International Institute for Applied Systems Analysis
IIASA www.iiasa.ac.at

Knowledge-based Model Building with KONWERK

Funke, B. & Sebastian, H.-J.

IIASA Working Paper

WP-96-105

September 1996

Funke B & Sebastian H-J (1996). Knowledge-based Model Building with KONWERK. IIASA Working Paper. IIASA, Laxenburg, Austria: WP-96-105 Copyright © 1996 by the author(s). http://pure.iiasa.ac.at/id/eprint/4922/

Working Papers on work of the International Institute for Applied Systems Analysis receive only limited review. Views or opinions expressed herein do not necessarily represent those of the Institute, its National Member Organizations, or other organizations supporting the work. All rights reserved. Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage. All copies must bear this notice and the full citation on the first page. For other purposes, to republish, to post on servers or to redistribute to lists, permission must be sought by contacting repository@iiasa.ac.at

Working Paper

Knowledge-based model building with KONWERK

*Birger Funke**
*Hans-Jürgen Sebastian**

WP-96-105
September 1996

IIASA International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria
Telephone: +43 2236 807 □ Fax: +43 2236 71313 □ E-Mail: info@iiasa.ac.at

Knowledge-based model building with KONWERK

*Birger Funke**
*Hans-Jürgen Sebastian**

WP-96-105
September 1996

*Rheinisch-Westfälische Technische Hochschule Aachen,
Operations Research, Templergraben 64, 52056 Aachen,
Germany
and
International Institute for Applied Systems Analysis
2361 Laxenburg, Austria

Working Papers are interim reports on work of the International Institute for Applied Systems Analysis and have received only limited review. Views or opinions expressed herein do not necessarily represent those of the Institute, its National Member Organizations, or other organizations supporting the work.

International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria
Telephone: +43 2236 807 □ Fax: +43 2236 71313 □ E-Mail: info@iiasa.ac.at

Abstract

Modeling a real world optimization problem in a form which can be processed by a machine (computer) is usually a very difficult and complex task. Therefore, building and verifying the model is often the most time consuming part of the whole process of solving a real world problem using methods of Operations Research. Software tools, which integrate representation methods developed in the field of Artificial Intelligence (AI) and methods of OR, can facilitate and speed up the process of model development.

The paper introduces the idea of knowledge based modeling as a model development and representation technique facilitating the complex process of model building. We describe the KONWERK tool-box which combines hierarchical structured knowledge representation and object oriented methodology thus providing a framework for model building and application of different optimization methods. We want the reader to form an idea of the methodology of model development and knowledge representation with KONWERK and to understand the hierarchical structure of the knowledge base.

The model of the Nitra River Case is used to describe and explain the modeling and knowledge representation with KONWERK. A given multicriteria model of the Nitra River Case was reimplemented using KONWERK within about three weeks and later enlarged by implementation of additional fairness criteria.

Contents

1 Introduction

In this paper we want to introduce knowledge based model building. In particular we describe the KONWERK tool-box [1] and show modeling of optimization tasks with KONWERK. We use the model of the Nitra River Case, which is described in [Mak95] as an example of the field of environmental research problems.

Solving a real world optimization task generally consists of three main parts.

- modeling the task in a form which can be processed by a machine (computer)
- solving a problem assigned to the resulting model
- interpreting and explaining the results

In many cases the modeling task is the most time consuming part. The modeling process can be considered as the "transformation" of a given problem from a humans formulation into a form which can be processed and solved by a machine (computer program). Why is this transformation so difficult and time consuming?

First of all, modeling an Optimization Problem generally includes at least two steps of "transformation":

- from the human formulation into an OR model
- from the OR model into a computer language or into a standard format file for a chosen solver program

Unfortunately, human experts descriptions are often difficult to extract and sometimes incomplete. Experts often can not describe and explain their knowledge in a proper way (even if they really want to do it - see [Kar90], [Wie94]). [2]

Furthermore, the structure of the real world problem often does not match directly with the structure of an OR model. Hence, in many cases the

[1] KONWERK was developed within the BMFT-project PROKON as a joint work of Aachen University of Technology, Hamburg University, Chemnitz/Zwickau University of Technology and Halle University. This project was supported by the German Federal Ministry of Research and Technology (BMFT) under grantnumber 01 IW 202 F.

[2] In the field of expert system developement the knowledge acquisition process plays an important role in the whole process of developing a decision support system (expert system). There are several techniques developed for this task like heuristic or hierarchic classification methods, different interview techniques and indirect acquisition techniques (see [Kar90]).

"transformation" includes reformulation of the model in a different way and sometimes even relaxation of restrictions and so on. Modeling is in general not just a straight forward transformation but often some "kind of art".

People who want to develope the model

- have to know the specific application domain very well or
- need good experiences in knowledge acquisition methods
- need to be familiar with the properties of different methods of OR
- must know the available OR-tools and other software and
- usually also need to be experienced in writing computer programs in at least one computer language

Summarizing, people who want to build a model for a real world optimization problem face many different tasks at the same time. The combination of all these subtasks makes the modeling process as a whole very complex and therefore tough. A model builder, who wants to develope an OR model accurately representing all important facts of the real world, almost needs to be an allround genie.

At the end of a complex model building process the expert himself will often not be able to read and understand the final model which is actually a computer program or a (low level) standard format like a MPSX file format. Who can check whether this model really expresses the same as the expert has in mind? Can we trust the model outcomes?

This paper aims to show how the use of knowledge-based modeling techniques can significantly ease and speed up the model building process of real world optimization tasks. The resulting model can be easier read, checked and modified.

2 Brief Description of the Nitra River Case

In this section we give a very brief introduction to the Nitra River Case [Mak95]. This application will then be used as a reference example in order to explain modeling with KONWERK and to show examples for representation of different types of knowledge in KONWERK.

In the Nitra River Case we consider a river basin consisting of a main river (River Nitra) and its tributaries. The tributaries themselves also may have tributaries. Along the rivers there are different types of nodes. These are monitoring points, emission points and other types of nodes which are not important for the less detailed description of the model in this paper (see figure 1). In order to reduce the pollution of the river and to improve the water quality of the river system there are treatments to be implemented

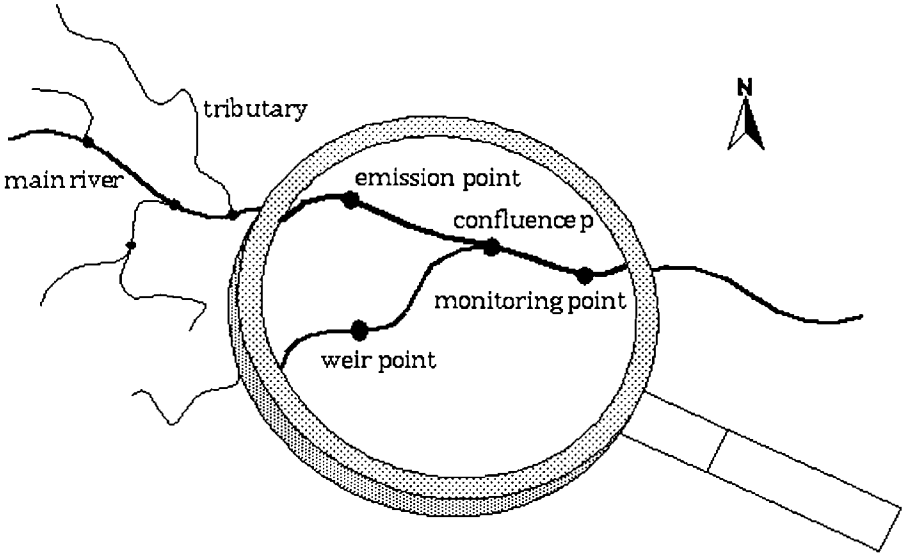

Figure 1: River System

at the emission points. For each emission point a set of different treatment technologies is considered. Only one treatment technology can be applied at an emmision point.

There are 6 objectives considered:

- Maximize the minimum concentration of oxygen at the monitoring points of the river system (DOmin)
- Minimize the maximum concentration of carbonaceous oxygen and nitrogenous oxygen at the monitoring points of the river system (BOD-max)
- Minimize the maximum concentration of ammonia (NH4) at the monitoring points of the river system (NH4max)
- Minimize total investment costs of all applied treatments
- Minimize total operating and managment costs of all applied treatments
- Minimize total annual costs of all applied treatments

The decision variables are the treatment technologies to be selected for the considered emission points. The Nitra River Case can be represented as a linear mixed integer problem (MIP). A more detailed description of the Nitra River Case is in [Mak95].

3 KONWERK - a modular tool box

3.1 The modular structure of KONWERK

KONWERK was originally developed to support configuration and design tasks in technical domains [Gün91]. During the development of KONWERK several applications mainly of engineering domains have been considered. Anyhow, the configuration of complex systems is not restricted to technical domains only. KONWERK was also applied for layout design (a logistic layout [Thä95a] and the layout of an aircraft cabin [Kop95a]) and is currently used for an environmental project (configuration of hydro geologic models for underground water quality management and planning tasks at the Halle University [Ang95]). Thus, methods of knowledge representation and modeling support developed and used in KONWERK can successfully be used for many problem classes. The extension modules for optimization tasks enable to apply KONWERK even for modeling and solving different types of optimization problems.

KONWERK is a tool-box that consists of four basic modules covering the following general tasks: [Gün95]

- representation of domain objects
- representation and processing of relations, constraints and heuristics
- formulation of the configuration task
- control of the configuration process

Additionally, there exists a variety of extension modules. The set of implemented extension modules enlarges the functionality of KONWERK for specific task types. Due to the concept of a tool-box the user of KONWERK can build his/her tool in a flexible way by selecting the necessary modules. Extension modules extend the basic modules only in a conceptual view. On the implementations level they are often complex modules which contain all the functionality of a basic module and a lot of functional extensions of the basic modules.

The conceptual hierarchy may serve as an example for the principle of the modular tool-box. The basic module for representation of domain objects is a "simple" version of the taxonomic hierarchy concept (section 4.1). This enables the user to describe the objects as a strict taxonomic hierarchy and to define their properties. There are several extensions of this basic module. We have one extension module for each of them. Therefore, the knowledge engineer can select the functionality he needs and does not have to pay for power he does not want.

engineer can select the functionality he needs and does not have to pay for power he does not want.

Extensions for the taxonomic hierarchy module are:

- views and mixins [Hotz95]
- fuzzy properties [Schu95]
- linguistic values
- spatial properties (2D and 3D) [Kop95b]
- fuzzy is-a hierarchy
- measures

In this paper we focus on the basic modules and extension modules which are relevant for optimization tasks only. In the next two subsections we give a brief introduction to the requirement modeling module and the optimization modules of KONWERK.

3.2 Requirement modeling

The extension module "Requirement Modelling" of KONWERK [Thä95b] supports the task specification. That means, this module helps to formulate the user requirements as well for crisp as for imprecise linguistic requirements. There are several possibilities to model such requirements, e.g. as

Restrictions:	required components (generic objects) or individuals, required parameter values, local constraints (binary relations between components), global constraints (restricted resources of the goal system)
Goals:	objectives or criteria for the goal system, which should be maximized or minimized by selecting or constructing an appropriate solution
Functional Requirements:	requirements formulated as functional abilities of the goal system modeled by crisp or linguistic expressions (e. g. the car should be "safe" and "fast", the computer should have "graphic ability").
Requests:	optional requirements of the types considered above; requests have a lower priority and might be relaxed or removed during the configuration process.

All the requirements formulated above might be crisp or imprecise. To be able to deal with the imprecise case, we need extension modules which allow us to deal with a fuzzy-is-a-hierarchy, compositional hierarchies with fuzzy attributes and fuzzy constraints and goals [Seb94], [Zim94].

3.3 Optimization Modules

As we have stated above, all optimization modules are belonging to the class of extension modules of KONWERK. That means: KONWERK can also be used without optimization modules. Optimization methods are only one type of problem solving mechanisms which can be used in KONWERK.

There are several optimization modules in KONWERK. One specific extension is the so called "Basic Optimization Module - BOM" [Fun95]. This module acts as an interface between the knowledge-based system and a set of optimization methods and algorithms which are available using the other extension modules for optimization like "linear optimization", "non-linear optimization with constraints", "optimization module MADM" and some other more.

The concept of integration of different optimization methods with the knowledge-based system is shown in the following figures:

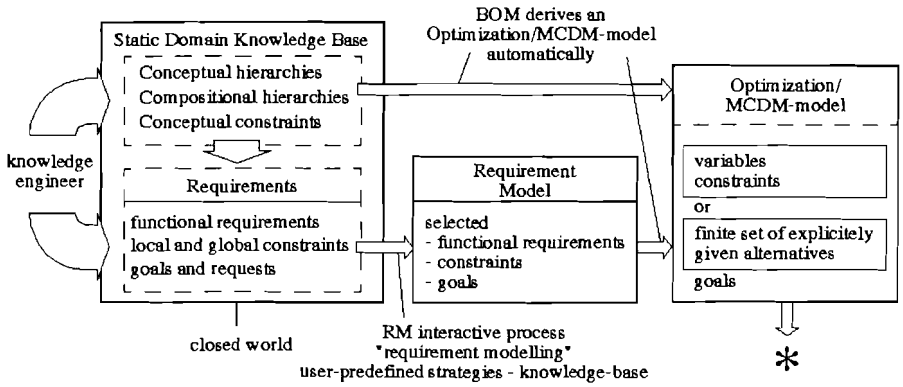

Figure 2:

The BOM transforms knowledge from the knowledge-base into a representation of knowledge which is required for an optimization model. In particular, the conceptual constraints, which are connected with the conceptual hierarchy and "activated" by the requirement model, will be transformed into a representation of constraints, which is usually used in the Operations Research.

After this transformation the BOM analyzes the properties of the optimization model and supports the interactive process of the definition of an op-

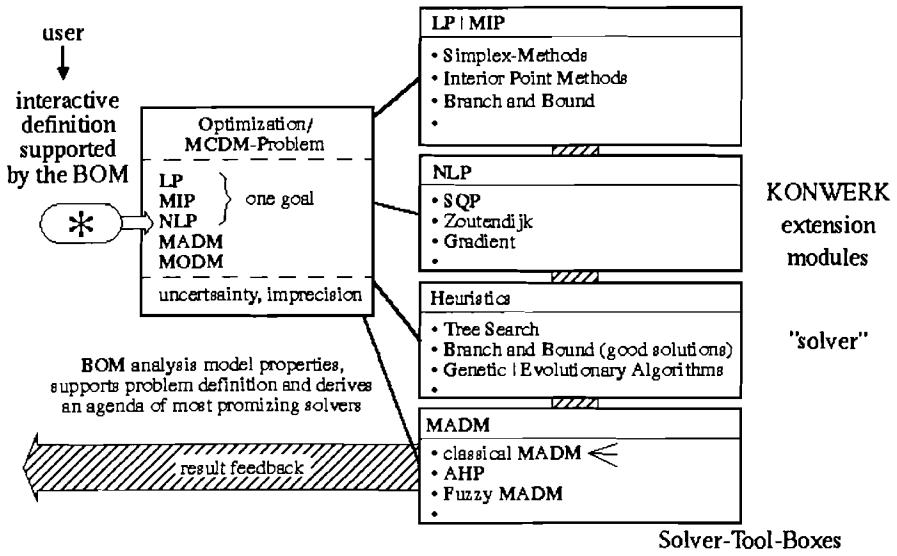

Figure 3:

timization/MCDM problem related to the model. The BOM does an automatic classification of the model components and supports the definition of an appropriate optimization problem. The next task of the BOM is to match these properties with the requirements and the assumptions of the optimization methods which are available. Finally, an agenda of the most appropriate - for the problem - optimization methods is generated and the respective optimization extension module is called to run the chosen optimization algorithm (see figure 3).

In KONWERK we can realize three different approaches to integrate knowledge-based configuration with optimization or multi-criteria decision making:

(1) The overall configuration problem (preliminary design or configuration of the target system) is modeled as an optimization problem. There is no decomposition of the target system.

(2) The target system is decomposed into components (subsystems, elements). Then, the control strategy of the configuration process uses optimization as one particular method to specialize subsystems (or to select subsystems).

(3) Under the assumption of (2), the control strategy is an optimization strategy e. g. a branch and bound algorithm. These different ways to

combine configuration with optimization are only briefly introduced, but discussed more detailed in [Seb94], [Fun95]

Modeling with KONWERK means to develope a knowledge base which represents the knowledge of a specific domain in a *declarative* way. People can define their own concepts and thus they can use the domain specific notions for the description of the model. In a second step objectives and optimization tasks can be added to the knowledge base thus building up a knowledge-based representation of an OR-model. The transformation of this OR-model into a form which can be processed by standard problem solvers (e.g. into a MPSX format for LP solvers) can be done automatically. The model builder does'nt need to pay so much attention to many technical details anymore. Therefore, he can focus on the process of model development itself.

In KONWERK, there are different types of knowledge strictly distinguished. This forces the user to develope a well defined and structured model of his/her domain, which is easy to read and understand.

4 Knowledge Base of KONWERK

The knowledge base of KONWERK is divided into the following types of knowledge. They will be introduced in detail in the next subsections:

- knowledge on the domain
  - hierarchy of concepts (4.1)
  - compositional hierarchies (4.2)
  - conceptual constraints (4.3)
- knowledge on objectives
  - conceptual objectives (4.4)
  - optimization tasks (4.5)
- knowledge on intended solution procedure
  - strategies (4.6)

4.1 Hierarchy of Concepts

The first step of building the knowledge base of a specific domain is always the definiton of concepts. All concepts the modeler want's to work with must become part of the taxonomic hierarchy. The root concept of the taxonomic

hierarchy is predefined and named "object". Any further concept of the domain is more specific than "object" and consequently at a deeper level of the taxonomic hierarchy. Figure 4 shows a (simplified) taxonomic hierarchy of the Nitra River Case.

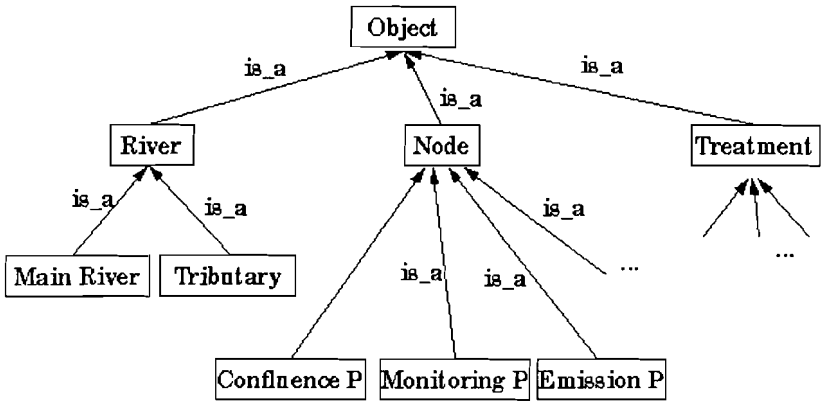

Figure 4: Hierarchy of Concepts

The object descriptions of a KONWERK knowledge base are represented in frames. The syntax used for the knowledge base of KONWERK is close to the syntax used in CLOS [3]. The paper is addressed to people who want to see and understand the ideas and a part of the implementation of KONWERK but not to show nice graphical user interfaces. Therefore, in this paper definitions of the knowledge base will be shown in their original form. However, the definition of concepts can be supported by a graphical user interface. The user (the model builder) does'nt need to learn the syntax shown in this paper.

The following example is the definiton of the concept "river" in the original syntax used in KONWERK:

```
(def-concept
  :name      river
  :super     object
  :parameter ((length   [0 100km])
              (node_num [0 100])
               ...)
   ...)
```

This definition can be read as follows:
The concept river *is an* object (the super concept is object) which has two parameters length and node_num. The "length" of any river (of this domain)

[3] CLOS is the object oriented extension of the computer language Lisp ([Ste90])

is restricted to be between 0 and 100km. The parameter node_num can hold any number between 0 and 100.

The taxonomic hierarchy has to be constructed by defining all new concepts (in our example the concepts "river", "main_river", "tributary", "node" ,...). The definition of a concept must consist of a name and the super concept. Furthermore, it is possible to assign attributes (parameters) to the objects (e.g. the length of the river or the number of nodes along the river). The super concept defines the position of the new concept in the taxonomic hierarchy.

4.2 Compositional Hierarchies

In the taxonomic hierarchy only the concepts themselves are defined but not the relations between these concepts. A compositional hierarchy expresses the compositional structure of a concept. In the Nitra River Case we have to declare that rivers may have tributaries (see figure 5), each node of our domain is belonging to a river (see figure 6) and each treatment object is a treatment of exactly one emission point.

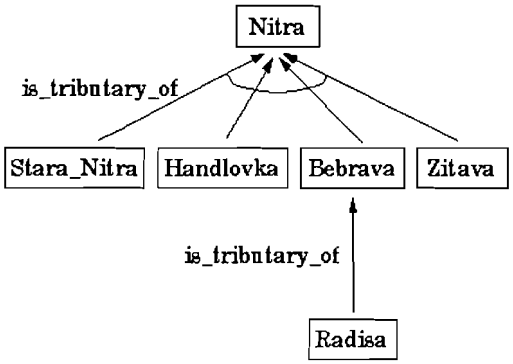

Figure 5: Tributaries of River Nitra

The implementation of compositional hierarchies has to be done in two steps. The first step is the definition of the compositional relations:

```
(def-relation
  :name     has_nodes
  :inverse  is_node_of
  :type     has_parts_relation)
```

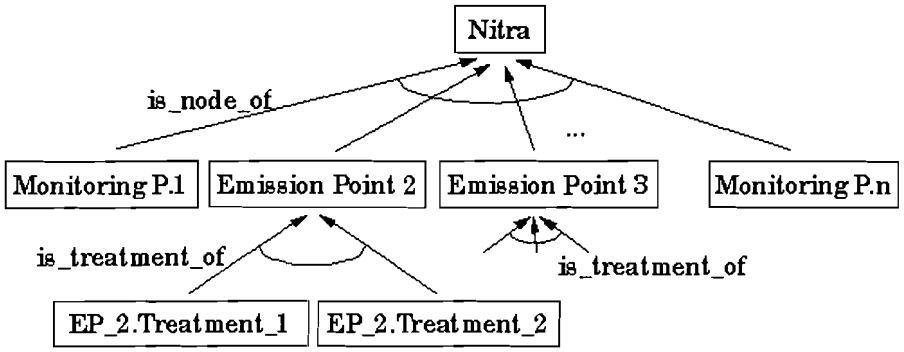

Figure 6: Compositional Hierarchies of the Nitra River

```
(def-relation
  :name    has_tributaries
  :inverse is_tributary_of
  :type    has_parts_relation)
```

Now, the definition of any object can include a declaration of a has_nodes or a is_node_of (has_tributaries or a is_tributary_of) relation to another concept. The second step of constructing compositional hierarchies is to add this information to the definition of the river as follows:

```
(def-concept
  :name      river
  :super     object
  :parameter ((length          [0 100km])
              (node_num        [0 100])
               ...)
  :relations ((has_nodes       [(a node) 0 100])
              (has_tributaries [(a tributary) 0 10])))
```

With this new definition we have stated that a river may have from zero to 100 nodes. At the same time it may have between zero and ten tributaries which belong to the river.

The parameter "node_num" is intended to hold the number of nodes which are at a river. However, "node_num" is just a name and the knowledge that the parameter "node_num" has to be the number of nodes of the river is not explicitly declared. In order to express such relations between parameters of an object or between parameters of different objects we can use conceptual constraints which will be introduced now.

4.3 Conceptual Constraints

The conceptual constraints of KONWERK can express relations between different parameters of one or more objects of the domain. The constraints are called "conceptual" because they are defined at the level of concepts. During the computations these constraints will be applied and taken into account for all objects of concepts which are mentioned in the definiton of the conceptual constraint. [4]

In general, the definition of conceptual constraints consists of a name, a set of object pattern and a formula or a set of formula. The set of pattern is used to identify the objects which have to fulfill the defined relation. The formula describes the relation itself.

A constraint, guaranteeing that the number of nodes of a river is always equal to its parameter node_num, could be formulated as follows:

```
(def-conceptual-constraint
 :name       set_node_number
 :patterns   ((?n :name node))
 :formula    "?n.node_num = card(?n.has_nodes)")
```

In this definiton "?n" is a kind of variable that can hold any instance of the concept "node". Then, an equation for the number in the parameter slot "node_num" and the cardinal number of entries in the "has_nodes" slot is formulated. This equation has to be fulfilled for all instances of the concept "node".

We give another example. The following conceptual constraints ensure, that the value of the parameter DOmin (BODmax, NH4max) of the main_river is always less or equal (greater or equal) than the oxygen concentration at any monitoring point along the river or its tributaries. We define two constraints. The first one guarantees that the DOmin values of all rivers are less or equal than any oxygen concentration (aq_DO value) of their monitoring nodes. The second constraint ensures that the DOmin value of any river is less or equal than the DOmin values of its tributaries. Hence, the DOmin value of the main_river becomes the smallest value of all rivers of the river system.

```
(def-konzeptuelles-constraint
 :name       water_quality_of_rivers
 :patterns   ((?riv  :name river)
              (?mp   :name monitoring_point
                     :relations ((is_node_of ?riv))))
 :formula    ("?riv.DOmin <= ?mp.aq_DO"
              "?riv.BODmax >= ?mp.aq_CBOD"
              "?riv.NH4max >= ?mp.aq_NH4")
```

In this constraint the pattern "?riv" represents any river of the domain and "?mp" is a monitoring_point of that river "?riv". This constraint is

[4] Similar to the use of the notion "object" in object oriented computer languages (where the instances of classes are called objects) we use "object" for the instances of concepts.

valid for all river-monitoring_point-pairs of the domain. There are three formulas given. All formulas will be taken into account for each river-monitoring_point-pair of the domain.

Now we make sure that the value of the parameter DOmin (BODmax, NH4max) of any river is always less or equal (greater or equal) than the DOmin (BODmax, NH4max) values at the tributaries:

```
(def-conceptual-constraint
 :name      water_quality_variables_main_river
 :patterns  ((?riv  :name      river)
             (?trib :name      tributary
                    :relations ((is_tributary_of ?riv))))
 :formula   ("?riv.DOmin  <= ?trib.DOmin"
             "?riv.BODmax >= ?trib.BODmax"
             "?riv.NH4max >= ?trib.NH4max"))
```

Here, the pattern "?riv" represents any river of the domain whereas "?trib" is a tributary of that river "?riv". The constraint is valid for all river-tributary-pairs of the domain even for tributaries and their tributaries.

4.4 Conceptual Objectives

The definition of objectives is very similar to the definition of constraints. The only difference is the need to specify the desired direction of the optimization (minimize or maximize). As an example we consider the problem of maximizing the minimum of oxygen concentrations (DOmin) for the monitoring nodes. Thus, we have to set up a Maximin problem. In the last section we already specified a constraint that restricts the DOmin value of a domain object main_river to be less or equal than the oxygen concentration of any monitoring_point of the river system. Now we want to introduce a corresponding objective.

We want to use the objective later also for a multi objective task (see section 5). Then, the objective value of the single objective optimization (maximizing the DOmin parameter) can be also used as the utopia value of DOmin in a multi objective task. Therefore, we introduce an additional parameter utopia_DOmin for main_river. We define a further constraint that restricts the parameter utopia_DOmin to be less or equal the DOmin value of main_river:

```
(def-conceptual-constraint
 :name      restriction_of_utopia_DOmin
 :patterns  ((?mr :name main_river))
 :formula   "?mr.utopia_DOmin <= ?mr.DOmin")
```

Now we can define the following objective:

```
(def-conceptual-objective
 :name        Maximise_DOmin
 :patterns    ((?mr :name main_river))
 :direction   :max
 :formula     "?mr.utopia_DOmin")
```

This definition can be read like the definitions of the conceptual constraints. The formula of the definition consists of only one term. It is a single expression in this example: the parameter "utopia_DOmin" of the main_river.

All six objectives of the Nitra Case (see page 3) are defined in a similar manner. These objectives can be used for the optimization of single objectives as well as for multi-criteria approaches like the reference point method [Mak95]. However, in order to run an optimization within KONWERK, an optimization task has to be defined.

4.5 Optimization Tasks

The definition of an optimization task in KONWERK allows the user to assign several requests and informations to an intended optimization problem. We consider the computation of the utopia_DOmin value. At first we define a single objective optimization task, but we want to know the utopia value in order to use it as input value for a multi criteria task, later.

```
(def-param-problem
 :name           determine_utopia_DOmin
 :objectives     Maximise_DOmin
 :assign-values  (:slots (utopia_DOmin)
                  :upper_bounds (nadir_DOmin)
                  :lower_bounds (nadir_BODmax)
                                (nadir_NH4max)
                                (nadir_TAC)
                                (nadir_inv_cost)
                                (nadir_op_cost))
 ... )
```

The optimization task "determine_utopia_DOmin" uses the previously defined objective "Maximise_DOmin". We define the optimization task in a way that only the value of utopia_DOmin will be set to the value which was found by the MIP solver. All the other variables (which are computed by the solver) will not be stored. This makes sense because we are just interested in the utopia value of DOmin. We do not need to know which policy is leading to this result.

The values computed by the MIP solver for the nadir values [5] can be used as new upper or lower bounds for the real nadir values.

[5] The nadir parameters of main_river are defined similar to utopia values. For instance, nadir_DOmin is greater or equal to DOmin.

Further entries in the optimization task definition can include a suggestion for a solver which has to be used, specific parameters for the solver programs or a list of conceptual constraints which have to be taken into account for the optimization (if not specified, all defined conceptual constraints are used by default).

It is possible to define several optimization tasks for one domain. This can be used to generate a sequence of optimizations. In our case, we can calculate the single objective tasks for all objectives and continue with the multi objective task taking the previously computed results into account. In order to do so, we have to define in which order the optimization tasks have to be processed in the session. This has to be done by strategies and will be shown in the next subsection.

4.6 Strategies

The original idea of KONWERK is to provide a frame work for defining and solving configuration tasks. Configuration processes usually base on the idea of decomposition of complex configuration tasks into smaller subtasks, which are easier to work out. The subtasks may be further decomposed themselves. Strategies are used to control the configuration process (decision making process).

In the KONWERK basic modules there are four different types of decision tasks distinguished:

- Decomposition of a (complex) object into its parts
- Integration of objects
- Specialisation of objects
- Specification of a parameter of an object

The optimization modules of KONWERK add another type:

- Optimization task

KONWERK puts all *single decisions* which have to be performed in the decision process on an agenda. The agenda entries are relating to one of the introduced types of decision tasks. They refer to a particular object, parameter or optimization task. An agenda entry could for instance represent the specialisation of a particular node object or the decomposition step of assigning a treatment object to a has_treatments slot of a nodes treatments of a emission_point (figure 7). [6]

[6] In the Nitra River Case we only need Parameter_Specification and Optimization because all objects of this particular domain (rivers, nodes and treatments) are fully specialised and their component structure is completely determined from the beginning. For instance, it is not part of the decision process to determine whether a particular node of the domain is a monitoring point or an emission point.

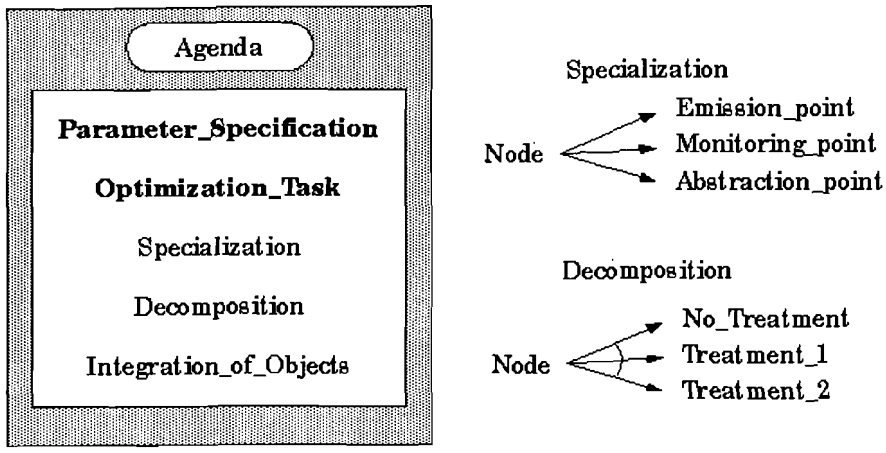

Figure 7: Types of agenda entries

Strategies specify the decisions for a specific part of the decision process. They can also be used to define an order of these decisions and to indicate a subset of methods, which may be applied to make a single decision [Gün92].

The following example is a strategy from the Nitra River Case:

```
(def-strategie
 :name            determine_utopia
 :strategy-class  depth-first-search
 :focus           (:type      (optimize)
                   :opt-task  (determine_utopia_DOmin
                               determine_utopia_BODmax
                               determine_utopia_NH4max
                               ... ))
 :order           ((:opt-task (determine_utopia_DOmin))
                   (:opt-task (determine_utopia_BODmax))
                   (:opt-task (determine_utopia_NH4max)))
 :methods         (optimization function dynamic-default
                   static-default user-query)
 :priority        250)
```

The option "focus" defines a filter for the agenda of KONWERK. As long as the strategy is active, only those tasks will be executed, which satisfy this filter. In the strategy given above, only tasks of the "type optimize", named determine_utopia_DOmin, determine_utopia_BODmax and so on, will be done. All other tasks will not be executed in this strategy.

The option "order" defines an order of the single tasks, which are matching

the focus. At first, all optimization tasks called "determine_utopia_DOmin" are executed. Then, the tasks with the name "determine_utopia_BODmax" follow and so on. [7]

A set of methods can be indicated by using the "methods" option. This set of methods is allowed to be used within that strategy. However, a method can only be applied for a decision if the needed knowledge is present (e.g. a default value must be specified for a parameter in order to use the static-default method for a parameter_specification task).

The "priority" value is used during the solution process for selecting a strategy. The strategy with the highest priority value will be performed at first.

5 Modeling the Reference Point Method

In the Nitra River Case six objectives are to be taken into account at the same time (see section 2). The resulting multi-criteria problem can be successfully handled by the Reference Point Method ([Mak95], [Steu90]). The utopia and nadir values can be used as initial aspiration levels. Utopia values are computed by selfish optimization of the six single objectives. The nadir values can not easily be determined. However, the nadir point will be used as an initial reservation point only. Hence, it is not necessary to spend to much effort on computing a very good approximation of this point. The vector of the worst values of the single objectives (derived from the selfish optimizations) is used in the implementation as a rough approximation of the nadir point.

The following table shows the results of selfish optimization of the six objectives using the KONWERK implementation of the Nitra River Case. The utopia values are close to the results documented in [Mak95]. [8] The row called RPM shows the first result of the Reference Point Method using the computed utopia and nadir values as aspiration respectively reservation levels for the six criteria.

Using the strategies described in section 4.6 it is possible to express even complex methodologies like the Reference Point Method in KONWERK. For the Nitra River Case this method was implemented by B.Funke using only the existing KONWERK modules.

Following the idea of making model building as easy as possible, it would be better to give the opportunity to the model builder, just to declare that the Reference Point Method has to be used for working on a given multi-criteria

[7] In general it is possible to have several optimization tasks with the same name: The objectives are defined at the conceptual level. Therefore, they may have many instances. An optimization task defined for such an objective will be instantiated as often as its objective is instantiated.

[8] Remaining small differences of these values seem to be due to differences of the model description in the working paper [Mak95] and the real implementation of the core model.

Minimized Criterion	Values of DOmin [mg/l]	BODmax [mg/l]	NH4max [mg/l]	TAC 10^6 $	InvCost 10^6 $	OpCost 10^6 $
DOmin	5.37	10.1	1.81	17.4	61.3	10.2
BODmax	0	10.1	1.81	17.4	61.3	10.2
NH4max	0	10.1	1.81	17.4	61.4	10.2
TAC	0	31.8	8.02	1.05	1	0.93
InvCost	0	25.7	4.65	3.54	0	3.54
OpCost	0	31.8	8.02	8.02	1	0.93
utopia	5.37	10.1	1.81	1.05	0	0.93
nadir*	0	31.8	8.02	8.02	61.4	10.2
RPM	3.55	17.7	3.52	6.80	22.0	4.22

Table 1: Pay-off table for six criteria of the Nitra River Case

problem. In this case it would neither be necessary to write constraints in order to represent the achievement functions nor to implement a strategy which allows to iterate over a sequence of user provided aspiration and reservation points. Therefore, though it can be implemented by strategies it makes sense to implement additional specific routines for KONWERK generating the desired behavior in an easier way.

6 Conclusion and Outlook

In this paper we reported a knowledge-based reimplementation of the Nitra River Case using the KONWERK tool-box. Though KONWERK is not a professional tool and exists as a prototype version only, the Nitra River Case was implemented in a rather short time and adjustment to model changes and the enlargement of the model was easily realized several times.

The success of this experiment (using the knowledge-based modeling components of a configuration tool-kit for implementing a optimization task which is not a typical configuration task) can be seen as a hint to the potential of the knowledge-based modeling approach in general. The advantages of knowledge-based modeling can be summarized as follows. The knowledge-based model is

- easy to develope
- easy to read and understand
- easy to change (adjustment to new situations)

Therefore, overcoming some remaining weaknesses of KONWERK, like missing of a good graphical user interface and further development of specific multi-criteria optimization approaches (e.g. Reference Point Method) could lead to an even more effective tool for model building and evaluation. In addition, the integration of knowledge-based model building and developing a core model of a specific domain (see [Mak94]) is a base for further model investigation.

References

[Ang95] A. Angelus, Ch. Bohley, and A. Kathe:
Konfigurieren von hydrogeologischen Modellen (in german). in Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt Prokon, A. Günter, Ed. Sankt-Augustin: Infix, 1995

[Ber93] R. Berkemer, M. Makowski, and D. Watkins:
*A Prototype of a Decision Support System for a River Basin Quality Managment in Central and Eastern Europe.*Working Paper WP-93-049, International Institute for Applied Systems Analysis, Laxenburg, 1993

[Fun94] B. Funke, K. Müller, H.-J. Sebastian and M. Thärigen:
Innovative Design by Integration of Expert System Technology with Multi-Criteria Decision Making and Fuzzy Systems., 1992

[Fun95] B. Funke:
Einsatz von Optimierungsverfahren und Modellierung von Optimierungsproblemen (in german). in Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt Prokon, A. Günter, Ed. Sankt-Augustin: Infix, 1995

[Gün91] A. Günter, H. Dörner, H. Gläser, B. Neumann, C. Posthoff, and H.-J. Sebastian:
*Das Projekt PROKON: Problemspezifische Werkzeuge für die Wissensbasierte Konfigurierung (in german).*PROKON-BERICHT Nr. 1, BMFT Verbundprojekt PROKON, 1991

[Gün92] A. Günter , R. Cunis:
*Flexible Control in Expert Systems for Construction Tasks.*Journal of applied Intelligence 2,369-385, 1992

[Gün95] A. Günter, L. Hotz, Th. Vietze:
Die Basis-Module von KONWERK (in german). in Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt Prokon, A. Günter, Ed. Sankt-Augustin: Infix, 1995

[Hotz95] L. Hotz, Th. Vietze:
Erweiterung der Begriffshierarchie um Sichten und Mehrfachvererbung (in german). in Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt Prokon, A. Günter, Ed. Sankt-Augustin: Infix, 1995

[Kar90] W. Karbach, M. Linster:
*Wissensakquisition für Expertensysteme: Techniken, Modelle und Softwarewerkzeuge (in german).*Munich, Vienna: Hanser, 1990

[Kop95a] M. Kopisch:
Konfigurieren von Passagierkabinen des Airbus A340 (in german). in Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt Prokon, A. Günter, Ed. Sankt-Augustin: Infix, 1995

[Kop95b] M. Kopisch:
Repräsentation und Auswertung von räumlichem Wissen (in german). in Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt Prokon, A. Günter, Ed. Sankt-Augustin: Infix, 1995

[Mak94] M. Makowski:
*Methodology and Modular Tool for Multiple Criteria Analysis of LP Problems.*Working Paper WP-94-102, International Institute for Applied Systems Analysis, Laxenburg, 1994

[Mak95] M. Makowski, L. Somlyódy, and D. Watkins:
*Multiple Criteria Analysis for Regional Water Quality Management: the Nitra River Case.*Working Paper WP-95-022, International Institute for Applied Systems Analysis, Laxenburg, 1995

[Schu95] S. Schumann, O. Schumann:
Modellierung von Unschärfe in KONWERK (in german). in Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt Prokon, A. Günter, Ed. Sankt-Augustin: Infix, 1995

[Seb94] H.-J. Sebastian:
*Optimization-based Control-Strategies.*in Expert Systems for Design and Configuration; Proceedings of the 3'rd OCAMD-Conference of IFIP WG 7.6., Prague 94, Lansa Pub. BV, to appear

[Ste90] G.L. Steele:
*Common Lisp the Language, Second Edition.*Digital Press, 1990

[Steu90] R.E. Steuer, L.R. Gardiner:
*Interactive Multiple Criteria Programming: Concepts, Current Status, and Future Directions.*in Readings in Multiple Criteria Decision Aid, C.A. Bana E. Costa, Ed. Springer , 1990

[Thä95a] M. Thärigen:
Layout eines Logistiksystems - Standort- und Routenoptimierung mit KONWERK (in german). in Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt Prokon, A. Günter, Ed. Sankt-Augustin: Infix, 1995

[Thä95b] M. Thärigen:
Wissensbasierte Erfassung von Anforderungen (in german). in Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt Prokon, A. Günter, Ed. Sankt-Augustin: Infix, 1995

[Wie94] Wierzbicki, A.:
Multicriteria Aid of Intuition in Decision Making., 1994

[Zim94] H.-J. Zimmermann, H.-J. Sebastian:
*Fuzzy Design - Integration of Fuzzy Theory with Knowledge-Based System Design.*in: Proceedings of the World Congress on Computational Intelligence, Fuzzy-IEEE 94, Orlando, Florida, Juni 1994