Decision Support Systems

(DSS)

What is a DSS?

- Very generic term, many definitions
- “Interactive computer-based system that helps decision makers in the solution of unstructured problems”
- something is missing...

Some definitions

- DSS are a model-based set of procedures for processing data and judgments to assist a manager in his/her decision [Little, 1970]
- DSS couple the intellectual resources of individuals with the capabilities of the computer to improve the quality of decisions. It's a computer-based support for management decision makers who deal with semi-structured problems [Keen & Scott-Morton, 1978]
- DSS is a system that is extendable, capable of supporting ad hoc analysis and decision modelling, oriented towards future planning, and of being used at irregular, unplanned intervals [Moore & Chang, 1980]

More definitions

- DSS enable mangers to use data and models related to an entity (object) of interest to solve semi-structured and unstructured problems with which they are faced [Beulens & Van Nunen, 1988]

- Main feature of DSS rely in the model component. Formal quantitative models such as statistical, simulation, logic and optimization models are used to represent the decision model, and their solutions are alternative solutions [Emery, 1987; Bell, 1992]

- DSS are systems for extracting, summarising and displaying data [McNurlin & Sprague, 1993]

Structure in problems

Structured	Semi-structured	Unstructured
The solution can be packaged in a computer program	A solution can be found, on the basis of previous experience, with small adaptation	General problem solving strategies: - analogy - redefinition - intuition - approximation

What is a DSS then?

- "A interacting computer-based system that helps the decision maker in the use of data and models in the solution of unstructured problems"

Scott-Norton (1971)

DSS features

- assist managers in unstructured/semi-structured tasks
- support rather than replace the human DM
- improve the effectiveness rather than the efficiency
- combine the use of models or analytical techniques with data access functions

DSS features

- Emphasise flexibility and adaptability to respect changes in the decision context
- Focus on features to enhance user interaction

Parker and Al-Utabi, 1986

Intelligent decision making

- The use of Artificial Intelligence tools and models provides direct acces to expertise, and their flexibility makes them capable of supporting learning and decision making processes. There integration with numerical and/or statistical models in a single system provides higher accuracy, reliability and utility [Cortés et al., 2000]

Environmental Decision Support Systems

(EDSS)

What is an EDSS

- An Environmental Decision Support System can be defined as:
- “a computer-based interactive system, helping the decision makers in the use of data and model to search for a solution to a natural resource(s) management problem”

More definitions

- An EDSS is an intelligent information system that reduces the time in which decisions are made in an environmental domain, and improves the consistency and quality of those decisions [Haagsma & Johanns, 1994]

- a DSS is a computer system that assists decision makers in choosing between alternative beliefs or actions by applying knowledge about the decision domain to arrive at recomendations for the various options. It incorporates an explicit decision procedure based on a set of theoretical principles that justify the “rationality” of this procedure [Fox & Das, 2000]

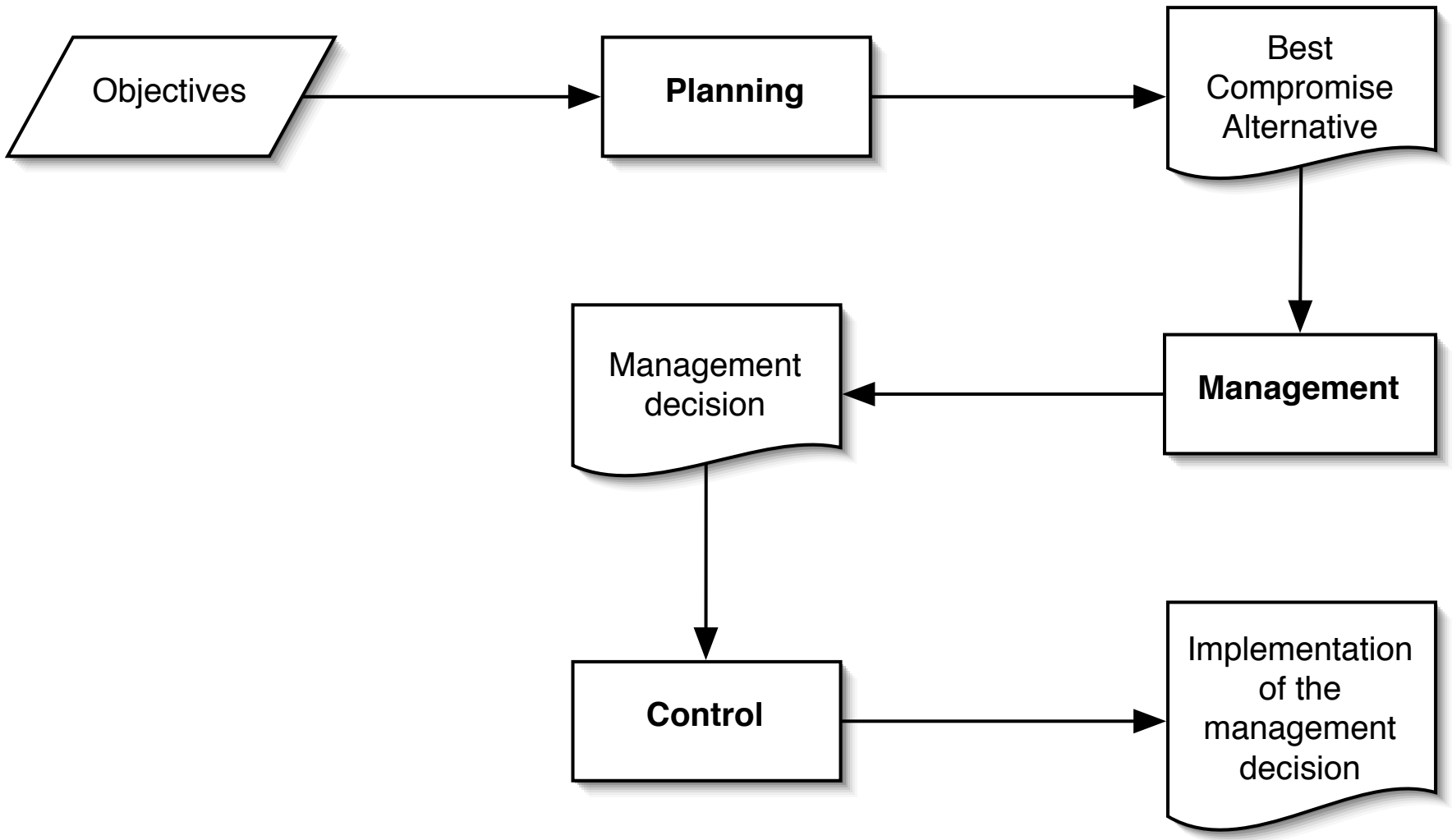
What an EDSS is not

- not a GIS
- not a simulation model
- not a statistical model of the data
- not a forecast model
- not a database
- ... all of this, and much more

EDSS types

- DSS, for complete rationality
  - it is typically used for problems with one DM and one objective
- MODSS (Multiple Objective DSS), for incomplete rationality
  - it is used in cases where there are multiple objectives and/or multiple decision makers.

The decisional process



EDSS/P
the planning level

- Select intervention options
- Long term horizon

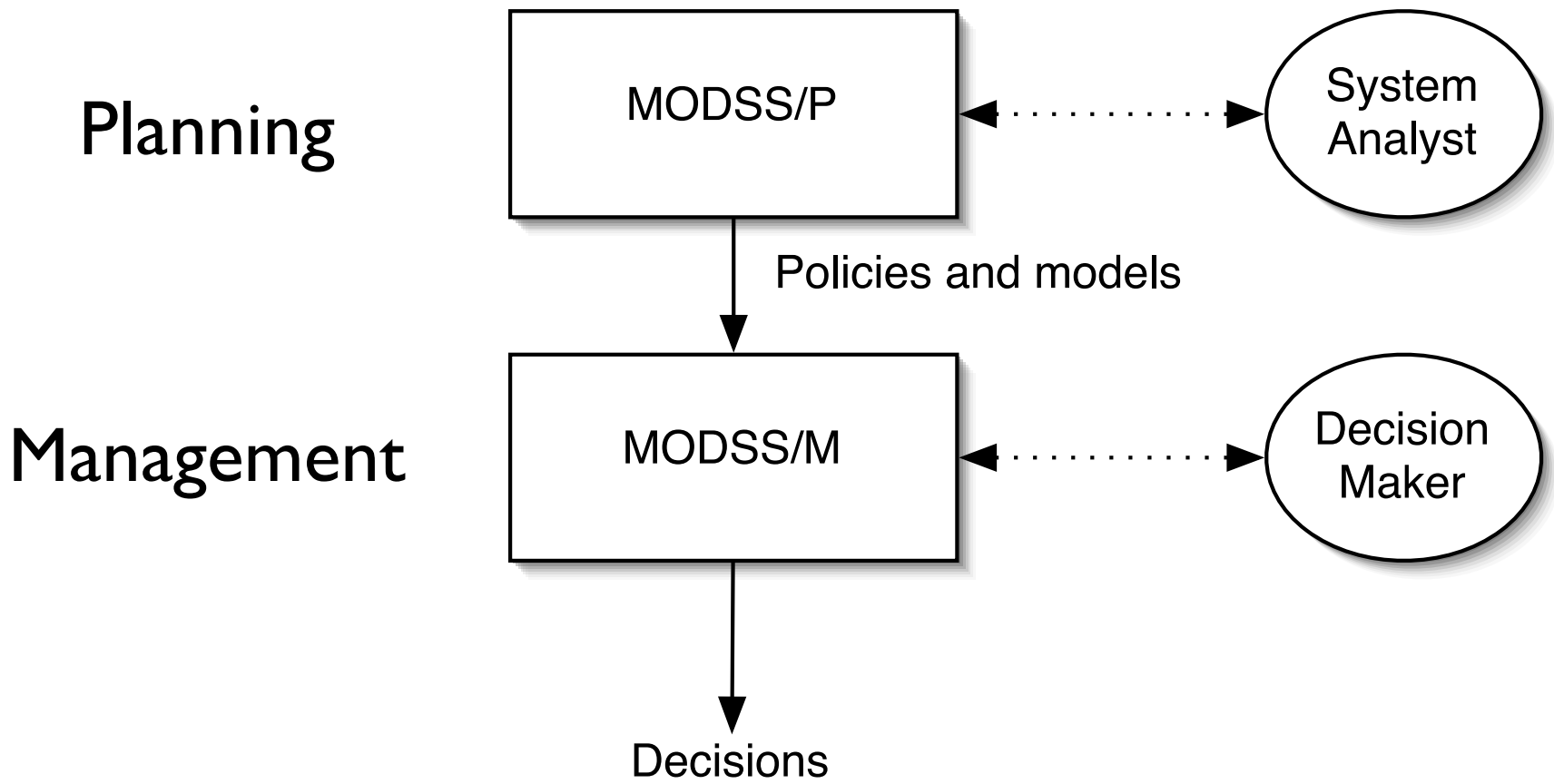
EDSS/M
the management level

- Make decisions, following the policies of the planning level (e.g. the daily release decision from a reservoir)
- Short to mid-term horizon

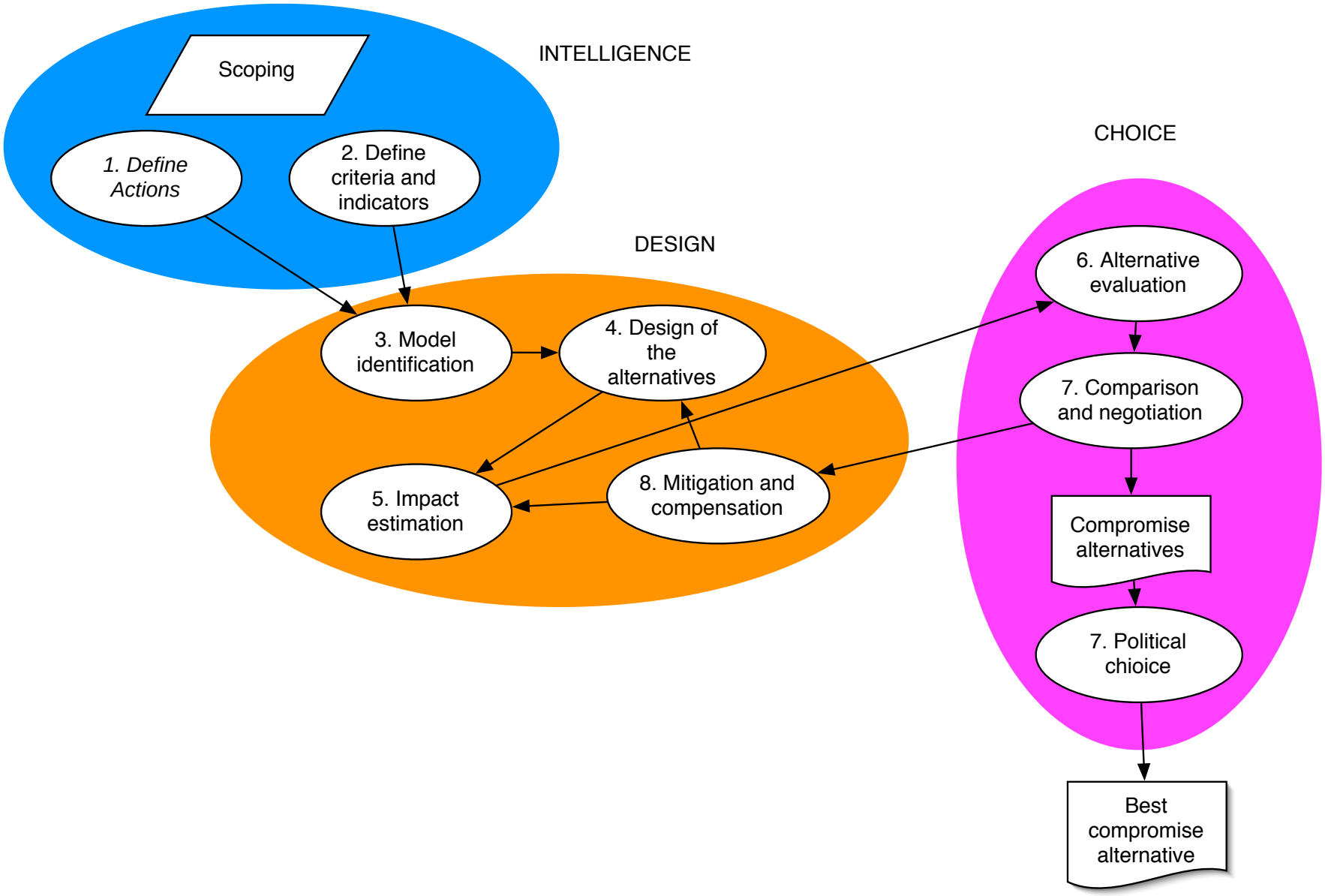
Control level

- Implement decisions
- Automatically
- Short term

A multi-level DSS



MODSS to support IMA-PIP



Building and Delivering Environmental Decision Support Systems

The EDSS IDE

- The EDSS ‘integrated development environment’
  - it is the set of components and the logic which can be used to build an EDSS application

The EDSS IDE

- It is an integrated software development environment (IDE) with access to domain specific libraries of components:
  - modelling components (integrated modelling environments)
  - numerical methods, solvers, ...
  - expert system shells
  - ...continues...

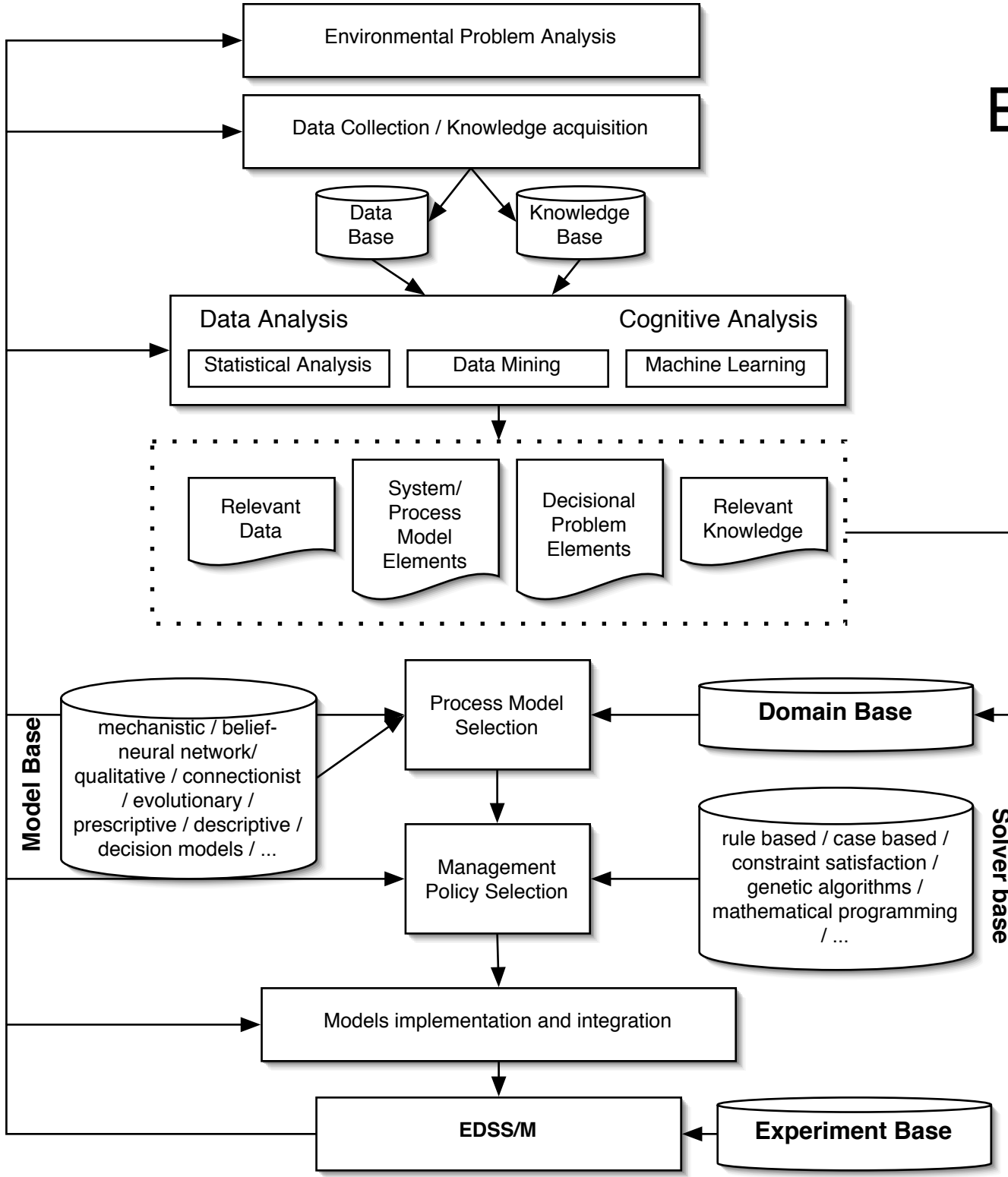
The EDSS IDE

- Data mining, statistical data analysis libraries
- Data Base management tools, data access libraries (e.g. ODBC, JDBC)
- Geographical Information Systems software components (e.g. MapObjects)
- ... continues

Delivering an EDSS

- The IDE is a programming environment which allows to build and deliver an EDSS application
- The EDSS ‘applications’
  - at the planning level
  - at the management level

Development of an EDSS/M



Environmental Scientists

AI Scientists

Modelling Experts

Computer Scientists

Adapted from Comas et al. 2002

EDSS applications: planning

- Full access to modelling environments
- Access to solver libraries
- Guided project management (intelligence, design, choice)
- Deliver policies for the management level

EDSS applications: management

- Implement the management policies
- Integrated with SCADA systems
- Close connection with operational control software level

Modelling languages and development platforms

Languages

- Matlab
  - very powerful set of libraries, simple syntax
- Mathematica
  - symbolic computations, solves ODE and PDE

Conceptual modelling tools

- Extend: discrete-event, continuous, hybrid simulation tool
- Modelmaker: visual simulation tool, parameter estimation capabilities
- Stella/iThink, Powersim, Vensim: adopt the World Dynamics approach
- Simulink: built on top of Matlab

Integrated modelling environments

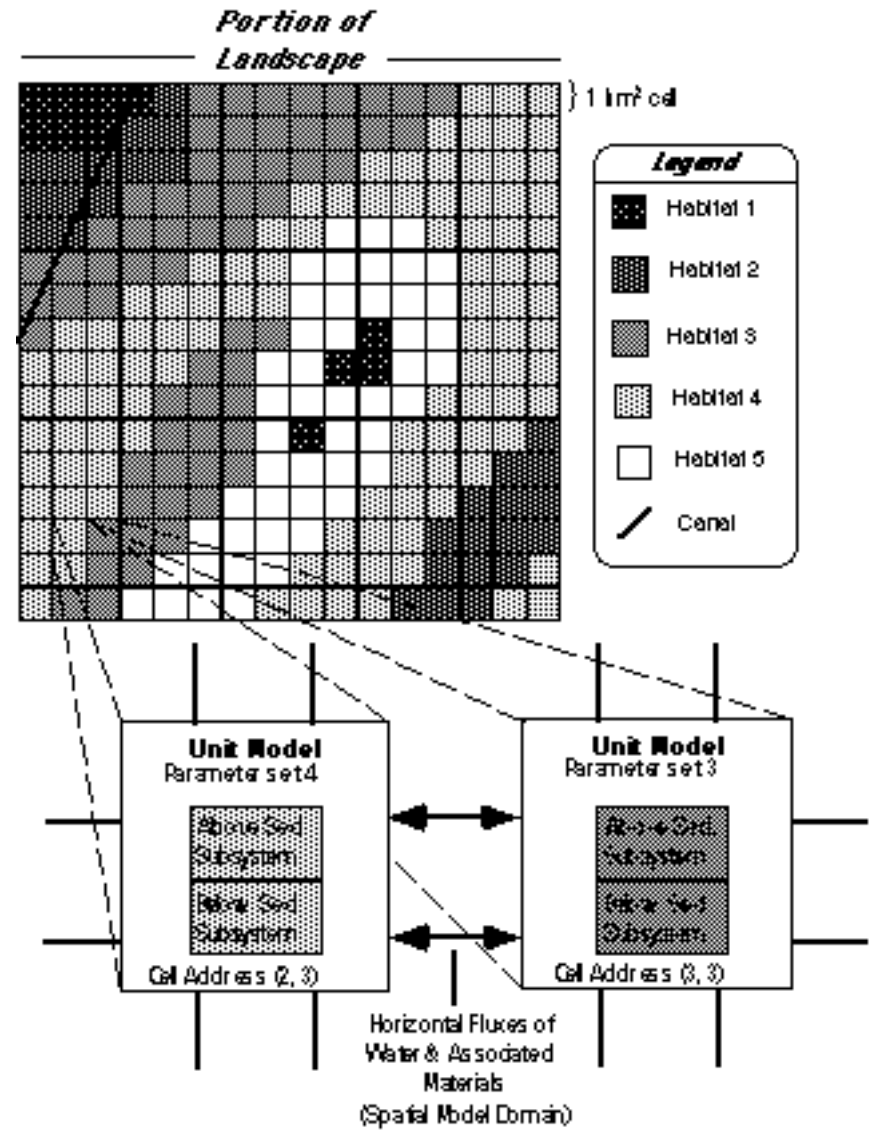
- SME
- ICMS
- Time
- Open-MI
- IMA
- Simile

Spatial Modeling Environment

- Addresses spatio-temporal modelling
- Icon-based graphical modelling environment
- Modular and re-usable model components
- May use parallel computer architectures
- Developed at Univ. of Maryland and now at University of Vermont, Gund institute

The Spatial Modeling Environment

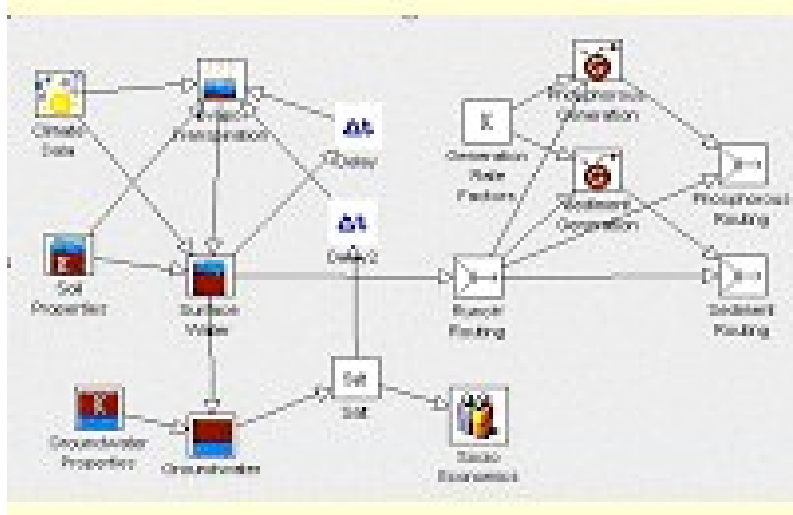
- Each cell has a (variable) habitat type, which is used to parameterize the unit model for that cell.
- The unit model simulates ecosystem dynamics for that cell in the above- sediment and below-sediment subsystems.
- Nutrients and suspended materials in the surface water and saturated sediment water are fluxed between cells in the domain of the spatial model.



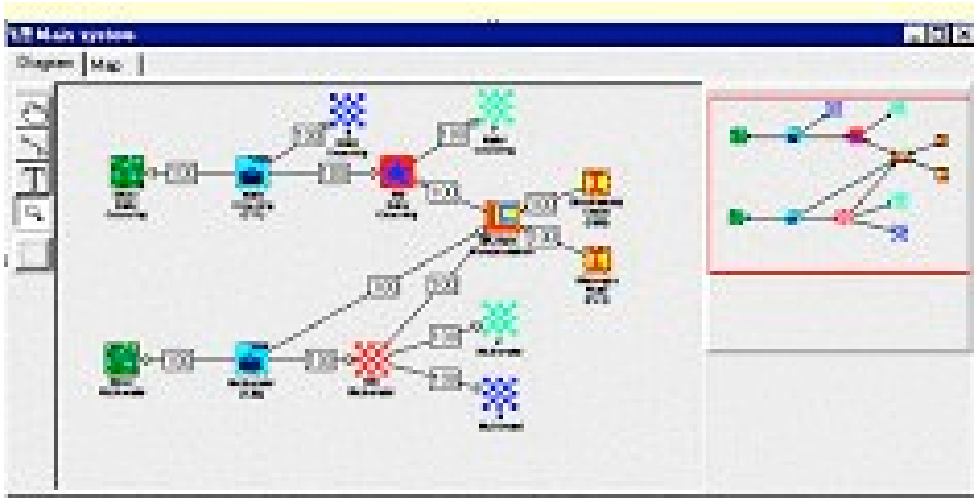
The Integrated Component Modelling Toolkit

- Originally designed for catchment modelling
- It allows to define the model as a spatial view, but also as a process view
- It allows to build complex networks of basic models to create compound models
- Designed by CSIRO Land and Water

A spatial view



A process view



Exploring a compound model

TIME, the Invisible Modelling Toolkit

- It is the building block of the Catchment Modelling Toolkit
- Supports for the represenation, management and visualisation of a variety of data types
- Support for testing, integrating and calibrating simulation models.
- http://www.toolkit.net.au/cgi-bin/WebObjects/toolkit.woa/wa/produ

TIME

- Models are described by means of meta data structures
- It is easier to design a user interface or to apply a parameter estimation algorithm
- It is based on the .NET architecture

Simile

- Based on declarative modelling
- System dynamics based user-interface
- Supports various modelling approaches: ODE, matrix algebra, cellular automata

Software Engineering for EDSS

A quick ‘n’ dirty note

O-O analysis, design and programming

- Object-orientation = objects + classes + inheritance (Wegner, 1987)
- A step towards modularity and reusability
- Object-orientation can affect all phases of software development

Objects

- According to Booch (1999) an object
  - has a unique *identity* (cannot be mistaken for another object)
  - has a *state* (identified by values of its *attributes*, which can also be objects)
  - has a *behaviour*, described by its *methods*

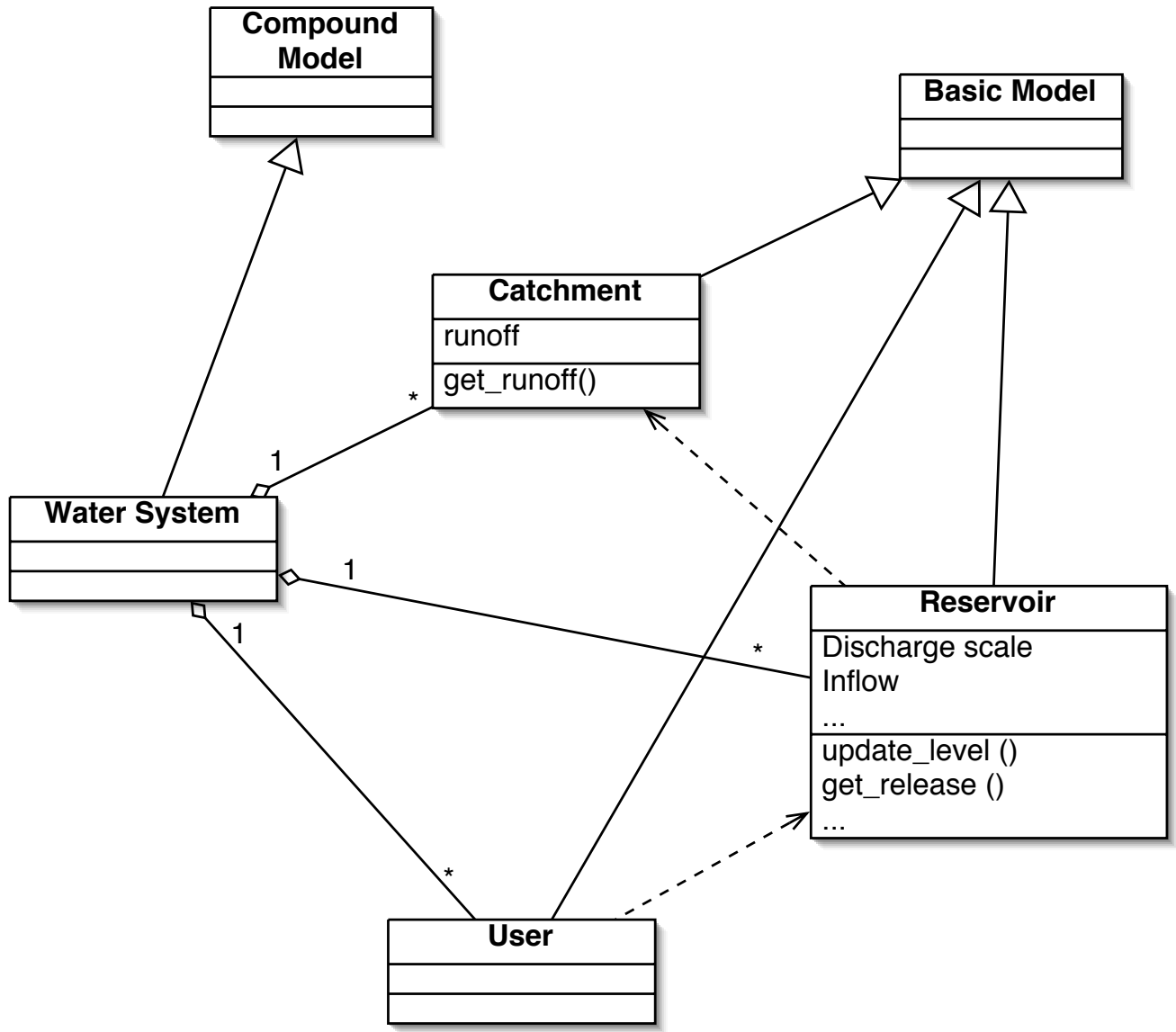
Objects at work

- They are useful to implement models
  - parameters are attributes
  - state transition and output transformation are methods
- E.g.: a simulator calls the object "run()" method to evaluate the alternative impacts

Classes

- Objects are data, Classes are data types
- Generalisation relationships are modelled in a lattice (is-a relationships)
- Association and aggregation (part-of) relationships can also be modelled

Class diagram



Separation and orthogonality

- Object-oriented programming allows to keep code separated and orthogonal
- Separation: the code is organised in *modules* with well defined *interfaces* (strong cohesion, loose connection)
- Orthogonality: modules can be combined in various ways to achieve a result (a solver is orthogonal to a model)

UML

- Unified Modelling Language (Booch, Rumbaugh, and Jacobsen)
  - describes static and dynamic interactions among objects and classes

Component Software Engineering

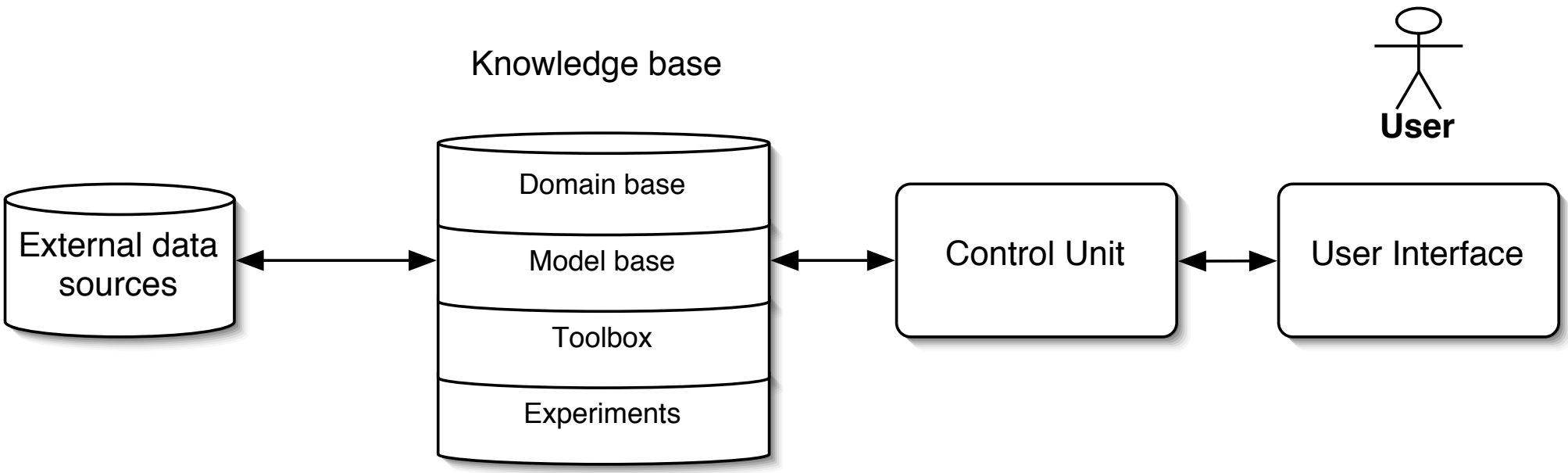
- Enhance software re-use
- A component has the following properties (Szyperski, 2003):
  - is a unit of independent deployment
  - is a unit of 3rd party copmposition;
  - has no (externally) observable state

Components and objects

- On the contrary, an object is:
  - a unit of instantiation, unique identity
  - may have an externally observable state
  - encapsulate its state and behaviour

An architecture for model and data re-use

The software architecture



The knowledge base

- the extended knowledge base
  - the domain base
  - the model base
  - experiments and case base
  - solver toolbox
- external data sources

Domains

- A domain base is a repository which contains the data relative to the physical description and the measurements of a system.
- The definition of domains is the first modelling phase:
  - it does not involve any assumption on the kind of mathematical relationships
  - it concerns only the representation of our knowledge on the system

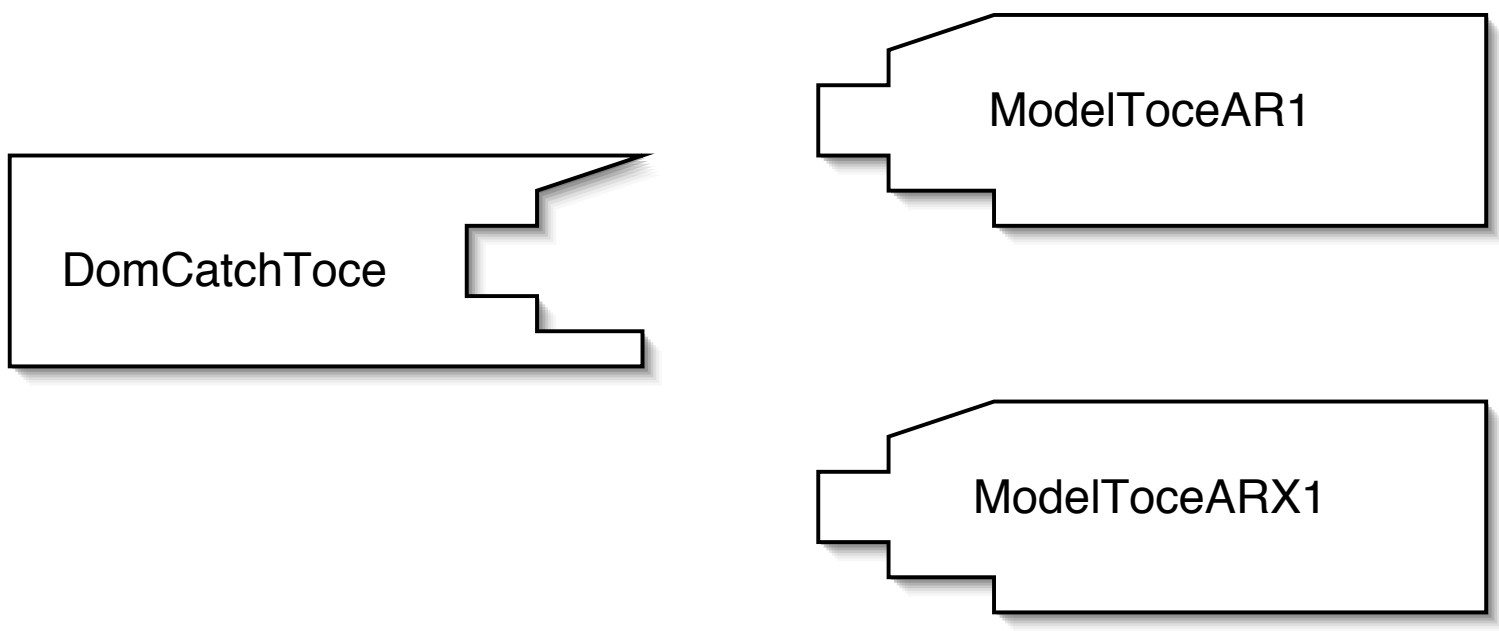
Domains (cont)

- Domains are represented using traditional knoweldge representation techniques such as semantic nets and frames
- An object-oriented design approach is adopted
- UML is used as a modelling tools

Models

- Models are representations of the processes taking place in our system
- They can be developed through successive approximations, from conceptual to mechanistic
- ‘Deployable’ models refer to data classes declared in the domain base

Model re-use



Domain Base

Model Base

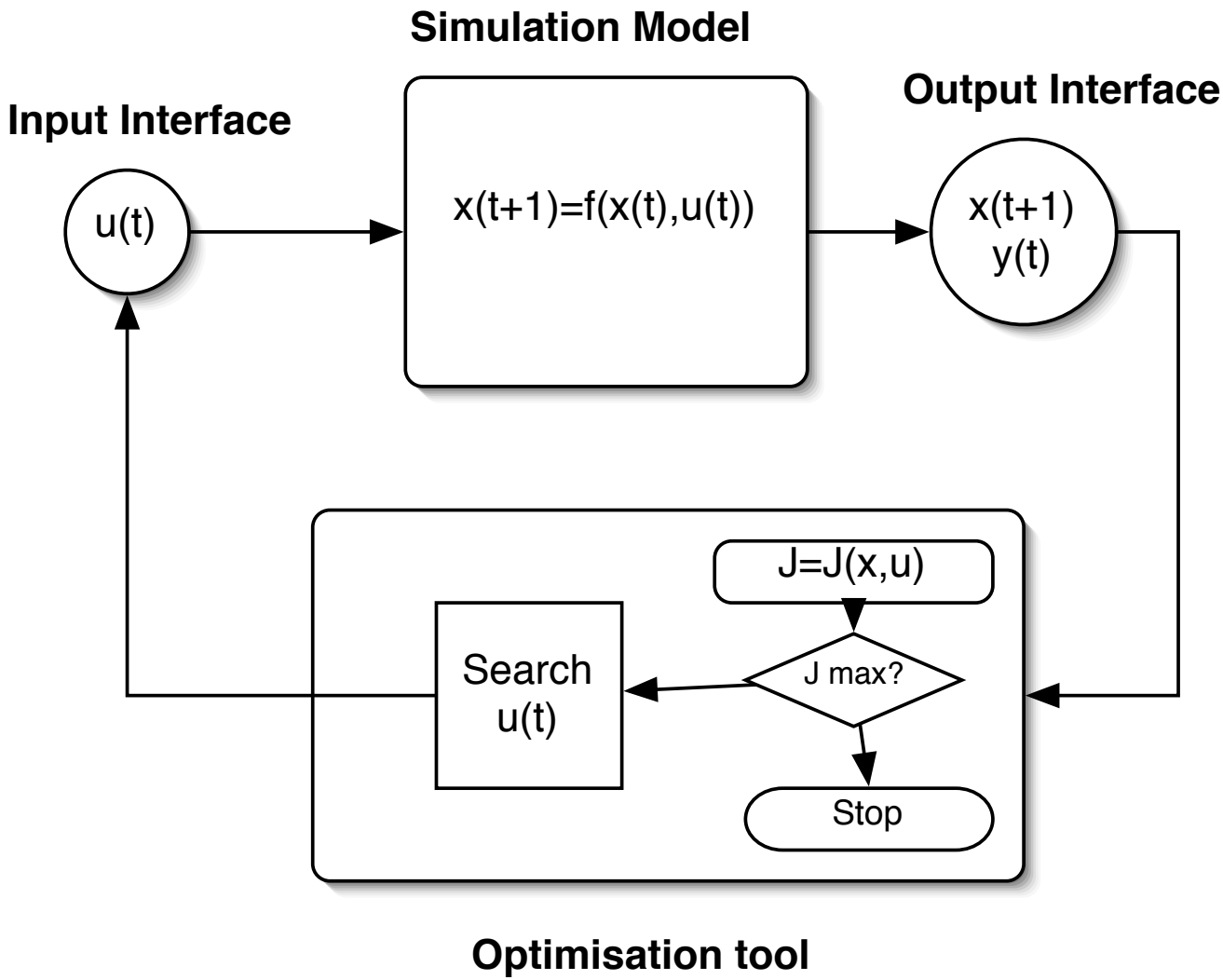
The solver toolbox

- It contains numerical methods, inference engines, statistical analyses and data mining tools
- They are implemented as software components

Tools as software components

A tool operates on a model instance

Tools as software components



Experiments

- Experiments are used to keep trace of the sequences of operations that led to a given result
- An experiment can be a policy design
- An experiment can be a scenario analysis
- An experiment can be a forecast

The control unit

- the inference engines (ES, CBR)
- data processing (ANN, Data Mining)
- workflow management tools

The control unit

- It is heart of the EDSS, it manages the interaction with the user, by means of the User Interface
- The control unit can operate in two different modes:
  - execution mode
  - development mode

The execution mode

- It supervises the execution of experiments and the flow of execution in the decision making process
- It allows to create new experiments analyse and store the results
- It supports both the Planning and the Management level,

Development mode

- This mode allows to access the EDSS IDE features, to create new domain, model, experiment classes, to be later used at the execution level

A quick look into the (possible) future

Promising evolutions

- Workflow management systems
- Content management systems
- The semantic web, web-services, distributed environmental ontologies
- Distributed, massive computing, grid computing

End of the course

Thank you!