

Major features
• Cross-platform application development
• Fully extensible environment
• Data analysis and visualisation
• Data from any source with flexible data I/O
• Access to ODBC-compliant databases
• Integrated mathematics and statistics
• Object-oriented interactive graphics technology
• Hardware accelerated 2D and 3D graphics of OpenGL
• Image processing (ROI), image analysis routines
• High-level 4GL IDE (Rapid Prototyping and Development)
• Cross-platform GUI development tools
• ActiveX Control tools
5.2. General Purpose Simulation Environments

These examples are presented to contrast the framework examples presented in Section 4.

5.2.1. The Scientists’ Intelligent Graphical Modelling Assistant (SIGMA)

<table>
<thead>
<tr>
<th>Category</th>
<th>General Purpose Simulation Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Popularity</td>
<td></td>
</tr>
<tr>
<td>Developer</td>
<td>NASA</td>
</tr>
<tr>
<td>Status</td>
<td>Legacy</td>
</tr>
</tbody>
</table>

Major features
- Knowledge-based visual IDE for prototyping scientific models,
- AI techniques employed to automate parts of the scientific software development process,
- Specialised GUI; high-level drag and drop data flow diagrams that capture the essential computational dependencies to be expressed in the model
- Missing details of the model specification are automatically filled in from and extensive object-oriented knowledge base containing extensive semantic knowledge about the scientific domain and the modelling equations
- Extends scientific modelling paradigm beyond a set of equations, by including an object-oriented description of the underlying experimental situation (whatever it means – PGK)
- The final product is an executable prototype of a scientific model.

Limitations
- Legacy Application
- Restricted to non-coupled algebraic and first order ordinary differential equations.

5.2.2. Stella/ iThink

<table>
<thead>
<tr>
<th>Category</th>
<th>General Purpose Simulation Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Popularity</td>
<td>Widely used in business applications</td>
</tr>
<tr>
<td></td>
<td>Has been used for underlying models in CME/SME</td>
</tr>
<tr>
<td>Developer</td>
<td>High Performance Systems</td>
</tr>
<tr>
<td>Status</td>
<td>Fully developed</td>
</tr>
</tbody>
</table>

STELLA simplifies model building via a three-step method in which equations that formulate simulations are automatically generated. In the first stage, called mapping, building block icons construct a graphical representation of input parameters. In the second phase, called the model phase, the software automatically creates equations that are needed to simulate a model. The simulation, or third step, can be viewed as graphs, tables, or an animation. STELLA II is a "completely rewritten and extended version of STELLA for Education," introduced in 1987, according to its developers. The software is used in a variety of disciplines in natural science, social science, and humanities at the secondary through postgraduate levels.
5.2.3. **Extend**

<table>
<thead>
<tr>
<th>Category</th>
<th>General Purpose Simulation Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Popularity</td>
<td>Somewhat used in business applications</td>
</tr>
<tr>
<td>Developer</td>
<td>Imagine That Inc.</td>
</tr>
<tr>
<td></td>
<td><a href="http://www.imaginethatinc.com/pages/extend.html">http://www.imaginethatinc.com/pages/extend.html</a></td>
</tr>
<tr>
<td>Status</td>
<td>Fully developed</td>
</tr>
</tbody>
</table>

Extend enables simulation of discrete events, continuous, and combined discrete event/continuous processes and systems. It has a library of pre-built blocks and no programming is necessary but new modules can be developed if desired. Model development is undertaken as follows:

- Place process specific blocks on model worksheet by selecting them from a menu – drag and drop style.
- Connect blocks
- Set parameters in dialog boxes

Data can be entered into dialog boxes or read in from files as the simulation runs.

Features

- Build a model as large as you want
- Track items in your model by assigning attributes
- Import data files, cut and paste to spreadsheets or create a live link and update an Excel spreadsheet as the model runs
- Custom effects by combining the function of several blocks into one through hierarchy or by using an equation editor.
- Build custom blocks in block development environment, which consists of a dialog editor and compiled language. There are also hooks for external languages such as C or Fortran.
- Results of running simulations are provided in numerous forms
- Cross platform compatibility between windows 3.1/95/NT Macintosh, and Power Macintosh systems
6 References


Benz and Knorrenschild, 1997 “Call for common model documentation etiquette”, Ecological Modeling, 97:141-143


Hoch et al., 1998 “Towards a standard for documentation of mathematical models in ecology”, Ecological Modeling, 113:3-12


7 Appendix A CME/SME Details

NOTE: This is taken from a brief review in early 2000, so may be partly out of date.

Role:
The CME is a software package and Graphical User Interface (GUI) that allows users to define and use hierarchies of object types representing the entities involved in the modelling process. The CME can be used to create objects, store them in repositories, and use them in a collaborative context to create arbitrarily complex models and interfaces.

Languages:
The CME uses TCL/Tk as a development language, with some proprietary object-orientated extensions to better integrate the TCL code with underlying C++ implementation. The TCL side of the CME works through a custom object-orientated addition to TCL, integrating C++ with TCL so that C++ objects become TCL commands. The description language of most shared objects uses formal XML grammars. Version 1.0 of the CME was written in C++ and TCL/Tk. Version 2.0 is being written in Java.

Environment:
The CME is a database front-end and GUI for typed objects which incorporate the functionalities of a particular modelling environment. Any modelling paradigm which can be thought of as a hierarchy of objects pertaining to one or more classes, each with its own set of operations, can be represented in the CME.

The main components of the CME are the object factory and the workbench. The workbench is where objects are used, alone or connected to others, to create an interactive, dynamic modelling environment. CME objects can also be connected together on the workbench to create collaborative models whose components come from different network locations, and can be run together on the user’s terminal. The object factory is where objects are created or imported from Network-based repositories, edited, saved and possibly published to benefit other CME users.

As soon as the CME starts, the user is presented with the CME main database window. The main window has two panes that are always present: the database pane and the log pane, where activities of the CME and its environments are logged. The main database screen has four main areas: the class browser, the object browser, the object editor window, and the database controls. The class browser shows the class types available and their hierarchical position. When a user double-clicks on a class name, an editor for an object of that class is created in the right-hand window. Users can create objects by selecting a class, filling in the information for the object in the editor, and saving the object. Objects can be modified, saved, and deleted or there is an option to also revert to the old version if you don’t like the modifications made.

Currently the CME is supplied with a DATA environment, equipped for spatial data. Developers can add new environments to the CME using a simple API. The Graphical User Interface for the Spatial Modelling Environment (SME), a model building environment and tool for spatial simulation, is implemented as an environment in the CME. The SME environment works with the CME DATA environment to provide shared, transparent access to data between simulations, external applications, and other environments in the CME. In addition to the graphical operation and the greater ease of use, the SME interface provides ways to amplify the SME functionalities by integrating user-written interpreted code in external functions, and by providing a gateway between the SME and external packages such as geographical information system and visualisation programs. To use the SME GUI it must be configured and compiled, then the SME environment must be installed in the CME.

The SME environment in the CME defines six classes: Project, Model, Configuration, Simulation, User Variable, and User Event. Objects of these classes can be created and modified using the editors provided by the SME GUI. These classes have a hierarchical structuring. To create any one of these objects, the user double clicks the corresponding class icon in the CME.
class browser to make the corresponding object editor appear in the CME editor window. After using the editor to define the properties of a new object, it can be saved in the CME object database by clicking the Create button.

**Flexibility/Modifiability:** Developers can create and add new environments to the CME using a simple API. Objects can be modified easily and there is no enforcement of any particular object meaning in the CME. Objects can be arranged the way the environment implementor defines them.

**Versatility:** The CME has also been developed as an interface for the SME and new environments are easily added using an API. Currently the CME is supplied with a DATA environment geared to spatial data. Data objects can be grouped together in Data Collections. Three basic data types are supported, and more can be added by means of the API. The data editor supports different visualization modes, which are selected accordingly to the type of object that is loaded into it, and can be modified by the user. Data can be shown as tables of values, histograms of values, line graphs, raster maps, three-dimensional wire-frame models, or three-dimensional surface models, with a choice of visualization options including changes in perspective and visual effects.

**Documentation:** There is a user’s guide titled ‘Usage of the Collaborative Modelling Environment’ at http://iee.umces.edu/~villa/cme/cme.html. This is not very comprehensive on how to use CME and is still not fully developed as a web site. It gives a brief introduction to Installing the CME, Starting the CME, Installing and updating environments and other aspects of the CME but is not helpful for detailed working or help.

**Ease Of Adoption:** The CME has been designed to be useable without any commercial software packages. Before installing the CME, a few pieces of software need to be installed. All the required and optional support packages are available at the CME ftp site under the ‘support directory’. All the packages used with the CME are freely distributable for non-profit use, well tested and widely used. The CME is known to compile fine on SUN workstations running Solaris 1 and 2 and Linux PC’s.
Details of installing CME are available at: http://iee.umces.edu/~villa/cme/node3.html
8 Appendix B DIAS Details

8.1. DIAS Background information

There is a significant degree of confusion about the fundamental nature of DIAS (Dynamic Information Architecture System). The system is generally considered to be complex yet flexible, fully extensible yet already extensive, object oriented (OO) yet complicated and difficult to learn. And, since it is known that it works only on Sun Solaris™, many assume that DIAS is a stand-alone software application that implements an advanced general-purpose modelling framework/environment. The Argonne National Laboratory DIAS Web Pages and other published information do not seem to alleviate the confusion.

The Dynamic Information Architecture System (DIAS) provides a range of functions, depending on the context in which it is used. Essentially, there are the following definitions, or descriptions, of the system (http://www.dis.anl.gov/DIAS/examples.htm, http://www.dis.anl.gov/idlams/building.htm):

- DIAS is an abstract OO architecture concept.
- DIAS is a generic, object-oriented (OO) architecture.
- DIAS is an advanced general-purpose modelling framework.

The apparent incongruence in the perception of the "true" nature of DIAS disappears when one realises that the DIAS was developed over the last seven years. Also not surprising is the analogy between the early development of the P1.1 conceptual requirements and the first two bullet points in the previous paragraph that undoubtedly describe only some aspects of DIAS.

Since 1992/1993, the Decision and Information Sciences Division of Argonne National Laboratory has started to develop an abstract methodology that was solely dedicated to resolving problems arising from attempts at integration of existing, legacy-type models and other applications. The Object Oriented system analysis and software design methodologies, in spite of being at the very early stages of their evolution, were adopted as the most capable and most promising technologies able to address the complexity of the problem.

Early documentation suggests that DIAS was initiated and then shaped by DEEM (Dynamic Environmental Effects Model). Over the next several years, ANL utilised the DIAS concept to implement a large number of models in such disparate disciplines as (i) the simulation of environmental effects on battlefield conditions, (ii) natural resources planning and ecosystem management, (iii) multi-aspect healthcare delivery management; and (iv) simulation of wind-generated wave transition processes in a marine environment. Presently, DIAS has reached a mature stage and developed into an advanced general-purpose modelling framework.

Recently, in 1998, DIAS framework was used to redesign and reimplement IDLAMS, (Integrated Dynamic Landscape Analysis and Modelling System). The resulting new system, OO-IDLAMS, fully demonstrates the advantages of object-oriented architecture approach. In addition, it provides an adaptable decision support system for environmental managers.

8.2. Role/Purpose

DIAS (Dynamic Information Architecture System) is, by far, the most advanced, sophisticated, general-purpose integrated modelling environment addressed in this review. It appears to be the only framework/environment that fully deserves the general-purpose classification. For the last seven years, the DIS division of Argonne Lab has been developing the object-oriented framework for a specific purpose of addressing complex "real-world" modelling and simulation problems, and to offer an advanced support for the decision-making processes. Careful and detailed conceptual design of the object-oriented (OO) software model ensued the extraordinary
flexibility and adaptability of the resulting software infrastructure of the Dynamic Information Architecture System (DIAS).

Figure B.1 DIAS System

DIAS facilitates holistic management of information processes, including the construction of simulations from component modules according to a user-supplied context. The domain of DIAS is flexible, determined by the objects available within DIAS and by the collection of models and other data processing applications. This implies that DIAS is easily re-configurable to different applications, subject domains, and user groups.

8.3. Availability

Presently, the availability is unknown, both as an executable environment and as a source code. However, Argonne National Laboratory is one of the U.S. Department of Energy's largest research centres. Thus, the code might be freely available, or at least, available to CSIRO as the non-profit, scientific research organisation of the Australian Government. This needs clarification before we decide to proceed any further.

8.4. Major Strengths and Weaknesses

Architecture/Usability/Versatility/Performance Strengths

• Seamless fusion of object-oriented (OO) infrastructure classes and the flexibility and dynamics of the system.
• Extreme flexibility and adaptability of the architecture.
• Capability to construct models:
  • with user-defined level of detail that reflects the dynamics of “real-world” systems at the required conceptual scale,
  • with complex, cooperative behaviours of agents (e.g., Persons, Organizations) to perform multidisciplinary simulations via full support for combining and integrating disparate models and other applications,
  • by integrating legacy models without extensive reworking, Reusability through capacity to develop object (class) libraries with the objects representing “real-world” entities,
• Simultaneous support for models and external applications operating at multiple spatial and temporal scales,
Specifically designed to accept a wide variety of data types and formats\(^1\)
- Cooperative distributed execution of models and applications on multiple processors linked across computer networks,
- Provisions to incorporate new data formats, concepts, and decision-making technologies.

**Architecture/Usability/Versatility/Performance Weaknesses**
- Single OS platform (UNIX; Sun Solaris), although MS-Windows has been mentioned as a platform.
- Programming languages (C++, SmallTalk, etc.) must be used for a DIAS model integration,
- Extreme complexity of DIAS; need for a substantial time and resource investment (e.g., highly experienced SEs) for full technical utilisation of the system.

**DIAS Capacity to meet Project 1.1 core requirements**

As far as construction and integration of models is considered, the DIAS environment fulfils almost all requirements of Project 1.1 with a substantial surplus of features (Sections 1.2-1.5 of Task B1 Report). Moreover, in many instances, it goes beyond the scope of the requirements, by providing generalised mechanisms of which the P1.1 requirements are just special cases.

The DIAS framework;
- Relies heavily on modular architecture;
- Provides all the required SE operations on modules (construction, modification, extension, etc.);
- Includes "Formal Registration" mechanism; a semiautomatic technology for incorporating external models and other applications to make them indistinguishable from the native objects;
- Has extensive GUI customisation capability, and
- Ensures that new ideas or new requirements (e.g., Model Representation Language, automatic model analysis, version control system, support for scenario simulation/multiobjective analysis, etc.) can be easily implemented as separate modules and incorporated into the environment.

However, there are few minor exceptions, especially regarding the system level requirements, such as:
- Availability of source code (or at least binaries) of DIAS framework and class libraries
- Platform dependency (Sun Solaris),

It is unlikely to conform to the Open Source Definition, (http://www.opensource.org/osd.html).

**8.5. Reasons for Further Investigation/Trial**

It appears that DIAS is the only meta-framework available, in that it is possible to create the P1.1 framework by using DIAS as an intermediate modelling/programming environment. One of the possible methods for this would be to define a Model Representation Language within DIAS using FACET (See Section 4.10.4) then “constrict” DIAS for the DIAS-Toolkit end users to allow them to create models that pertain only to the Integrated Catchment Management concept. In other words, it should be possible to encapsulate DIAS within a shell that is restricted to integrated catchment management problems only, thus it is conceptually straightforward and simple to use as a framework/toolkit for creating stand-alone catchment models.

---

\(^1\) DIAS accepts terrain elevation and surface features in DMA, USGS, and TIGER formats; interpreted remote sensing imagery; CAD-type data of 3-D objects; weather data in World Meteorological Organization formats, and data from most commercial DBMS.
Over the years, a large number of DIAS extensive class (object) libraries pertaining to the environmental (including hydrologic) modelling and decision support systems were developed within many disparate projects, a majority of which were conducted by Argonne National Laboratories. If those libraries can be made available to us (preferably as source code), they should be investigated very carefully. It is apparent that they contain solutions to many Software Engineering problems the P1.1 is yet to face.

8.6. Selected Models Created with DIAS

Dynamic Environmental Effects Model (DEEM)
The DEEM application was the first instantiation of DIAS. DEEM was developed to provide an architecture for multidisciplinary modelling of terrestrial, aquatic, and atmospheric processes to simulate environmental effects such as rainfall, soil moisture, effect of clouds on line-of-sight, trafficability and general Intelligence Preparation of the Battlefield (IPB).

Healthcare Management Simulator
This DIAS application was developed to simulate all major aspects of healthcare delivery-clinical/physiological, procedural, logistical, and financial, at a level of detail appropriate to the user needs.

Integrated Ocean Software Architecture
This application of DIAS is an object-based virtual maritime environment within which existing models are employed to simulate the transition of wind generated waves in the deep water to waves in the near shore environment, then to surf height and currents. The application allows for the use of several different model combinations according to the complexity of the shoreline and near shore sea bottoms.

Object-Oriented Integrated Dynamic Landscape Analysis and Modelling System (OO-IDLAMS)
OO-IDLAMS is a prototype DIAS application developed to demonstrate the advantages of this type of object-oriented architecture approach to providing military environmental managers and decision-makers with a strategic, adaptive approach to integrated natural resources planning and ecosystem management.
9 Appendix C EPA Models-3

9.1. Background information
The concept of an integrated modelling and analysis framework, Models-3, was formulated by U. S. Environmental Protection Agency primarily to facilitate coordination between many disparate air quality models and other software tools. The secondary aim was to introduce, to other federal agencies, academia, and research institutions, a unified and holistic approach to multi-discipline environmental modelling.

Because of the vast extent and considerable complexity of environmental domain, EPA limited the initial Models-3 system development scope to air quality modelling. Thus, the goals for the Models-3 modelling framework were to improve for:

1. Natural resource managers: evaluation methodologies for assessing the impact of air quality management practices for multiple pollutants at multiple scales,
2. Scientists: means of investigation, understanding, and simulation of chemical and physical interactions in the atmosphere.

These two groups (model user and developer) have diverging requirements and expectations from a modelling framework. However, there are significant advantages in using the same problem-solving environment. Thus, Models-3 is intended to serve as a community foundation for the widespread application of air quality models and for their continued scientific advancement. Another goal of the Models-3 program was rapid transfer of scientific ideas and methodologies to everyday users, especially those people in state and local governments, industry, and academia.
9.2. Role/Purpose
Models-3 is not a single model or modelling system, but rather it is a problem-solving environment containing components that help you build, evaluate, and apply air quality models. The initial version of Models-3 contained a Community Multiscale Air Quality (CMAQ) modelling system for urban to regional scale air quality simulation of tropospheric ozone, acid deposition, visibility and fine particulate.
Presently, Models-3 and CMAQ in combination form a powerful third generation air quality modelling and assessment system that can treat multiple pollutants simultaneously up to continental scales and incorporate feedbacks between chemical and meteorological components. Future efforts toward fourth generation systems will extend linkages and process feedback to include air, water, land, and biota to simulate the transport and fate of chemical and nutrients throughout an ecosystem.

9.3. Availability
The Models-3/CMAQ modelling environment and air quality model is available free of charge from EPA ftp server. However, other necessary software packages are not distributed by EPA and have to be purchased separately.

9.4. Major Strengths and Weaknesses

Architecture/Usability/Versatility/Performance Strengths
- Full source code for Models-3 framework and component models comprising CMAQ (but only for the Solaris platform),
- Capacity for Models-3 framework to create stand-alone models

Architecture/Usability/Versatility/Performance Weaknesses
- No other model than CMAQ has been implemented to date,
- Source code of Models-3 is written in FORTRAN77 and FORTRAN90
- Models-3 employs outdated software design methodology and does not adhere to OO paradigm,
- Models to be incorporated into the framework must be written in FORTRAN77/90,
- Commercial Windows-NT™ version from NTIS probably does not contain the source codes for Models-3 or CMAQ,
- Requires prohibitively expensive commercial software such as SAS (The Statistical Analysis System) (US$2,500 to $34,000), Arc/Info (CSIRO licence costs), IBM Visualization Data Explorer (US$ free?).

9.5. Models-3 capacity to meet Project 1.1 core requirements
Although Models-3 appears to be integrated modelling and analysis framework it appears to be so tightly integrated with the three types of environmental modelling systems meteorological, emission, and chemistry transport that is appears doubtful that it could be used for creating integrated catchment management models.

9.6. Reasons for Further Investigation/Trial
Possibly none. Outdated information technology. Scientific areas not related to P1.1. There is some pretty advanced visualisation technology in Models-3 Visualisation Subsystem possibly worth investigating for concepts and ideas.

9.7. Models-3 Architecture
There are ten major components of Models-3 framework accessed through the Models-3 main window. These are:

**Program manager**
It allows the user to perform various functions on executable programs or UNIX shell scripts to integrate them with and make them available to the other components of the Models-3 environment. The main functions are (i) registration, (ii) update and (iii) search; with registration being by far the most important one. The registration process involves interactive entering of the following information about the program/script: (i) descriptive model documentation, (ii) I/O requirements and specifications, (iii) the target front-end hardware platform, the host OS, and run-time environment variables. Registered entities are available to the Study Planner Models-3 framework component for designing extensive scenarios, exploring various simulation configurations including those, in which previous results are used as inputs. Unregistered programs can be also used, however configurations for typical modelling domains are pre-registered within the Models-3 system to make the development easier for the typical user.

**Source Code Manager**
The Source Code Manager provides advance functions for source code management of component models. Within the Models-3 framework, the user can retrieve, edit/modify any part of any source code file. The manager incorporates an advanced version control technology keeping track of the updates and making them available, together with the latest version, to the user and the FORTRAN compiler.

**Science Manager**
The Science Manager offers partial functionality of the Model Manager, as understood in today's AME (Advanced Modelling Environments) field. It allows specifying global information shared between all models in Models-3 environment, such as spatial co-ordinates, mapping projections or, in case of CMAQ, chemical reaction mechanisms, new chemical species, etc. In other words, it allows for substitution of alternative component sub-models, as long as they conform to the information exchange mechanism within the Science Manager. However, it appears that the user has to ensure the sub-model compliance.

**Model Builder**
It appears again that the Model Builder offers the functionality of the AME Model Manager paradigm. It complements the functionality of the science manager by configuring a model for execution in a different location, within a different coordinate system, selection of the 3D grid resolution, interchange of sub-models, etc. It also assists with preparation of new models that explore alternative configurations of new and existing science components.

**File Converter**
It is an old style, parser-like file conversion utility with limited capacity regarding file formats and requiring user-supplied configuration information.

**Dataset Manager**
The Data Set manager follows the Federal Geospatial Metadata Standard for input data format definition. The physical location of the meta-data and data files needs to be supplied by the user. The Dataset Manager also provides limited capacity for data examination and sub-set search.

**Strategy Manager**
In part, the Strategy Manager is a functional equivalent of a moderately advanced scenario designer and supervisor. Among other things, it allows performing an automated sensitivity analysis, determining the relative effectiveness of base scenarios, and, in general allows for setting up scenarios according to user specifications.
In addition, the Strategy Manager has built-in predictive capacities based on EPA’s Multiple Projection System. On the basis of past data, the Strategy Manager can estimate future emission of air pollutants, such as carbon monoxide, nitrogen oxide, particulate matter, sulphur dioxide, and volatile organic compounds.

**Tools Manager**
The tools manager, as the name indicates, is used for administration of various external programs and software tools; once a tool is registered within the Models-3 framework it is available for use. The tools that are accessible are Vis5D, Text Editor, Models-3 Emission Processing and Projection System (MEPPS), Package for Analysis and Visualization of Environmental Data (PAVE), Statistical Analysis System (SAS™), ARCIInfo™, Inventory Data Analyzer (IDA), and VisDriver.

Since the framework requires ARCIInfo™ and SAS™ licences to function, they have to be purchased separately, as they are not supplied with Models-3.

**Study Planner**
The Study Planner allows the user to interactively (using "Drag and Drop" GUI) create models from the sub-models and other entities available within Models-3 framework. The model creation process partially follows the object-oriented paradigm since it attempts, although not entirely successfully, to separate models as abstract entities from their instantiating data. The fully defined and data configured models can be saved as physical files for future access. The files can be opened within the Models-3 framework, edited, modified for different data sets and even exchanged between users.

**Framework Administrator**
This component allows the Models-3 framework administrator to perform various, obscure, and loosely related OS dependent functions. For more information, please refer to the Models-3 installation guide (http://www.epa.gov/asmdnerl/models3/doc/install/install.html) and Framework Administrator Models-3 architecture component (http://www.epa.gov/asmdnerl/models3/doc/overview4.html).
10 Appendix D ICMS Builder Details

Figure D.1 provides a conceptual view of the ICMS components

**ICMS Components**

10.1. Role/Purpose

ICMS Builder is an integrated environment for creating, linking and executing advanced spatial models. It has “evolved” from a conceptually rudimentary spatial discharge model (Catchment Management Support System, CMSS) into a complex development environment that, although almost a general-purpose, it is still somewhat biased towards surface runoff, discharge and routing processes within a catchment. In fact, during the recent workshop, Mike Reed, the ICMS Builder creator, has confirmed, “ICMS Builder was specifically targeted at environmental management and developed to rapidly get science to the catchment managers.”

The primary focus of ICMS is on model construction and integration. Nevertheless, significant infrastructure for controlling the execution of models (e.g., sensitivity analysis, scenario investigation), analysis and visualisation of results and input data is also provided.

10.2. Availability

ICMS Builder has been developed by Michael Reed, CSIRO Land and Water. The source code and other related materials, including documentation, a large number of papers, and Web based tutorial, are owned by CLW and freely available to the Project 1.1.

10.3. Major Strengths and Weaknesses

Architecture/Usability/Versatility/Performance Strengths
• OME (Open Modelling Engine) implementation,
• OO (Object-Oriented) design and OO implementation language (Borland Delphi),
• MickL: integrated, simple to use Model Representations Language (MRL) for model definition (MickL is based on C syntax),
• “Parent-child” hierarchical relationship of models, i.e., fully formed models can be used as components to create more complex models,
• Good support for spatial models with raster domains,
• Standard algorithm/function libraries for MickL (e.g., mathematical functions, sort, linear programming)
• Libraries of reusable (ICMS native) models,
• Visual “Drag and Drop” interface for constructing, linking and integrating models.

Architecture/Usability/Versatility/Performance Weaknesses
• Limited capacity of implementing new data types within the ICMS infrastructure,
• No provision for creating new data types in MickL,
• Algorithm/function libraries for MickL cannot be extended without changes to the ICMS source code,
• Only “native” models are supported; no import/encapsulation mechanism for external models,
• Limited capacity of creating customised user interfaces,
• Slow execution of models, even with only moderately large sets of data,
• Single environment for construction and execution of models (no provision for generating “stand-alone” applications).

Capacity to meet Project 1.1 core requirements
ICMS Builder fulfils most of the basic requirements of Project 1.1 (Sections 1.1-1.5 of Task B1 Report) from the software engineering perspective and as a model development environment:
• Design and implementation follows the object-oriented (OO) paradigm (“module based system”),
• Source code (in Borland Delphi) is freely available (“Open Source”),
• Proprietary (Borland Delphi and others) component libraries are royalties free,
• Software modules can be added and modified at the source code level,
• MickL language provides an advanced sub-model (“model module”) definition mechanism,
• “Intelligent” compilation of sub-models; the MickL compiler makes semantic inferences about the sub-models (partial “automatic analysis”) providing templates for data and a set of enforceable rules for meaningful linking of sub-models (limited “safety of use”),
• Built-In classes to handle spatial and temporal data; ICMS understands the data formats and GUI knows how to display them
• ICMS provides extensive graphical infrastructure for sub-model linking and integration,
• ICMS implements the Open Modelling Engine (OME) standard,
• Supports binaries created with any programming language that adheres to the MS-Windows™ standards for intercommunication between executable objects.

Conversely, some of the limitations include:
• MS-Windows™ platform only (“platform dependent”),
• External existing models cannot be used (“wrapper facility presently not available”),
• MickL is a “Typeless” or “Dynamically Typed” language that defies the “safety of use” requirement; it is inadvertently easy to design a connection between two objects that resolve into incompatible types generating a run time error,
• No version control system,
• No stand-alone applications and, as yet, no support for customised model interfaces (“Custom Views”)

10.4. Reasons for Further Investigation/Trial.
Many deficiencies of ICMS Builder have been addressed and are currently being implemented. The new version (2.0), to be released in May 2001, contains the following new modules and improvements;

- **ICMS Scenario;** full support for scenarios/multiobjective simulations,
- **ICMS View;** model interfaces tools should provide “Custom Views”, especially to create tailored DSS-type GUIs (in prototype stage),
- **ICMS Tools;** data processing tools (in implementation stage),
- New memory model should significantly improve execution speed.

The following features of the MickL modelling language play a significant role in making it into the major strength of ICMS Builder:

- Syntactical and semantic simplicity,
- Tight, expert integration with the modelling environment,
- Capacity to make “intelligent” inferences about models,
- Built-in support for manipulating spatial (raster based) data sets.

It appears necessary to investigate MickL further in order to determine the feasibility of how (and if) the following features could be added to the language without sacrificing its present functionality:

- Ability to create structured data types (perhaps even OO classes),
- Explicit declaration of every variable (even if it is a “void” type),

Also, since ICMS follows OO paradigm, it is worth investigating how much of it is reusable as class libraries and patterns (or as source code, as a last resort).

### 10.5. Models Implemented with ICMS Builder

Over the last two years, the Integrated Catchment Assessment And Management Centre (ICAM) of ANU has been cooperating with CSIRO Land and Water in the development of ICMS. The iCAM Centre has developed extensive ICMS model libraries that were subsequently used in development of models for the following catchments;

- Namoi
- Upper Murrumbidgee
- Yass

The Namoi Catchment ICMS model addresses, amongst others, water quality, water allocation, ecological and socio-economic management issues. This model is an excellent example of the framework capabilities and, in particular, the capacity for harmonious interaction of diverse component models. These component models are stored in ICMS Model Libraries and can be reused for development of models for other catchments. The following component models were developed by ICAM and used in the resulting Namoi DSS;

- Rainfall-runoff model capable of simulating the effects of vegetation change and the influence of farm dams on flows,
- Routing model component for flow between catchment nodes (implementing the Muskingham procedure),
- Inverse problem model for estimation of Muskingham model parameters from known flow gauge data,
- Sediment and phosphorus transport models,
- Disaggregation technique for predicting flow in ungauged catchments,
- Effects of storm event intensity on sediment concentration model,
- Radiation prediction model and simulation of resulting lake evaporation,
- Ecological flow indicators analysis module,
• Socio-economic model for management of water allocation issues (uses linear programming for economic optimisation.)

10.6. Advanced information

Framework as Computer Program (Software Engineering Perspective)

Architecture
ICMS is an integrated environment that incorporates the following conceptual components:
• Open Modelling Engine (OME),
• Support for models with spatial grid based (2D), lumped (1D) and time domains,
• Syntax aware editor, and fully featured compiler for the MickL, the Model Representation Language (MRL) that was specifically designed for OME and programming simplicity,
• User-transparent “semantic investigator” that facilitates “intelligent reasoning” about a MickL sub-model representation in order to determine the structure of its containing class and permissible ways the sub-model might interact with other entities,
• Model Manager for integration (linking) of sub-models,
• Testing provisions for sub-models and assembled models,
• Run Manager for model execution (including support for scenario investigation),
• Advanced interactive GUI including a rich set of tools for I/O data visualisation (maps, graphs, etc.) and data analysis.

All ICMS components are housed in one application and adhere to the object oriented (OO) software design and construction paradigm.

ICMS Builder is implemented on MS-Windows™ platform in Borland Delphi language that closely follows the object oriented paradigm. Compatible class libraries are very extensive and the binaries can be distributed for-profit, royalties free, together with ICMS executables. Also, other languages (e.g., C++, VBA) that follow the MS-Windows™ protocols for information exchange between executable objects (ActiveX, DDE; Dynamic Data Exchange, DLL; Dynamic Link Libraries, MSMQ; MS Message Queuing) can be used for further development.

Transparency
Transparency of the software implementation is another major advantage from the perspective of the model developer, undoubtedly the most extensive user of the ICMS environment. The model builder need not know about the implementation details, in other words, there are very few “tricks and tips” related to the ICMS environment that don’t directly pertain to the model development activity.

Documentation
At present (November 2000), it is not known how recent is the documentation of ICMS Builder architecture; it is likely that the new version, with the improved memory management and “Custom Views”, is not well documented. However, there is a multitude of papers and working documents disseminating various aspects of ICMS architecture at different detail levels. No information is available on the methodology and the standards of source code documentation.

Flexibility/Modifiability/Reusability
Object oriented paradigms, if closely adhered to, in essence imply all three properties contained in this section title. Since ICMS Builder follows OO methodology, it should be relatively easy to modify the source code or reuse the object classes. However, a more complete assessment could only be made after more detailed investigation of ICMS documentation.

Platform Integration
At present (November 2000), there appear to be no provisions within the ICMS architecture to communicate with external programs. This means that neither external models can be executed nor external software applications (such as viewers or solvers) can be incorporated into an ICMS model by a developer.

**Framework as Model Construction Environment (Model Builder Perspective)**

**Architecture**

ICMS implements the latest version of Open Modelling Engine (OME), an object oriented (OO) software architecture concept specifically designed to control the linking and simultaneous execution of a number of models.

OME treats models in a way similar to the object oriented concept of class, namely, as a data template with the algorithms for processing the data. Modelling objects are class instances that share the same model definitions but operate on different data sets. From the model developer’s point of view, this approach provides a sufficient level of abstraction for conceptual separation of models from data.

The ICMS Model Manager is capable of “reasoning” about models. It infers, essentially from the MickL models’ definitions, and then enforces the syntactical and, partially, the semantic rules for interconnecting the model objects. This ability stems from the design of MickL, the ICMS Model Representation Language and, in particular, from the functionality of its “semantic investigator”. The MickL compiler, after invoking the “semantic investigator” presents the developer with a “class” structure containing (i) a list (zero to many elements) of definitions of models that are always public, (i.e., visible outside the class), (ii) a list of public variables used in the models, usually representing “connectors” for data I/O and models’ parameters. (iii) Global variables, used in models definitions, are also detected but they are assigned to a special, build-in ICMS class.

The ICMS Builder environment also provides a multitude of other high-level model development functions that are invaluable to a model developer. These include (i) advanced visual representations of various aspects (“views”) of the developed model; the structure and components of model classes and instantiating objects can be examined in-depth, and if necessary, altered interactively; (ii) “intelligent” “Drag and Drop” linking and integration of model objects that transparently employs the information obtained by the “semantic investigator”; (iii) support for reusability of (ICMS native) models that can be stored in external libraries, (iv) a rich set of tools for I/O data visualisation (maps, graphs, etc.) and for data analysis. This functionality makes the model developer’s task easier by allowing them to concentrate on algorithms and model integration, while leaving all tedious peripheral tasks to the modelling environment.

While the ICMS Builder environment is general enough to be considered appropriate for solving numerous and diverse modelling problems, it has several drawbacks that limit its usefulness. Implementation of the new data types is severely limited. An ICMS class can contain only scalar, and 2D matrix data members. While this gives scope for defining new data abstractions, the functionality is limited because it is not possible to define a collection (array, list, etc.) of objects as another ICMS class. For example, we could create a Polygon class that defines a list of points, an area and a list of other attributes, but we couldn’t create, say, a Coverage class that contains a list of Polygons.

However, it appears that there is a mechanism that partially offsets the above drawback. A feature, that one might call “scale recursivness”, allows establishing hierarchical “parent-child” relationships between models, i.e., fully formed models (consisting of linked model objects) can again be used as components (or as model objects) to create more complex models.

As stated previously, there are neither any provisions to import external (non-native) models into the ICMS environment nor any capabilities to interact with external programs, such as optimisers, editors, or data analysis and visualisation tools.

The MickL language, while being one of ICMS’ major strengths and the facilitator of other great features, it is also a major weakness.
Certainly, by removing unnecessary programming constructs and providing built in support for matrix and grid operations as well as foundations for syntactical and semantic inference, MickL made the model development process simpler, especially for users with little or no prior programming experience.

Nevertheless, it appears that the language was exceedingly curtailed in the area of extensibility. It has no provisions for defining/implementing new data abstractions; the scalar and 2D matrix are the only available data types. Furthermore, there is little scope for implementing libraries of functions in MickL (or any standard programming language, e.g., as DLLs). Reusability can be achieved, either on the source code level (essentially, by cutting and pasting), or, in a very restricted and cumbersome way, through the model linking system. Inclusion of new, MickL native functions can be done only by extension of its grammar, with the new functions declared as the language keywords. This requires reprogramming the compiler and related parts of ICMS Builder, which, in most instances, is not feasible.

Furthermore, MickL was design exclusively as Model Representations Language (MRL), thus it cannot be used for development of new user interface components. This restriction, however, is conceptual in nature; it stems from the philosophy underlying the ICMS architecture, i.e., it enforces the paradigm of separating the model definition from the user interface. Unfortunately, this has made it difficult to create custom user interfaces for models. The problem has been addressed recently by the development of ICMS View; an interface for run-time linking of customised user interfaces implemented as MS-Windows™ Dynamic Link Libraries (DLLs).

Implementation of Standards
ICMS Builder implements the newest version of the Open Modelling Engine standard. The OME concept was conceived, designed and developed by Andrea Rizzoli from the Milan Polytechnic. Recently, Andrea has started a project, in which he proposes the use of XML (The EXtensible Markup Language) as a meta-data standard to define the I/O data abstraction for the OME architecture. The interfaces of the resulting models would be HTML web pages. The models would reside on a server, thus it would be possible to exchange data to and from the models using XML meta-data specifications. He promotes the opinion that XML has a great potential to become the meta-data standard if it gains sufficient popularity. Nevertheless, he also points out that there is already a format specification for environmental data, namely the HDF (Hierarchical Data Format). At present, there appears to be no convergence of HDF and XML.

Although the MickL model specifications language is based on C, the differences, however, are significant enough that even a seasoned C programmer could have serious initial difficulties in areas such as matrix indexing syntax, scope and definition of global and local symbols (especially functions), and rather peculiar semantics for memory allocation for symbols.

ICMS environment incorporates advanced GUI with sophisticated visual tools that implement the “Drag and Drop” metaphor, which is quickly becoming the expected standard of high profile software applications.

Documentation
At the model creator level, ICMS Builder doesn’t implement any documentation standards. There are, however, numerous provisions within the environment to manually enter and store user comments. At present, there is no context sensitive help, neither for the model creator, nor for the model user.

Framework as Modelling Environment (Model User Perspective)

---

1 HDF is often misconstrued for a meta-data specification language, in reality; it is a software toolkit and C, Java, C++, FORTRAN90 APIs for scientific data management, i.e., for analysing, visualising, and converting data. The HDF software is freely available from NSCA (National Center for Supercomputing Applications)
**Model Customisation**  
In spite of some deficiencies, ICMS Builder is an excellent model construction environment that is almost unlimited in its applicability. However, from the perspective of the end user the final product, *i.e.*, "The Model" doesn't really exist. There is no difference between the appearance of the model "construction" and "execution" interfaces. People who are not specifically trained in modelling (e.g., natural resources managers and policy-makers) find the interface entirely confusing and frustrating.\(^1\) As stated many times before, this problem is being addressed by the new version of ICMS Builder. The new ICMS View module would allow the model creator to construct highly specialised user interfaces that could be customised specifically for the target user group, according to their needs and wishes.

**Uncertainty Analysis**  
Also, there is no native support for tracking or controlling propagation of uncertainties or errors associated with input data and operations performed by the component models. Presently, uncertainty analysis has to be incorporated by the model creator into each and every component model.

**Modelling Cycle support**  
The support for the entire "Modelling Cycle" is quite poor. The graphical metaphor of the model, implemented as a directed non-cyclic graph, is conceptually close to the "natural" representation of the modelled system. However, the meaning of the graph nodes (*i.e.*, component models), rendered as icons, is often somewhat obscure, especially in the model with many disparate processes.  
As yet, there is no support for scenario/multiobjective simulations, although they can be emulated through the “Parent-Child” mechanism by constructing specialised/unique component models. There is a limited documentation/configuration support for subsequent execution of the model. The input data is persistent, ICMS Builder and also record the output and other configuration details for future retrieval and analysis. There are no provisions for inverse parameter optimisation unless they are implemented as component models. The linear programming inbuilt function is limited to linear physical or linear regression models only.

---

\(^1\) This is an observation (by PGK) from a very recent (24/11/2000) ICMS Builder seminar/workshop in Perth. There was some degree of confusion regarding the distinction between the model construction environment and the model itself.
11 Appendix E MMS Details

MMS is a framework for modelling that can be used to develop, support and apply any dynamic model. A major focus of MMS has been for models used in the environmental and natural resource management disciplines.

MMS is an integrated system of computer software developed to:
- provide the research and operational framework needed to enhance development, testing and evaluation of physical-process algorithms
- facilitate integration of user-selected algorithms into operational physical-process models
- provide a common framework in which to apply historic or new models and analyse their results

MMS uses a module library that contains modules for simulating a variety of physical processes. A model is created by selectively coupling appropriate modules from the library to create a suitable model for a desired application. When existing modules do not provide appropriate process algorithms, new modules can be developed.

MMS enables the development of models that are problem and scale specific, and provides the capability to evaluate alternative modelling and parameter estimation approaches. A variety of disciplines can work cooperatively on multidiscipline problems by developing and testing model components in their own area of expertise and then combining these modules to develop a complete system model.

11.1. Overview

The Modular Modeling System has three major components:
- A pre-process component which includes tools to input, analyse and prepare spatial and time series data for use in model applications
- The model component includes the tools to develop and apply models
- A post-process component provides tools to display and analyse model results, and to pass results to management models or other types of software

An X-window graphical user interface acts as a system supervisor and provides users access to all components and features of MMS.

MMS was developed for UNIX-based workstations and uses X-windows and Motif for the GUI, although it is apparently currently (Nov 2000) being transferred to an MS-Windows platform. In the current system, models are written in C or Fortran under UNIX and are written with standard interfaces for input and output.

Spatial data analysis can be accomplished using a GRASS-based GIS toolset, or by linking to a similar tool.

The current time-series database structure used to interface with MMS, as reported in 1995, is an ASCII flat file.
12 Appendix F SMS/WMS Details

Software vendors such as SSG (http://www.scisoftware.com/), WRCS (http://www.waterengr.com/) classify SMS (Surface-Water Modelling System) as a comprehensive environment for hydrodynamic modelling, and WMS (Watershed Modelling System) as a comprehensive environment for hydrologic analysis. Both of these classifications are rather misleading in the context of Project 1.1. Both software packages belong to the category of Integrated Modelling Suites\(^1\). The component hydrodynamic modules (comprising SMS suite), and the hydrologic analysis modules (WMS suite) are intrinsic to both software packages and tightly integrated within very impressive graphical user interfaces (GUIs). In other words, the user can neither construct and incorporate new modules (sub-models), nor modify the existing ones. The Environmental Modelling Research Laboratory (EMRL) of Brigham Young University in cooperation with the U.S. Army Corps of Engineers Waterways Experiment Station (USACE-WES) developed both systems. The Federal Highways Administration (FHWA) Office of Bridge Technology, also contributed to the development of SMS by providing coastal component models.

EMRL: http://www.emrl.byu.edu/
USACE-WES http://www.wes.army.mil/
FHWA: http://www.fhwa.dot.gov/bridge/hyd.htm

12.1. Role/Purpose

(Note: this section contains mostly information obtained from SSG (Scientific Software Group) web site, abridged in accordance with Project 1.1 relevant interests. For full information, please see the SMS and WMS home pages.)

SSG: http://www.scisoftware.com/
SMS: http://www.waterengr.com/SMS/index.htm
WMS: http://www.waterengr.com/WMS/index.htm

The Surface-Water Modelling System (SMS)
The Surface-Water Modelling System (SMS) is a comprehensive graphical user environment (or graphical pre and postprocessor) for 1D and 2D hydrodynamic modelling of:
- Complex riverine systems,
- Lakes,
- Estuaries and coastal systems.
SMS includes interfaces for the finite-element simulation packages/modules (RMA2, RMA4, HIVE2D, SED2D-WES, FESWMS and WSPRO) and version 7.0 also includes interfaces for selected coastal models developed by FHWA (ADCIRC, STWAVE, CGWAVE). All modules can be purchased separately as needed for specific problems. SMS also provides tools for:
- Mesh and grid generation,
- Data interpolation,
- Graphical visualization.

\(^1\) The latest SMS version (ver. 7.0) includes a new feature ("Generic Model Interface") that appears to make the first attempt to extend SMS functionality in the direction of integrated modelling environment; the user has the capacity, using the tools available in SMS, to define an interface? (seems more like an input data set properties?) for an external model that is not explicitly supported by the SMS environment.
Modelling domain property objects, patterned after the ARC/INFO GIS data model are used to construct conceptual models that define the mesh domain and material zones. Boundary conditions and other parameters are assigned to the property objects. A mesh can be automatically generated from the conceptual model using sophisticated mesh generation tools available in SMS. Materials and boundary conditions are automatically assigned to the appropriate mesh nodes or elements.

**The Watershed Modelling System (WMS)**

The Watershed Modelling System WMS is a state-of-the-art computer application that provides advanced interfaces for component packages/models simulating various processes within a catchment (HEC-1, TR-20, TR-55, NFF, The Rational Method, HSPF).

The WMS program utilises digital terrain data to
- Delineate catchment and sub-catchment boundaries,
- Computes geometric parameters used in hydrologic modelling (elevation, slope, flow direction, etc.),
- and provides tools for formulation and development of a catchment model such as
  - Data entry and import/export, parameterisation, etc,
  - Importing/Exporting GIS feature objects
  - Graphical terrain visualization

**12.2. Availability**

Both systems are distributed commercially and are only available as executable programs i.e., the source code or binary APIs or toolkits are not available. The cost of both packages is somewhat prohibitive too; full versions of SMS and WMS cost US$8,100 and US$1,600, respectively. This excludes 50% discount for academic institutions.

**12.3. Major Strengths and Weaknesses**

**Architecture/Usability/Versatility/Performance Strengths - SMS**
- Excellent, integrated, context sensitive, modular GUI for the following, built-in operational modules;
- 2D Mesh Module for automated mesh generation and editing,
- Grid Module for boundary fitted grids,
- Scatter Point Module for interpolation of 2D properties from scattered data points,
- Map Module for manipulation of graphical elements and feature objects,
- River Module for the creation of cross sections of rivers,
- Visualisation and Animation incorporating the most advanced scientific visualisation and animation technologies.
- Science modules contain very advanced concepts and ideas about hydrodynamic processes.

**Architecture/Usability/Versatility/Performance Weaknesses - SMS**
- New models can be constructed and incorporated into the system (SMS is not a modelling environment/framework.
- The software system components are scientifically advanced, the system is multifunctional, easy-to-use, and wrapped in impressive GUI. Nevertheless, it still represents nothing but conceptually outdated pre- and post-processor technology.

**Architecture/Usability/Versatility/Performance Strengths - WMS**
- Excellent GUI with a wealth of interfaces for component modules;
- HEC-1 and TR-20 flood hydrograph programs for rainfall-runoff process simulation,
- TR-55 module for a sub-catchment hydrograph and peak-flow simulation,
• NFF for evaluation of n-year flood-peak discharges for rural and urban watersheds from US derived regression curves for over 200 regions (n = 2, 5, 10, 25, 50, 100, and 500),
• The Rational Method for evaluation of peak-flows (Intensity Duration Frequency) for urban catchments,
• HSPF an advanced (and ancient and FORTRAN) program for simulation of land use changes and agricultural practices on hydrologic and water quality processes on land surfaces, within streams and impoundments.

Architecture/Usability/Versatility/Performance Weaknesses - WMS
• The weaknesses of WMS are essentially identical to the SMS, in addition,
• The science concepts are mostly regression based engineering methods that are not significantly advanced, and do not represent the current leading-edge methodologies in catchment runoff, chemical transport, and routing processes.

12.4. Capacity to meet Project 1.1 core requirements
None. Both packages do not provide any facilities for creation and incorporation of new models/modules.

12.5. Reasons for Further Investigation/Trial
The science in the SMS component models appears to represent the current state of knowledge about hydrodynamic and hydrological processes. Thus it is possible, or even desirable, to use the SMS (maybe even WMS) for generation of I/O data that can be employed as scientific reference standards for the Integrated Catchment Modelling Toolkit testing, validation and verification. The I/O analysis visualisation capacities of SMS and WMS are very impressive indeed and should be investigated further and harvested for SE ideas and concepts.
13 Appendix G Tarsier Details

13.1. Tarsier

Tarsier is a Windows based modelling application and a framework for developing new models. Tarsier is very “data” focused, with interactions between models occurring through shared data. Tarsier currently includes several models, as well as powerful data analysis tools such as a Spline interpolation engine and a digital terrain analysis module. All Tarsier models support dynamic visualisation of model operation and results.

Tarsier supports the rapid development of new models, as well as visualisation tools, data types and data analysis tools. Figure G.1 shows the conceptual framework underlying the current Tarsier structure, while Figure G.2 highlights the data messaging and updating system.

![Diagram of Tarsier Conceptual Framework](image1)

**Figure G.1 Tarsier Conceptual Diagram**

![Diagram of Data Handling and Messaging](image2)

**Figure G.2 Data handling and messaging**

13.2. Overview

**Data Types**

Tarsier currently supports Grids, Time Series, Sites and Node Link Networks. Furthermore, support exists for incorporating new data modules such as the Topog element network or flow tube aquifers. The openness of data types is one of the major strengths of the system.
What Does It Model?
Tarsier is a general-purpose modelling environment. It currently includes Macaque, a large-scale water yield model, Colobus, a rainfall runoff model that has also been integrated with Marmoset, a routing model for node link networks.

GIS Interaction
Tarsier supports various GIS data formats, including Arc Rasters and Arc Shapefiles. Tarsier does not currently support any GIS/Modelling component interface conventions.

13.3. Major Strengths

Speed
Tarsier models run quite fast, with little or no speed penalty as a result of running within a framework. While dynamic visualisation slows down model execution, the user has the option to select as much or as little visualisation as they require.

Simple Model Development
Tarsier provides model developers a quick way to implement and test models. All model development occurs within the Borland C++ Builder environment, a Rapid Application Development (RAD) system that provides support for developing graphical user interfaces for models.

Reasonably Lightweight framework
There is little computational, memory or system loading overload created by the Tarsier system.

13.4. Major Weaknesses

While data forms the centre of the Tarsier world, the relationship between data types is quite weak. Generalising the data manipulation interface would make it easier to develop data analysis tools (such as statistical functions or scripting languages). The Tarsier framework provides less modelling infrastructure than toolkits such as ICMS. In particular, Tarsier does not have model linking tool, model execution engine or the automatic generated user interfaces.

Tarsier requires more programming expertise than other toolkits, such as ICMS, but still less than traditional model development. In particular, the use of C++ as a model implementation language requires a model developer to have more programming expertise than languages such as Visual Basic or MickL.

Message Passing
The message passing system is a generalised mechanism for supporting communication between models, data and visualisation tools. While it provides good support for interactive visualisation and simple model integration, it is an error prone approach when major model integration is required. In particular, it requires model developers to be acutely aware of possible model integration combinations and provide tests to prevent deadlocks.

Documentation
There is currently little technical documentation on Tarsier. There is a user manual for learning to operate the application shell as well as the established models such as Macaque.

13.5. Minor Issues

Scalability
While Tarsier performs quite well on moderately large modelling problems, it’s reliance on Windows as well as its software architecture, which requires all data to be in memory at one time, may limit its usability in large modelling applications.

**Portability**
While Tarsier’s language of implementation, C++, is a reasonably portable language, Tarsier relies heavily on the libraries of C++ Builder – currently a Windows only development environment. An Intel Linux version of C++ Builder is due for release early next year and should facilitate multi-platform development.

**Generalised tools**
Much of the visualisation in Tarsier is within independent windows, such as Raster View or Time Series View. This gives the user a great deal of flexibility in how they view their data. Unfortunately, it doesn’t give the developer a lot of scope for reusing and extending existing views in custom user interfaces. This makes it difficult to create highly tailored interfaces for particular clients. It also makes it more difficult to control the users interaction with the data within a particular application. For example, there is no way to use a raster view within a window that asks the user to select a point on a map. The only solution is to create an independent raster view of the data and record “move cursor” messages from the data.

**C++SL (C++ as a Second Language)**
Borland C++ Builder supports the development of applications that load Dynamic Link Libraries at runtime. Within the Tarsier environment, these DLLs typically either provide a visual (such as a data view, or a model view) or a non-visual (such as a data type or a model) module. As C++ Builder applications rely on Borland’s proprietary VCL (Visual Component Library) for interaction with Windows, there are question marks over the support of other Visual modules built with User Interface toolkits. In particular, VCL relies on several global variables for managing interaction between windows. Development environments built with non VCL libraries do not include these variables. This problem would be less of an issue with Microsoft Foundation Classes, which are supported by many Windows development environments, or Qt, which is a cross platform user interface toolkit for C++.

It is not clear if Tarsier module development can occur in environments other than Borland C++ Builder.