Environmental Risk Assessment Support System Based on Multiple Domain Knowledge

by

Yasunobu MAEDA

April 1, 1994

A DISSERTATION
submitted to the University of Tsukuba
in partial fulfillment of the requirements for the degree

DOCTOR OF PHILOSOPHY
in
Urban and Regional Planning

DOCTORAL PROGRAM IN SOCIO-ECONOMIC PLANNING
THE UNIVERSITY OF TSUKUBA

Abstract

I have developed the knowledge system RARCOM (Risk Assessment and Risk COMmunication support system), which aims to support risk assessment and risk communication about an environmental risk, carcinogenicity of drinking water in urban water networks. The system holds a set of knowledge bases involving the multidisciplinary object model that represents such a variety of phenomena as environmental sources of risks, process of risk propagation, and exposure by the risk factors involved in the urban water network.

To deal with interdisciplinary nature of the environmental risk, the *composite object model*, a modeling framework to construct object models on the basis of multiple domain knowledge, is developed. This modeling framework utilizes not only multiple domain knowledge, but also bridging or interpreting knowledge over gaps among domains. The composite object model gives the foundation to construct the knowledge bases in the RARCOM on multiple domain knowledge; such as hydrology, waterworks engineering, administration, etc. The knowledge bases are described by object-oriented programming and logic programming. By using the composite object model, the RARCOM can execute knowledge-based simulation to support risk assessment, also enable users to examine easily meta-information related to modeling procedures, which has been embedded in models.

This thesis is based on two papers published on journals that have referee systems. Chapter 2 and 3 are mainly based on [Maeda92], simultaneously, Chapter 4 and 5 are based on [Maeda93].

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Environmental Risk Problems

The environmental risk is defined as a kind of environmental problem that has the following uncertain characteristics: difficulty of identification and low reliability of measurement of contaminants, no clear existence of deterministic causal relationships, and low quantitative predictability of estimated risks[Whyte80, Ikeda89]. The environmental risk problems, for example; ozone layer destruction, pollution by radioactive wastes, and health effects by hazardous chemical substances, are typical ones that occur in wider spatial areas and in longer periods by a smaller amount of contaminants than conventional environmental problems(e.g., air pollution by sulfoxide or smokes), which have direct sources and are transmitted to particular areas in short periods by a large amount of contaminants.

To deal with these environmental risk problems, various methodologies of risk analysis have been developed for the past decades. The typical ones are risk assessment and risk communication. Risk assessment is a procedure to identify source and pathway of risk propagation, and to estimate the exposed dose of contaminants and its health effects. Risk communication is a type of two-way communication between experts, citizens, and municipal administrators about risk information, such as; nature, magnitude and urgency of handling risks.

Two reasons, however, obstruct risk assessment and risk communication procedures. First, risk propagation processes are very complex and involve many different mechanisms and media, therefore, risk assessment procedures become burdensome for the assessors because they must deal with a variety of categories and a large number of data with different levels of uncertainties as mentioned before. Second, environmental risk phenomena have an interdisciplinary nature, namely, they are associated with people based on many different standpoints and a variety of disciplinary knowledge, that is, domain knowledge. To understand source generation, propagation, and exposure of risks, a wide set of domain knowledge is required. However, each domain knowledge involves distinct terminologies, formalisms, and theories; consequence of risk assessment is hence difficult to understand for citizens or municipal administrators. Even between experts in each domain, some terminologies sometimes may cause a trouble, because of the different interpretations that each expert has.

Accordingly, we need a system of supporting environmental risk assessment and risk communication that can deal with such a variety of and a number of data and contribute to solve such interdisciplinary issues of environmental risk assessment and risk communication. To construct such system, knowledge engineering, an application technology of artificial intelligence, is expected to be effective. This is a field involving researches to let computers perform reasoning similar to human knowledge processing and understand structure or representation of knowledge. Expert systems that are intended to perform reasoning that is similar to reasoning done by domain experts are remarkable results of knowledge engineering.

The system of RARCOM, which stands for Risk Assessment and Risk COMmunication support system, is constructed by using knowledge engineering technology[Maeda92, Maeda93]. The system's aims are to help decision makers and the public involved in the risk problems;

1. to assess risks by processing related knowledge, and
2. to communicate the risk information between related people by displaying the information.

For these aims, the RARCOM mainly uses the knowledge-based system of risk problems. The knowledge base consists of the object model representing environmental processes of risk generation, propagation, and exposure of the hazardous chemicals, and the rule base of knowledge to evaluate human impact by exposure of the chemicals. Knowledge-based simulation based on a constraint solving algorithm is implemented in the system to estimate the risk generation, propagation, and exposure processes. The Common ESP(Extended Self-contained Prolog), which is a computer language based on logic programming and object-oriented programming, is employed to implement the RARCOM in the computer system.

Water contamination by hazardous chemicals, especially trihalomethanes (THMs), which are carcinogens generated in the disinfection process of drinking water, are taken as a water risk problem in a large-scale metropolitan area. Tokyo metropolitan area is a typical one that has an interdisciplinary nature of carcinogenic risk generations and propagation as well as the structural complexity of the water network. The RARCOM deals with the THM problems in Tokyo metropolitan area.

To deal with the interdisciplinary nature of the environmental risk problem, the *composite object model*, a modeling framework to construct object models on knowledge base of multiple domain knowledge, is developed. For example, modeling of the THM problem requires a variety of domain knowledge; a model of rivers is described by using terms in hydrology and water environment engineering, while a model of filtration plants is done by using terms of waterworks engineering. In the two models, behavior of these is described by different terminologies respectively. For instance, water quality indices used in these models are different with each other. The composite object model enables modelers to construct models by integrating knowledge of those multiple domains and by bridging over different terminologies in various domains.

The RARCOM is intended to be used primarily by decision makers and domain experts associated with the problem. However, domain knowledge bases involved in the system, which are based on the composite object model, are designed as highly portable and reusable multipurpose knowledge bases. In other words, domain knowledge bases in the composite object model can be reused for other systems to deal with other problems related to the domains. The composite object model will be able to be a foundation of organizing and integrating knowledge related to various environmental problems.

In the past decade, a number of researches have been made on the construction of such systems as the RARCOM. In the field of environmental science, Fedra and Winkelbauer[Fedra91], and Fiksel and Covello[Fikse87], constructed knowledge systems to support environmental impact assessment or decision making for environmental policies. Further, *Eco-Logic* project[Robert91] in the Edinburgh University developed several knowledge systems for the modeling of ecological systems. However, the former two did not have explicit model of environmental systems in their knowledge bases, at the same time, the latter could not cope with the interdisciplinary nature of the environment, because these systems specialized to describe zoo-ecological systems. They did not yet discuss the importance of knowledge-based model involving environmental multiple domain knowledge to understand phenomena in the environment.

In the field of knowledge engineering, Zeigler[Zeigl84], and Kiriyama, Tomiyama, and Yoshikawa[Kiriy91], proposed modeling methods in which a model can be constructed on the basis of many different standpoints. The project CYC[Lenat90], the project EDR[Yokoi90], and Miki et al.[Miki93] devote to develop knowledge-based systems that involve knowledge of very large size and various domains to solve complex problems. However, the first four did not deal with issues related to terminology of multiple domain knowledge. Although Miki et al. proposed a framework to construct knowledge systems based on multiple domain knowledge and to deal with the terminology of multiple domains, their framework has not yet been implemented as a working system. In contrast to them, this paper proposes the composite object model, which is implemented in the RARCOM, as another framework that can cope with such issues.

1.2 Terminology

This section defines terminology used in this paper, which is particularly related to models and constructing models.

Definition

object In this paper, the term 'object' is used in two ways. First, It is used to represent an entity in the world that is the target to be modeled. Second, It is done as a program module in *object-oriented programming*(See Section 3.3). In the remainder of the paper, the term used in the later sense is denoted as *object.*

model Formal representation of mechanism and behavior of objects to be investigated. In this paper models of 'cities', 'basins', 'environment', etc., are argued.

modeling schema Way to divide the object to submodels or components and to articulate submodels or components of the model. For example the grid system is a schema to model land. On the other hand the schema based on municipal administration may model land as a system of polygons representing cities, towns, and wards, which is another modeling schema for land.

modeling framework Methodology to construct the model. It includes a modeling schema, representation of structure of the model, and method to describe behavior of the model.

The remainder of this paper is structured as follows. Chapter 2 discusses requirements to support risk assessment and risk communication. Following it, features of the knowledge system supporting risk assessment are described in Chapter 3. Chapter 4, the core of this paper, describes the composite object model and how it is used to construct the object model of the urban water network. Finally, Chapter 5 shows implementation of system RARCOM. Features for supporting risk communication are explained in the chapter.

Chapter 2

Risk Assessment and Risk Communication for Trihalomethane Problems

This chapter shows characteristics of the THM problem, and specifies system's requirements to support risk assessment and risk communication. Necessary features for the knowledge system to support risk assessment and risk communication in the THM problem are derived from these arguments.

2.1 Characteristics of Trihalomethane Problems

THMs are halogen substituted, single-carbon compounds having the general formula CHX_3, where X may be fluorine, chlorine, bromine or iodine, or their combination. Chloroform($CHCl_3$) is the most commonly encountered of these chemical substances[WHO84].

Some researches pointed out carcinogenicity and mutagenicity of these substances. Since Rook[Rook74] reported that THMs are formulated through disinfection process for drinking water, carcinogenicity of THMs has become a problem that water service administrators cannot ignore.

THMs occur in water primarily from reactions between chlorine and organic compounds such as humin acid. These organic compounds, which are called precursors, consist of two types[Tanbo83]. The first are substances naturally produced from peat or humic soils. The second are discharged from sewage treatment plants or human excreta treatment plants.

In water networks in urban areas, for instance, in Tokyo or Osaka, the latter substances are more significant because discharges from upward urban areas may be a source of the environmental risk in downward areas. For example, carcinogenicity of drinking water in Tokyo may be increased by adding urban activities in upward Saitama, similarly in Osaka, it may be done by activities in Kyoto. Namely, the THM problem in urban water networks is closely related to those urban system structures including water network topology and land use patterns.

The THM problem in urban water networks has the following four characteristics[Maeda92, Maeda93].

1. The problem has uncertainty.

   The uncertainty has two aspects, complexity and fluctuations. The first aspect is related to two factors.

   - Complexity of each process involved in the risk phenomena; such as generation of the risk, transmission of concerned substances, human exposure, and health effects.
   - Structural complexity of risk propagation.

In the THM problem, uncertainties in carcinogenic risk of drinking water emerge from complex processes involved in risk propagation; for instance, processes of domestic waste water discharges, treatment in sewage plants or human excreta treatment plants, biochemical interactions in rivers, treatment in filtration plants, and water supply for each household. Therefore, systems to deal with the problem must have capability to handle hydrological, biochemical, and technological behavior involved in each process.

The other aspect of the uncertainty is related to stochastic fluctuations of magnitude and frequency of the risk. Systems to deal with the problem, hence, have to handle parameters related to fluctuations of the risk.

2. The risk is generated and propagated on the urban system.

   The urban area is a kind of system. The risk problem, as mentioned above, occurs through the urban system structure. Moreover, cities are continuing its growth and change, which occurs not only in physical infrastructure composing the urban structure, but also in socio-cultural infrastructures, such as changing life style of citizens or introducing new material circulations in the urban area. Accordingly, description of the urban system involving the system structure and its socio-cultural system is needed to deal with the risk propagation process.

3. The risk problem is a kind of design problem while it is also a diagnosis problem.

As shown in the next section, the risk assessment is a procedure to estimate magnitude and distribution of the generated risk. This is considered as a kind of diagnosis procedure. However in a broader sense, risk assessment has to be done in connection with risk management procedure to find policies or actions to reduce the risk. Namely, combined process of risk assessment and risk management is necessary for making desirable policy that reduces existing risk. In this sense the risk problem is considered as a design problem for policy making.

In policy making processes it is required to assess effectiveness of designed policy variables(for example, construction of water supply/sewer systems, regulatory actions, and institutional improvements). Hence, decision makers have to go back to risk assessment and to validate effectiveness of the alternative policy to be designed. Namely, such policy makings require cyclic process between risk assessment and risk management.

4. The THM problem is interdisciplinary.

   The THM problem has interdisciplinary nature. For instance, let us consider effects of urbanization in an upper basin in a river system. A progress of urbanization in the upper basin may cause an increase of THM generation on a lower basin, because the urbanization in the upper basin brings an increase of discharge from sewers in the upper basin. This causes organic contamination in the source of drinking water that is to be supplied for the lower basin cities.

Finally, the source-water contamination may result increase THM generation in filtration plants in the lower basin.

To understand the whole sequence of these processes, knowledge in various domains is required. First, knowledge of sewage works engineering is required to estimate impact of the urbanization on quality of sewage plant discharge. Secondly, hydrology will be suitable to describe the behavior of water in rivers. Furthermore, understanding the change of water quality in rivers requires knowledge provided by water environment engineering or water quality engineering. Thirdly, to understand processes that occur in filtration plants, waterworks engineering is needed. Fourthly, information related to citizens exposed to the risk, for example, distribution of population or distribution of gender/age in population, will be supplied by means of knowledge of municipal administration. Finally, knowledge required to estimate human effect of the substance is involved in epidemiology or social medicine.

As shown above, the interdisciplinary nature of the THM problem calls for knowledge in various domains that are related to the risk phenomena. To support risk assessment for the problem, it is required to deal with such multiple domain knowledge.

Consequently, the knowledge system that deals with the THM problem has to satisfy the following requirements.

1. To have the model of the urban system in which the risk occurs and propagates, and the people are exposed to the risk.

2. To have the model describing behavior of various processes involved in the urban system and variables related to the risk.
3. To be able to evaluate fluctuations of the risk.
4. To be able to simulate the risk phenomena or risk reduction policies by means of the model of the urban system.
5. To deal with multiple domain knowledge associated with the problem.

2.2 Requirements to Support Risk Assessment

According to The National Research Council(NRC) in the United States, risk assessment consists of the following four steps[NRC83].

1. **Hazard Identification(HI)** The determination of whether a particular chemical is or is not causally linked to particular health effects.
2. **Dose-Response Assessment(DRA)** The determination of the relation between the magnitude of exposure and the probability of occurrence of the health effects in question.
3. **Exposure Assessment(EA)** The determination of the extent of human exposure before or after application of regulatory controls.
4. **Risk Characterization(RC)** The description of the nature and often the magnitude of human risk, including attendant uncertainty.

Figure 2.1[NRC83] shows the relation among the four steps and risk management. As shown in the figure, the step RC, the final step of risk assessment, follows consequences of both the step DRA and the step EA. The

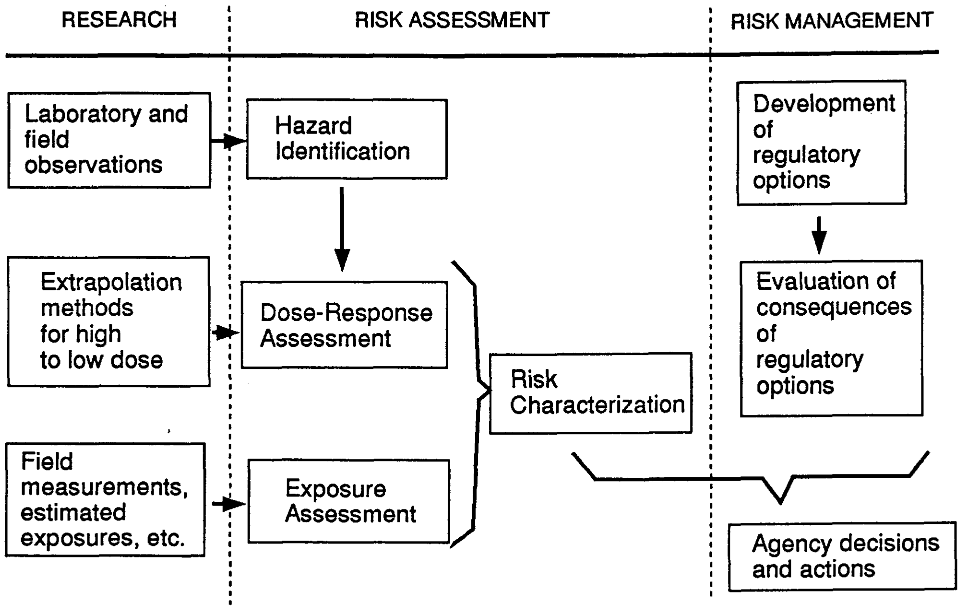

RESEARCH
RISK ASSESSMENT
RISK MANAGEMENT
Laboratory and field observations
Hazard Identification
Extrapolation methods for high to low dose
Dose-Response Assessment
Field measurements, estimated exposures, etc.
Exposure Assessment
Risk Characterization
Development of regulatory options
Evaluation of consequences of regulatory options
Agency decisions and actions

Figure 2.1: Elements of risk assessment and risk management[NRC83].

latter, the step EA, can be regarded as a kind of simulation process of exposure situations. As shown in Figure 2.1, the step EA is based on field researches that identify structure, behavior and nature of the exposure environment. The field researches are considered as an identification process of environmental model for the simulation in the step EA. The step EA requires such identification of model of the environment. Akiyama[Akiya89] pointed out that the urban system structure, which is closely related to risk propagation, is very important to understand water-borne risk in urban areas. Accordingly, adding to the four steps, identification of model of the urban area is necessary to risk assessment for such risks.

Supporting of risk assessment will, therefore, have to consist of procedures to support some part of the four steps plus a process of modeling the urban area. In this paper, I intended to support steps EA and RC on the basis of the knowledge of substances, dose-response relationship, and the structural model of the urban system. In the remainder of the paper, it is assumed that consequences of steps HI and DRA and structure of the model are given.

Risk assessment process involves a logical sequence of scientific estimates of processes; such as emissions of precursors or related substances to the environment, propagation along the structure of urban system, biochemical reactions in the propagation paths, exposure to humans, and its health effects. In such a sequence, it is critical to consider uncertainties involved in each estimate and gaps between estimates.

Furthermore, as shown in Figure 2.1, the risk assessment is also a procedure to prepare the basic data for subsequent risk management. Based on the consequence of risk assessment, risk management can determine a new policy among alternative policy options to reduce the risk. If a policy is selected, another EA step on the situation based on the new policy and the subsequent RC step must be done. Namely, in such a cyclic process of EA and RC, each alternative policy requires EA based on the scenario representing the condition and assumptions employed in the policy.

To support the EA procedure in the cyclic process, the system to support risk assessment must have the following specifications.

1. The model representing behavior and structure of environment and urban area.

2. Knowledge about dose-response relationship associated with the concerned risk.

3. Computational environment to simulate a variety of exposure situation on the model.

4. Capability to evaluate uncertainties associated with the estimated risk.

2.3 Requirements to Support Risk Communication

According to Fiksel and Covello[Fikse87], risk communication is the process of conveying to interested parties the output of the various stage of risk assessment and risk management including the nature, magnitude, urgency, and acceptability of the risk, strategies for mitigating the risk, the relative merits of different options, and the justification for the decision.

The following problems, however, obstruct desirable risk communication[Covel87].

1. Message problems

   - Deficiencies in scientific information resulting in large uncertainties in risk estimates.
   - Difficulties in assessing complex phenomena, such as accident sequences, synergistic effects, and effects for sensitive populations(e.g., children).
   - High technological analysis that is often unintelligible to the non-scientist.

2. Source problems

   - Disagreements among scientific experts.
   - Lack of resources and legal constraints.
   - Failures to disclose limitations of risk assessment.

- Limited understanding of cognitive aspects(interests, concerns, values, etc.) of the individual citizens and public groups.
- Use of bureaucratic, legalistic, and technical language.
- Lack of trust and credibility.

3. Channel problems

   - Selective and biased media reporting.
   - Premature disclosures of scientific information.
   - Inadequacies in interpreting technical risk information.

4. Receiver problems

   - Inaccurate perceptions of levels of risk.
   - Lack of interest in risk problems.
   - Overconfidence in one's ability to avoid harm.
   - Strong beliefs and opinions that are resistant to change.
   - Exaggerated expectations about the effectiveness of regulatory actions.
   - Desire and demands for scientific certainty.
   - A reluctance to make tradeoffs between different types of risks or between risks, costs, and benefits.
   - Difficulties in understanding probablistic information.

Fiksel and Covello[Fikse87] discussed that knowledge system technology could contribute to enhance risk communication against these problems in the ways shown below.

1. Generic knowledge representation

   Knowledge about risk phenomena can be organized into well-defined categories of attributes, impacts, and other important factors. Knowledge systems will make it possible to account explicitly for these considerations and to acknowledge their importance in the risk communication process.

2. Repository of consensus knowledge

   Knowledge system technology permits the electronic capture, maintenance, and distribution of knowledge. This capability will help clarify the nature of scientific controversies and disagreements about facts and assumptions.

3. Clarity of logic and assumptions

   Knowledge system technology will permit the reasoning, logic, and assumptions used in the process of constructing statements about risk to be carefully controlled and examined. For example, knowledge systems can be used to explore the sensitivity or adequacy of statements in various situations.

4. Flexible explanation capability

   Different users may require different level of information about concerned risk. Knowledge system technologies make it possible to derive as deeply as needed into the knowledge base to explore particular questions about assumption, facts, models, or theories.

5. Handling of Qualitative Information

Knowledge system technology can simultaneously accommodate both qualitative and quantitative information for a specific risk problem.

6. Natural language presentation

   Knowledge system can output information derived from reasoning on the system in style like natural language.

7. Handling of Uncertainty

   Knowledge system can treat uncertainty with tools, such as probability, certainty factors, and fuzzy measures.

8. Interactive dialog

   Knowledge systems are capable of engaging in an interactive dialog with a related group.

9. Graphic visual aids

   The use of knowledge systems with active graphic interfaces can help users understand complex subjects.

10. Rapid information retrieval

    The use of knowledge systems enables specific types of information about risks and risk management strategies to be rapidly accessed and displayed.

Among the items listed above, we are not concerned here with the item 5, handling of qualitative information, because the paper deals with support for quantitative, not qualitative, risk assessment process. Moreover, the item

10, rapid information retrieval, can be aggregated into the item 2, because the nature of information retrieval should be argued as an aspect of the information repository. Following the above discussions, our specifications for the knowledge system to support risk communication is summarized as listed below.

- Explicit and generic knowledge representation.
- Consensus knowledge base associated with the risk.
- Capability to examine the knowledge in various situations.
- Flexible explanation function.
- User interface using natural language.
- Handling of uncertainty.
- Interactive user interface.
- Graphical user interface.

2.4 Summary

From discussions in this chapter, requirements for risk assessment and risk communication for the THM problem are summarized as follows.

1. Requirements for the knowledge base.
   - To have knowledge related to risk assessment. The knowledge involves,
     - model of structure and behavior of the urban system in which the risk is generated and propagated, and
     - knowledge of dose-response relationship associated with the risk.
   - To have multiple domain knowledge needed to understand the risk phenomena.
   - To be able to evaluate uncertainties, especially fluctuations, associated with the risk.
   - To have computing environment to simulate exposure situations in various conditions.
2. Requirements for user interface of the system
   - To be interactive.
   - To have graphic visual interface.
   - To have natural language interface.
   - To be able to explain flexibly consequences and processes of inference done on the system.

The next chapter follows the first set of requirements and discusses the design of the knowledge base of the system. The second set of requirements is referenced in Chapter 5 to discuss how to implement the user interface of the system.

Chapter 3

Knowledge System to Support Risk Assessment

This chapter is devoted to specify such necessary functions of the knowledge system that satisfy the following five systems requirements discussed in the preceding chapter.

- A model of structure and behavior of the urban system.
- Knowledge of dose-response relationship associated with the THM problem.
- The knowledge base that consists of a set of knowledge concerning to multiple domains involved in the risk problem.
- Computing environment to simulate exposure situations on the model in the knowledge base.
- Capability to evaluate uncertainties, especially fluctuations, associated with risk estimating processes.

Section 3.1 discusses the first requirement. The third requirement is discussed in Section 3.2. Following these two sections, Section 3.3 discusses how the model is programmed in the knowledge base. Section 3.4 shows how the RARCOM system cope with the fourth requirement. Finally, the fifth is discussed in Section 3.5. How the second requirement is dealt with is discussed in Chapter 5.

3.1 Object Model

As mentioned above, the system must have a model representing structure and behavior of the object, in which the risk of exposure to the carcinogenic substance is generated and propagated, and knowledge about dose-response relationship of the carcinogen. The latter is probably written in a *rule-based system*[Winst84]. The former is a kind of knowledge that is called *object model* in knowledge engineering[Ueno85], which is a formal representation of some people's mechanism-understanding about the object. This section focuses on the former knowledge and discusses how the object model should be to deal with the THM problem in urban water networks.

There are two approaches, *discovery approach* and *postulated approach*, to model an object in the world[Klir91]. In the former approach, the model is constructed by bottom-up procedure based on data observed in the world. At the same time, the model can be built by top-down procedures based on known premises in the latter approach. Object model in the RARCOM is constructed along the latter approach, because it follows, as mentioned in Section 2.2, the premise that identification of the model must be done in advance.

When an object in the world is mapped onto a model, the mapping process

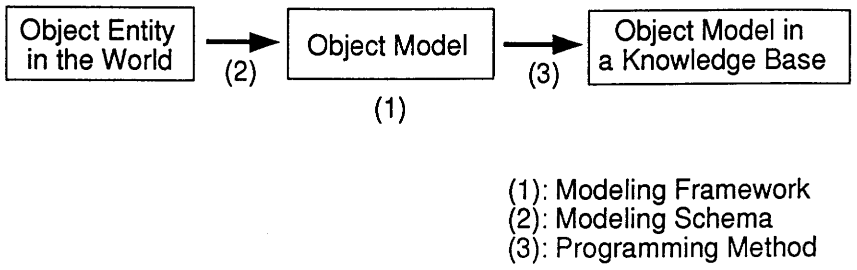


Figure 3.1: Procedures to model an object in the world.

should follow the procedure shown in Figure 3.1[Maeda92]. The object is first mapped onto the object model on the basis of a modeling schema((2) in the figure), Secondly, the object model is described in a knowledge base by using a programming method(3). This section discusses the modeling schema that is used in the object model description. The modeling framework(1) is argued in the next section. In particular, the framework to deal with multiple domain knowledge is fully discussed in the next chapter. Section 3.3 shows the programming method to describe explicitly and comprehensively knowledge associated with the risk.

Component-conduit model

The system uses *component-conduit model* as the modeling schema to represent the object model. This is the same schema that de Kleer and Brown[deKle83] used in modeling for qualitative physics. In component-conduit model, the object is modeled as a network consisting of a number of sorts of *components* and one sort of *conduits* that connect between components. This network is called *device topology*, which represents the structure of media that prop-

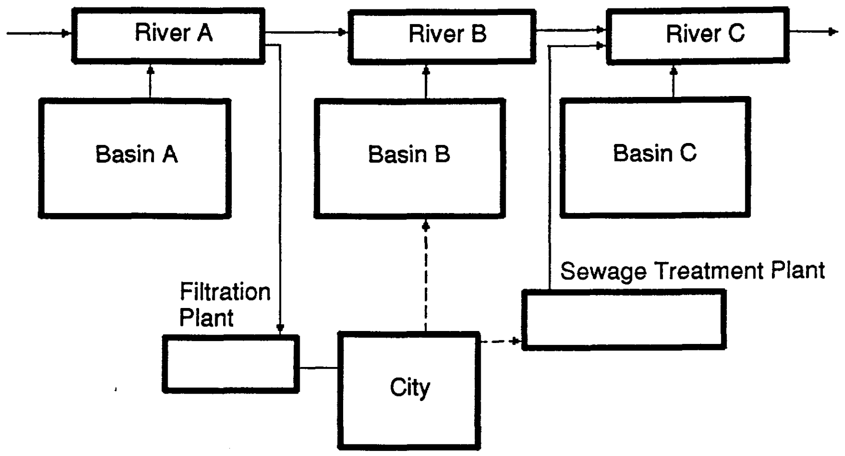


Figure 3.2: A component-conduit model of an urban water network.

agate *stuff.* Each component in the topology varies attributes of the stuff when the stuff passes through it, while conduits have no effect on attributes of the stuff.

For example, a model of an electric circuit is represented, as shown by de Kleer and Brown, as a device topology that involves components representing resistor, condenser, switch, and so on. On the other hand, stuff of the model represents electric current.

In this paper, components of the model are river channels, river basins, filtration plants, sewage treatment plants, cities, towns, and wards (in the remainder of the paper the latter three are aggregated and denoted as 'cities'). The device topology involving these components represents the urban water network. Figure 3.2 shows an example of a device topology representing an urban water network.

The claim that an urban water network can be modeled as a component-

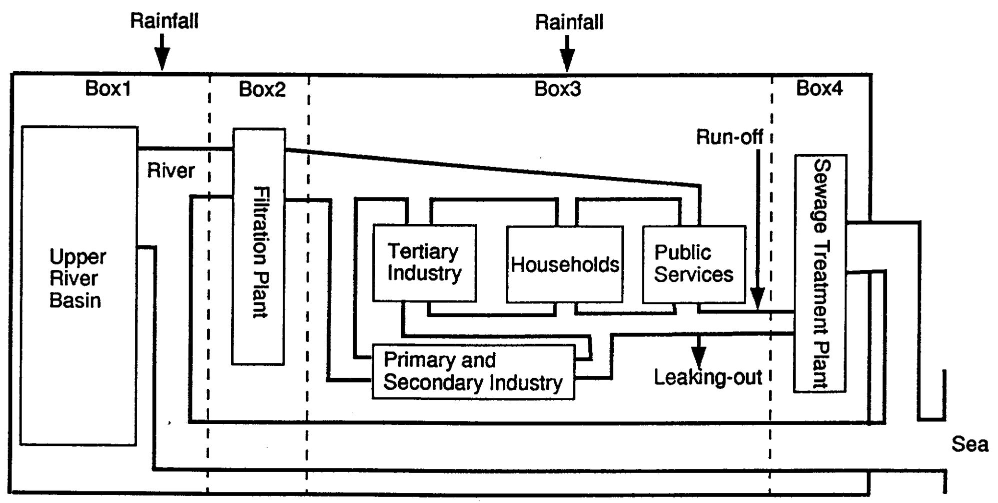


Figure 3.3: Conceptual model of the artificial water cycle[Akiya89].

conduit model is based on the empirical works by Akiyama[Akiya89] and others. Akiyama propounded for risk analysis a conceptual model of urban structure associated with the water cycle. In this model, urban water system is structured by boxes that have distinct functions and nature and are connected with each other by water flow(Figure 3.3). This model is based on the premises listed below.

- Water flows through each box from its upward side to its downward side.
- Each box varies rate of flow and water quality in it by effect of various natural, socio-economic, and technical factors associated with the box.
- Both rate of flow and water quality do not change during the connecting processes with other boxes.

Such a way of formulation is similar to the modeling schema applied to *storage function method*[Kimur62] or *tank model method*[Sugaw72], that are conventionally used for modeling water networks in hydrology and river engineering. The model of urban water network can be therefore described by using these way of formulation. By substituting these boxes to components and by considering connecting processes between boxes as conduits, these types of models can be called component-conduit models. Namely, the concept of component-conduit model can be utilized in modeling urban water networks. In this paper, the following premises are prepared for modeling the urban water network.

- **Steady state assumption**

  It is assumed that the urban water network is modeled as a steady state flow system. The knowledge system deals with carcinogenicity risk associated with drinking water in the urban water network. The critical point of assessment about the carcinogenic risk is not a peak concentration of the carcinogen in drinking water, but a level of average concentration during long-term water use. The model is therefore needed to reflect water networks in yearly average or steady state conditions. The water network is modeled as a static system under this assumption. In other words, the model does not have explicitly the term of time. Accordingly, the model cannot represent dynamic changes in the network through time.

- **Definition of stuff, component, and conduit**

  Stuff: Stuff is transmitted through a network consisting of components and conduits. Stuff at a point in the network is characterized by rate of flow and other attributes. Under steady state assumption, the model does not consider dynamic change of flow potential in a point. Therefore, the model does not need variables representing flow potential. Stuff at a point is represented as a vector of attributes f expressed below:

$$f = (q, c_1, c_2, \ldots, c_n)$$

where q denotes rate of flow and $c_i (i = 1, \ldots, n)$ denotes value of the other attributes. In the RARCOM, stuff represents water flow. Namely, q represents rate of water flow, and c_i does a water quality index or a contaminant load involved in water flow.

Component: Component is a primitive to construct the model. A component, which has two terminals, upward one and downward one, runs stuff through itself from upward terminal to downward, and simultaneously varies attributes of the stuff. Behavior of component is represented by the following equation. Namely, when f_{in} denotes stuff running into the upward terminal and f_{out} denotes the stuff running off through the downward terminal, the following function $G(\cdot)$ represents behavior of the component.

$$f_{\text{out}} = G(f_{\text{in}}, p)$$

where p is vector of parameters related to behavior of the component. Types of the component determines the function $G(\cdot)$, while parameter p is determined in each component.

Conduit: Conduit is another model primitive. A conduit, which also has upward terminal and downward terminal, only transmits output stuff from a component to another component. Points where conduits are gathering at their upward terminal are dividing points of flow, simultaneously, points where conduits are gathering at their downward terminal are confluent points of flow. Behavior of stuff in conduits and dividing/confluent points follows the conservation of flow principle. Namely, stuff attributes are not varied while flowing through conduits. Total amount of each attribute is equal at upward and downward side of dividing/confluent points.

System represented as a network that is composed by these components and conduits shows a device topology. Components in a device topology are connected with zero or more upward conduits and zero or more downward conduits. In upper reach of the device topology, there are several components connected with no upper conduit. These components compose the upper boundary of the device topology. Output stuffs from these components mean the boundary condition for the model. Stuffs determining the boundary condition are set up as given or as determined from parameters of these components before executing simulations.

3.2 Dealing with Interdisciplinary Nature

To model an object in the world, modelers have to extract key principles from the world and to represent structure of the object in a form appropriate for solving problems. A number of researches pointed out the importance of these activities. For example, Hori and Ohsuga[Hori90] defined these requirements for modeling as *articulation problem*, Haggith[Haggi90] and Robertson et al.[Robert91] called these activities *idealisation*. Checkland[Check81] propounded *soft systems methodology*, which involves these modeling activities dealing with complex and ill-structured problems. Moreover, several researches in artificial intelligence discussed *ontology*, which is the conceptual definition and cataloguing of entities and relationships existing in the world, underlying in models in knowledge bases[Liu90, Colli87, Ramon92, Motod93].

According to Haggith, idealisation activity depends on purpose and point of view of the modeler. In particular, *weltanshauung* of the modeler, how the modeler recognizes the world, is the most important for modeling[Check81].

Following the above discussion, consider interdisciplinary nature of the THM problem. As mentioned in the previous chapter, multiple domain knowledge is required to deal with the problem of THM generation and its health effects. Each domain knowledge reflects thinking and underlying weltanshauung of the expert engaging in each domain. Thus, multiple weltanshauungen must be reflected in construction of the model dealing with multiple domain knowledge. Such a coexistence of weltanshauungen however, causes several issues. This is concerned with two of these significant issues, multiple ontology problem and inconsistencies between concepts in domains.

3.2.1 Multiple ontology problem

One of the issues is multiple ontology problem[Maeda91]. Conventional modelers are used to assume that ontology of the model is given or self-evident. For instance, in modeling knowledge systems to diagnose failure of electric circuits, modelers could assume implicitly that components of the model represent electric devices in the circuits. However, as discussed in the previous chapter, the model of urban water network must reflect multiple weltanshauungen to deal with the risk issues of THM problem, that is, multiple ontologies are required to construct such a model.

In recent years, several researches, for example Liu and Farley[Liu90], which developed knowledge systems for solving complex problems, pointed out that dealing with multiple ontologies is a requirement to solve these

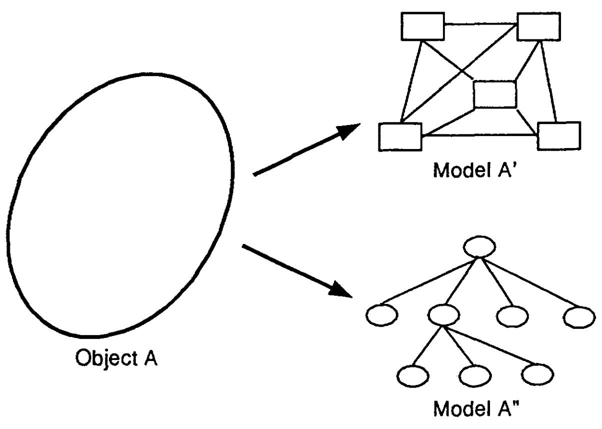


Figure 3.4: Deriving different models from one object.

problems. In the background of these claims, there is an understanding that to decide a unique model description, modelers have to specify primarily a level of abstraction of the object and their point of view. Because ontology is not always self-evident and sometimes subjective, there may be more than one modeling schemata for the object to be modeled.

It is considered that there are two approaches to model an object in the world on the basis of multiple ontologies. The first approach, derives different models from an object by using multiple ontologies(Figure 3.4). The second, makes a model consisting of several submodels, which represent parts of the object and are described on ontologies appropriate to represent the parts(Figure 3.5). Detailed discussions about these approaches are shown below.

a) Deriving different models from one object

Some researches proposed methods that use multiple ontologies in terms of reflecting distinct levels of abstraction to give rich representation of the world for adding detailed explanation capabilities on knowledge systems(Figure 3.4). For example, Collins and Forbus[Colli87], employed both *contained-stuff ontology* and *molecular-collection ontology* in qualitative reasoning system to describe behavior of liquids. Liu and Farley[Liu90], used two models that are based on *device ontology* and *charge-carrier ontology* to explain behavior of electric circuits. These researches use two different ontologies that are based on two different *class-wide assumptions*[deKle83], namely macroscopic and microscopic ones.

However, multiple ontologies are also used within the same level of the class-wide assumption. For example, an intelligent computer-aided design(CAD) system developed by Kiriyama, Tomiyama, and Yoshikawa[Kiriy91] holds multiple models based on different ontologies reflecting different points of view. The CAD system controls and integrates models reflecting various viewpoints, such as those considering physical constraints, engineering constraints, and geometrical constraints, by using their meta-model, i.e. a formal representation of common features of these models.

Model description of the urban water network in the system also requires use of multiple points of view to deal with an object in the real world. For example, consider land in the urban area. A viewpoint of municipal government will divide land area into such administrative units as cities. Therefore, modeler standing on this viewpoint will make a model of land as a set of cities. On the other hand, a viewpoint of hydrology or river engineering will divide land area by using the concept of basin. This viewpoint makes the model as a set of land units representing basins. Accordingly, the land area as a

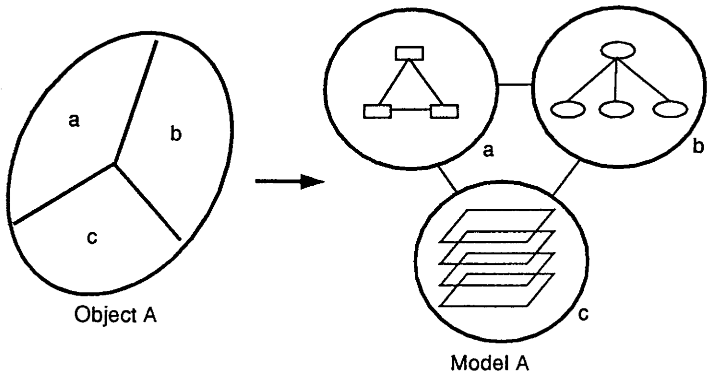


Figure 3.5: Model consisting of submodels based on different ontologies.

unique physical existence is modeled as two different types of models, 'cities' and 'basins', by using different ontologies, namely area in which people live and area which water runs through. To develop the system dealing with the THM problem, both concepts are needed for modeling urban water networks. The RARCOM must have such models based on different viewpoints.

b) Making a model that consists of submodels based on different ontologies

Another approach to model the object on the basis of multiple ontologies, is that it makes a model consisting of submodels that are described on different ontologies that are the most appropriate to the specific parts of the object(Figure 3.5).

For example, models of machines or electric circuits that are used for failure diagnosis, could be based on a single ontology. However, to deal with problems that occur in interaction between multiple domains, such as; environmental risk problems, ecological problems such as eutrophication in closed water environments, global environmental problems such as the greenhouse effect, and problems in international trading, it is impossible to describe the problems on the basis of a particular modeling schema.

Considering the THM problem, as mentioned in section 2.1, components of the urban water network correspond respectively to different viewpoints of different disciplines. For example, components representing river channels, filtration plants, and cities should be described by ontologies of hydrology, waterworks engineering, and administration respectively.

Modeling of the urban water network requires therefore a modeling framework in which firstly, each component is described by using the most adequate ontology corresponding to each domain viewpoint, and secondly, these components are aggregated into one model. The model in the RARCOM have to be constructed by using the approach.

3.2.2 Inconsistencies between concepts in domains

Another significant issue related to interdisciplinary problems such as the THM risk, is differences in terminologies and inconsistencies between multiple domain knowledge associated with the problems[Maeda93]. As shown in the previous section, it is required that parts of the model are based on different viewpoints appropriate to represent respective parts, and behavior of the parts of the model are represented by using domain knowledge corresponding to the respective viewpoints. Differences between concepts involved in multiple domain knowledge are, however, problems in such a modeling procedure. A term used in a domain may have different semantics in other domains. A statement that is true in a domain is possible to be negative in another domain.

Consider, for example, water qualities in urban water networks. As an index of organic contamination in drinking water, waterworks use potassium permanganate consumption(chemical oxygen demand measured by using potassium permanganate: COD-Mn), while biochemical oxygen demand (BOD) is used to measure organic contamination in rivers[1]. Although they are named the same water quality, they are actually different indices measured by different methods. Thus, we need a method to interpret these two indices in order to discuss the relationship of water quality between filtration plants and rivers. Fortunately, since there is a correlation between these two indices under some condition, we can use the correlation for its interpretation. Environmental simulations have so far been on models in which water quality is unified in a single index by using such relationships. However, knowledge about such interpretations has been considered as a kind of know-how of modelers or the domain experts, therefore, the knowledge has been implicitly embedded in models and has not been available for model users. This is one of the reasons why it is said that conventional models have low reusability or simulation programs are difficult to be manipulated by ordinary people.

Consequently, if a modeler intends to construct a model as a set of submodels that are described by using multiple domain knowledge, the modeler has to bridge gaps between domains or to solve inconsistencies between concepts. Moreover, it is desired that knowledge used to such bridgings is explicitly represented to make users understand the knowledge. To do this, I have developed *composite object model*, which is a modeling framework to deal with multiple domain knowledge. The framework is fundamentally de-

[1]This is valid only in Japan. In other countries, pottasium bichromate consumption(COD-Cr) or total organic carbon(TOC) may be used to measure organic contamination.

veloped to model urban water networks in knowledge bases by using multiple domain knowledge required to describe behavior of the networks. This paper presents detailed discussion about the framework in the next chapter, and shows an implementation of the framework in Chapter 5.

3.2.3 Premises to model on the basis of multiple points of view

As shown above, modelers have to cope with

- multiple ontology problem, and
- inconsistencies between concepts in domains,

to model urban water networks. In particular, modeling based on multiple ontologies consists of two approaches,

a) deriving different models from one object, and

b) making a model that consists of submodels based on different ontologies,

both which are needed to construct models of urban water networks. However, modeling in the approaches a) and b) may cause inconsistencies in models. To eliminate the inconsistencies in models, the following four constraints are introduced. These constraints are premises to keep consistency of models based on the two approaches. These are also premises to construct models on the composite object model framework, by which models can cope with inconsistencies between concepts in domains. This section shows the four premises[Maeda91].

- **Component-conduit model as the modeling schema**
  The first premise is that the model is described by using schema of

component-conduit model. Namely, all part of the model based on different ontologies have to be described in component-conduit models. This is a strict premise. For example, if the object to be modeled is an air pollution problem in urban areas, component-conduit model is not adequate to model such a problem. In this paper, the RARCOM deals with the THM problem. This enables one to have the premise because the THM risk is mainly propagated through the structure of urban system and, as mentioned in section 3.1, the urban system can be modeled as component-conduit model.

- **Common stuff**
  The second is that all components based on different viewpoints process the same stuff. In this paper the common stuff represents flow of water. The premise constrains that information transmissions between components are uniform independent from viewpoints of source and destination components.

- **Structural compatibility** The third is that stuff transmission between two components must be done in the point where terminals of the components correspond to a common spatial point in the real world. If any two components have such a compatibility in structure of the real world, these components can transmit stuff independently of underlying viewpoints. For example, as shown in Figure 3.6, it is needed that the discharge from the sewage treatment plant has to be confluent at a discontinuity point of segmented models of the river channel(as the upper of right side of the figure), while it is not permitted that discharge point from the plant does not correspond to discontinuity points between the river segments(as the lower of right side of the figure). The premise corresponds to *structural compatibility principle* defined in Liu

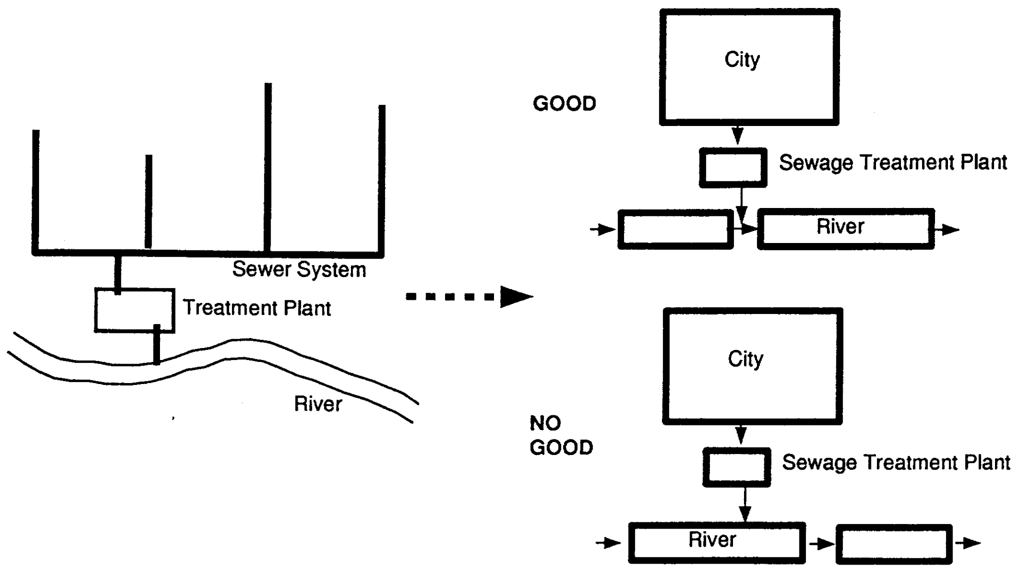


Figure 3.6: Structural compatibility.

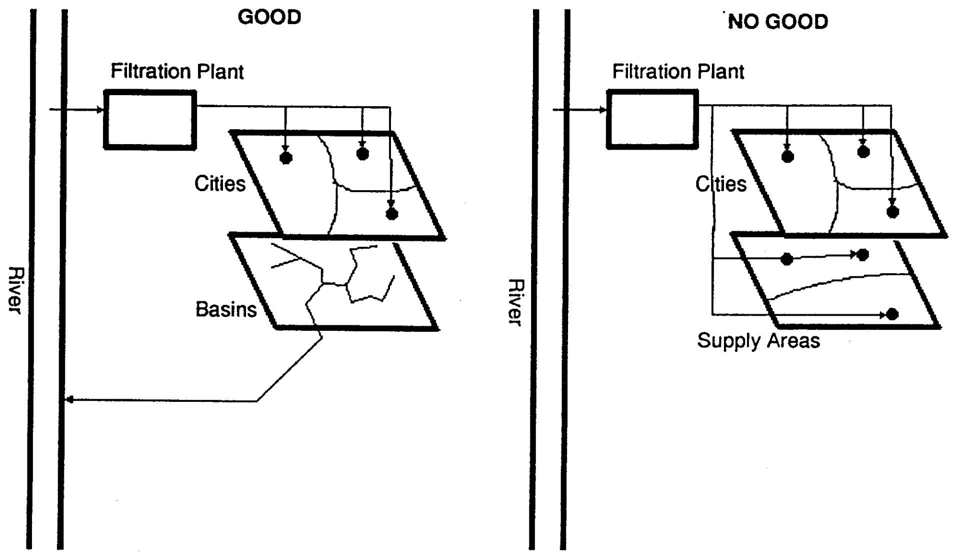


Figure 3.7: No collision in modeling of flows.

and Farley[Liu90].

- **No collision in modeling of flow in the world** The final premise is that a flow in the world has not to be dualized in the model. To model on the basis of multiple ontologies, a physical existence in the world is possible to be modeled in more than one component. However, if a flow in the real world is modeled as more than one component, these components collide with each other in some position in the whole flow system. This breaks consistency of the model. It is the reason that this premise is introduced for the modeling based on multiple ontologies. For example, Figure 3.7 shows two types of dual modeling of land in an urban area. The left side has no collisions in the model of water flow because two types of components, cities and basins, have different behavior about water flow. One is to supply the drinking water to households in cities, while the other is to catch rainfall and the discharged water from the households. On the other hand, in the right side of the figure, cities and supply areas, which are models of land based on the ontology of waterworks engineering, collide in modeling of water flow from a filtration plant to households. because the both have same function of water supply from the filtration plant. The latter modeling violates the premise.

Based on the four premises, The system models the urban water network in the knowledge base by using the composite object model framework. Detailed explanation of the composite object model is shown in the next chapter.

3.3 Programming Methods to Implement the Object Model

This section shows programming methods to represent knowledge associated with the risk problem. The methods have to have explicit and comprehensive knowledge representation, as mentioned in the previous chapter. To satisfy the requirement, the RARCOM uses two declarative programming methods, *object-oriented programming* and *logic programming*[Maeda93].

3.3.1 Model representation in object-oriented programming

While conventional programming methods intend to describe how it works, object-oriented programming intends to describe what it is. In object-oriented programming, things and concepts are represented as *objects*, which are basic units of program modules consisting of data representing those attributes, and procedures representing behavior and nature of those[Rumba91]. Namely, the programming method represents things and concepts by combination of those attributes and behavior or nature. Such a representation is closely related to the *frame* theory[Minsk75], which is proposed in the field of cognitive science as a framework to represent human knowledge.

In the programming method, data concerning attributes of *objects* are called *slots*, while procedures representing behavior and nature of those are called *methods*. *Classes* of *objects* define what slots and methods the *objects* have. A class is a description that defines the nature of *objects* in it.

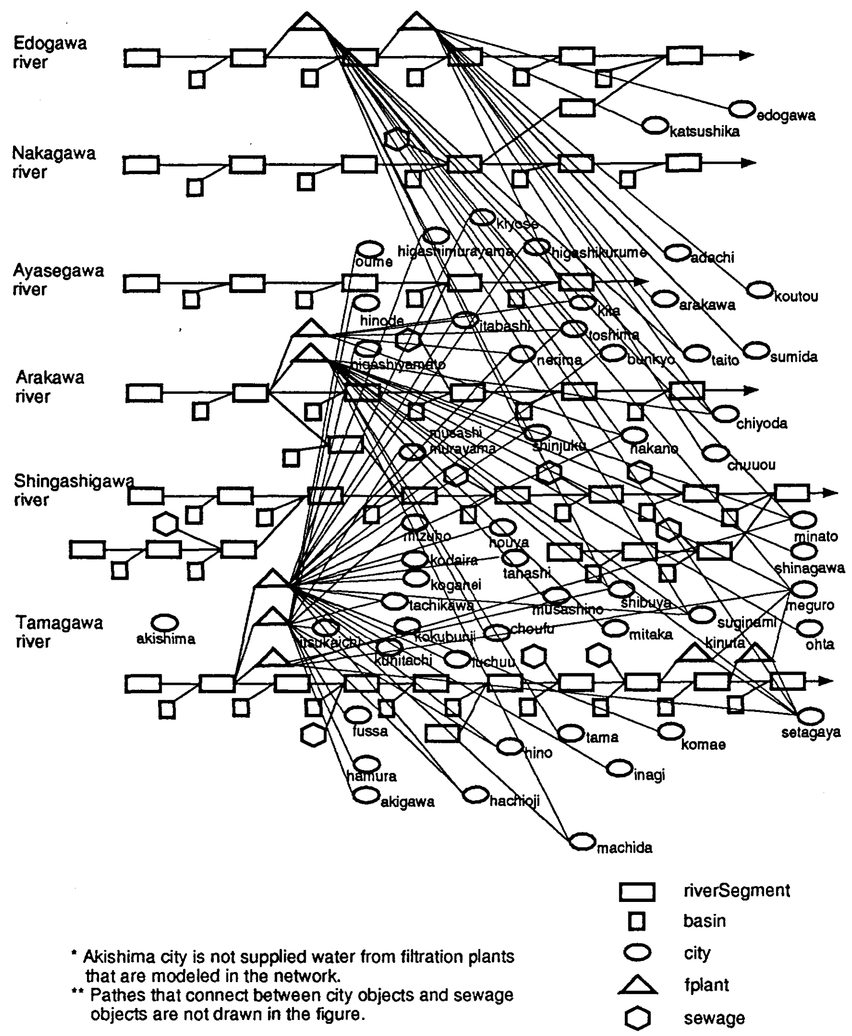
For example, consider an *object* representing a filtration plant. The model of water network in Tokyo has a number of *objects* corresponding to the actual

Table 3.1: A class definition of "filtration plant".

Class	Filtration Plant
Slots	Upward terminal (Intake point) Downward terminal (Destination of water-supply) Filtration method ...
Methods	- The object does 'filtration' and 'disinfection'. - While 'disinfection', THMs are generated. ...
Super-class	Waterworks Engineering
Instances	Higashi-Murayama Filtration Plant Sakai Filtration Plant Kanamachi Filtration Plant ...

filtration plants, such as Higashi-Murayama Filtration Plant, Sakai Filtration Plant, and Kanamachi Filtration Plant. Those *objects*, which are called *instances*, have the same slots and methods. The slots and methods of the instance *objects* of filtration plants are defined by 'class of filtration plant'. Table 3.1 shows part of class definition for filtration plants. According to the table, a filtration plant *object* has slots of 'upward terminal', 'downward terminal', and 'filtration method', and has methods of 'filtration', 'disinfection', and 'THM generation'.

If a class of *object* is more abstract than another one, *objects* in the latter class *inherit* nature of the former class. It means that the latter class has automatically slots and methods defined in the former class. In such *inheritance* relation, the former is called *super-class*, the latter is called *sub-class*. In the example in Table 3.1, a super-class of 'filtration plant' is the class of 'waterworks engineering'. The class of 'filtration plant' inherits the 'waterworks engineering' class, which defines the nature of waterworks engineering as an *object* (Detailed explanation of it is given in the next chapter).



* Akishima city is not supplied water from filtration plants that are modeled in the network.
** Pathes that connect between city objects and sewage objects are not drawn in the figure.

Figure 3.8: Model of urban water network in Tokyo.

One of the most remarkable characteristics of object-oriented programming is in its execution mechanism. Execution of *object* programs always requires any messages prompting the execution. In other words, *objects* cannot work without receiving any messages. This mechanism is called *message passing.* All valid messages sent to an *object* correspond respective to particular methods. When an *object* receives a message, it invokes the corresponding method. The execution of method may invoke another message passing to another *object.* Such a chain of message passing produces whole behavior of a society of *objects.*

This paper presents five *object* classes to model components in urban water networks; `riverSegment`, `city`, `fplant`, `sewage`, and `basin`, which correspond respectively to segments of river channels, cities, filtration plants, sewage treatment plants, and river basins.

These are *objects* representing components of the model of the urban water network shown in Section 3.1. Namely, these *objects* have a method that corresponds to the function $G(\cdot)$ representing behavior of the components, and slots that correspond to parameters of the components. Implementation of these *objects* are shown in Chapter 5.

The model of urban water network is constructed as a network consisting of *object* instances of these classes. Figure 3.8 shows the Tokyo water network model represented as a network of *objects* connected along the six major rivers. The network consists of over 170 instances of component *objects* and conduits connecting between `sewage` and `riverSegment`, `basin` and `riverSegment`, `riverSegment` and `fplant`, `fplant` and `city`, and `city` and `sewage`, however conduits connecting between `city` and `sewage` are not drawn in the figure to reduce confusion in the figure.

3.3.2 Description of relation and behavior in logic programming

Logic programming is based on the idea that logic can be used as a programming language[Lloyd87]. In this programming method, program is described as a set of logical formula, namely as an axiom set, and processing system for the program works as a theorem prover for the axiom set.

In the system, Prolog-like logic programming language is used for program description. Namely, the logic programming is based on first-order predicate logic, and each axiom is represented by Horn clause. In such a language, a predicate is represented by combination of a *functor*, i.e. name of the predicate, and arguments. For example, a statement, "Fuchuu city contacts with Tamagawa river." can be represented as:

```
contact('Fuchuu', 'Tamagawa').
```

Further, a logical formula, such as "If A and B then C.", can be represented as:

```
C :- A, B.
```

This axiom means that if both predicates A and B are true then the predicate C is true. In other words, the predicate C can be validated by checking validity of the predicates A and B. This axiom can be considered as a definition of the predicate C by using the predicate A and B.

Such a programming method has two advantages for the modeling.

1. Declarative description of relation:
   Structural description in models involve a number of representations like "A has relation C with B." The forms of predicate logic is appropriate to describe such relation.

2. Rule description:

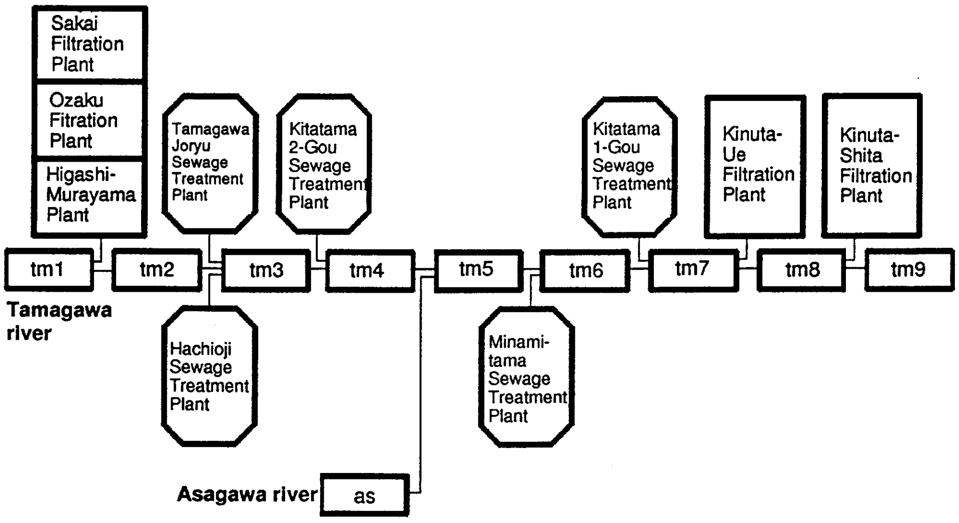
   Knowledge about system models involves a number of IF/THEN style rules. The form of predicate logic is also adequate to represent such rules.

As shown above, a program in logic programming consists of declaration of facts and definition of rules. Regarding the declaration as a database and the rule definition as a searching condition for the database, information system of managing environmental resources or ecosystems can be adequately constructed on the basis of logic programming[Robert91].

In this paper, logic programming is used for describing:

- relation associated with *objects* in the model; such as what *objects* a class involves, and what kind of relation there is between components,
- interactions between *objects*,
- methods of *objects*, which representing nature and behavior of them, and
- meta-information, such as definitions of predicates and terminology associated with symbols used in the axioms.

Figure 3.9 shows an example of representing the relation between *objects* along Tamagawa river. In the figure, for example, the predicate `modelComponent(tm1,riverSegment)` shows that the *object* `tm1` represents an instance of the class `riverSegment`, and the predicate `connect(tm1,sakai_f)` shows that the *object* `tm1` connects the *object* `sakai_f`, which is an instance of the class `fplant`. Detailed explanations of implementation of the axioms are shown in Chapter 4 and Chapter 5.


Sakai Filtration Plant
Ozaku Fitration Plant
Higashi-Murayama Plant
Tamagawa Joryu Sewage Treatment Plant
Kitatama 2-Gou Sewage Treatment Plant
Kitatama 1-Gou Sewage Treatment Plant
Kinuta-Ue Filtration Plant
Kinuta-Shita Filtration Plant
tm1
tm2
tm3
tm4
tm5
tm6
tm7
tm8
tm9
Tamagawa river
Hachioji Sewage Treatment Plant
Minami-tama Sewage Treatment Plant
Asagawa river
as

```
modelComponent(tm1,riverSegment);
modelComponent(tm2,riverSegment),
modelComponent(sakai_f,fplant);
modelComponent(minamitama_s,sewage);
          ...
connect(tm1,tm2);
connect(tm1,sakai_f);
connect(minamitama_s,tm6);
          ...
```

Figure 3.9: Model representation of Tamagawa river system.

3.4 Knowledge-Based Simulation

The RARCOM must have computing environment to simulate exposure situations of the risk for supporting the EA step in risk assessment. To do this, the system employs *knowledge-based simulation*[Fox90] on the basis of the object model that is discussed above.

Knowledge-based simulation, which is a knowledge system technique to support and to enhance various simulation procedures, involves:

- qualitative or quantitative simulations on models constructed in knowledge bases, which are considered as one of model-based reasoning[Hopgo93], and
- to support processes in simulation life cycle, such as model identification and construction, execution of simulation, and evaluation of consequences.

The system uses the former technique, especially in executing quantitative simulation on the object model.

Knowledge-based simulation has the following advantages.

- In knowledge-based simulation, the simulation system can involve in the knowledge bases not only the model itself but also background knowledge and assumptions that are used for construction of the model.
- The system can not only simulate, but also explain processes in the simulation.
- Knowledge related to simulation processes is described in knowledge bases by using declarative notation, which can make the descriptions comprehensive.

- The system can describe a nature of non-deterministic behavior that is changed according to the external conditions.

In conventional simulations in environmental science, the models have so far been programmed on the basis of mathematical forms by using procedural languages. Such a modeling method, however, has two problems. First, since system users often want to know only the consequences or the estimated results of simulations, they misunderstand the consequences because of lack of background information such as limitation of the models, assumptions underlying the models, and knowledge required to understand the consequences. Second, if the users want to know the background information, they cannot extract those directly from the models. Because the information, which is implicitly scattered and embedded in the models, is not described in forms that users can access. In contrast with conventional systems, knowledge-based simulation systems can explicitly equip and add such information in the simulation models. Knowledge-based simulation systems can utilize such information and data to help user's understanding.

As shown in Section 3.1, the model of the urban water network is represented as a set of declarations that represent static quantitative constraints between stuffs at all points in the network. In other words, the model representation is equivalent to a set of simultaneous equations that represent the static relation among the components of the system. The simulation on the model means, therefore, solving a set of simultaneous equations with additional equations that represent the boundary condition for the simulation. Consequently, the simulation mechanism in the RARCOM is implemented as a constraint solver for such a set of simultaneous equations.

The simulation mechanism determines value of the stuff f at a component in the model by solving constraints existing in a component-conduit network

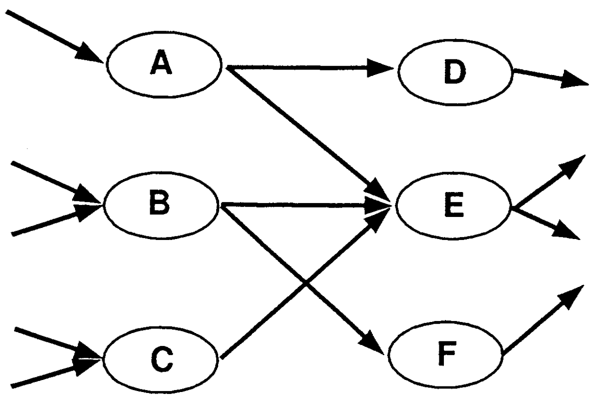


Figure 3.10: An example for algorithm `simCompo`.

in upper reach of the component. For example, in Figure 3.10 the value of stuff at the component E is determined by solving constraints in network involving components A, B, C, D, E and F, and components in upper reach of components A, B, and C.

The algorithm named `simCompo` that is for determining value of stuff at downward terminal of a component is implemented in the system. Algorithm `simCompo` is implemented as a method that is common to all components in the model. If anyone sends to a component a message to invoke the method `simCompo`, the component will reply the value of stuff at the component that is obtained by executing the following algorithm:

Algorithm 3.1: simCompo Let D_i denote a component in the model, $U_j (j = 0, \ldots, n)$ denote component directly connecting the upward terminal of the component D_i, $D_k (k = 1, \ldots, m)$ denote component directly connecting the downward terminal of the component U_j, and

$i \in \{1, \ldots, m\}$. Stuff at the downward terminal of the component D_i are obtained by the following algorithm.

1. If there are any components in upward of the component, namely $n > 1$, go to the next step. Otherwise, since it means that the stuff at the component D_i is a part of the boundary condition for the model, which has been set prior to the simulation, evaluate the set value and terminate the algorithm.
2. Send messages that request to execute `simCompo` to all component U_j.
3. Solve the constraint between outputs from all U_j and inputs to all D_k. The constraint can be represented in a set of simultaneous linear equations. Accordingly, the constraint can be solved by applying solving algorithm of the equations(**Algorithm 3.2**).
4. The previous step gives the input, namely the stuff at the upward terminal, to the component D_i. Let I denote the input. Obtain the value of stuff O at the downward terminal of the component D_i from the following function,

$$O = G(p, I)$$

   where $G(\cdot)$, which is defined in Section 3.1, represents behavior of component D_i, and p denotes the parameter of the component.

For instance, consider obtaining the value of stuff at the downward terminal of the component E in Figure 3.10. Let I_i denote the value of stuff at the upward terminal of the component i, and O_i denote the value of stuff at the downward terminal of the component i, $i = \mathrm{A, B, C, D, E, F}$. At first the procedure passes by the step 1 because there are three components, A, B, and C, in upper reach of the component E. The step 2 send a message "execute simCompo." to each components A, B, and C, respectively, then obtain the value of stuffs O_A, O_B, and O_C. In the step 3, by solving a set of quantitative constraints with respect to stuffs, O_A, O_B, O_C, I_D, I_E, and I_F, the value of I_D, I_E, and I_F are obtained. Finally, O_E, the value of stuff at the downward terminal of the component E, is obtained by applying behavior of the component to the stuff I_E. Namely, the stuff O_E is obtained from the function,

$$O_E = G_E(p_E, I_E)$$

where $G_E(\cdot)$ represents behavior of component E, and p_E denotes parameter of component E.

The solving algorithm for a set of simultaneous linear equations, which is employed by the RARCOM, is shown below.

Algorithm 3.2: solving algorithm for simultaneous linear equations

The following algorithm solves a set of simultaneous linear equations involving $n (n > 1)$ variables, which consist of n equations.

1. Let $i = 1$.
2. Find a variable which exists in the most left in the i th equation. Let x_i denote the variable.
3. Transform the i th equation to form as shown below;

$$x_i = RHS_i$$

   where, RHS_i is a form not involving variables $x_1, \ldots, xi$.

4. If $i = n$, then go to step 7, or else go to the next step.
5. Substitute RHS_i for x_i in all j th equations, where $1 \leq j \leq n, j \neq i$.

6. Let $i \leftarrow i + 1$ and go to step 2.

7. Evaluate all $RHS_j, j = 1, \ldots, n$.

It is said that such a constraint solving algorithm, which determines its behavior dynamically during the execution according to the structure of the model, is generally less efficient in computing performance than algorithms in conventional programs, which work by using procedures that are determined on the basis of the structure of models prior to the execution. Although this may be a disadvantage of the algorithm, the RARCOM employs it because the system needs a universal algorithm that can cope with structural changes of the model, for instance adding new filtration plant *objects* to the network model, during the simulation processes of the system.

3.5 Dealing with Uncertainties

As mentioned in Section 2.2, the system of supporting risk assessment has to equip a function to deal with uncertainties associated with the risk problem. Akiyama[Akiya91] pointed out that fluctuations of data along propagation of risks is important to estimate risks in urban water networks. Following it, the RARCOM deals with the fluctuations of data as an aspect of uncertainties associated with the risk problem.

For example, in the network shown in Figure 3.11, attributes of the stuff at the component E are directly affected by fluctuations of attributes of the stuff at components A, B, and C. If the network represents a model of an urban water network, it means that water quality at the point E in the urban area is affected by water quality fluctuations in its upper reach.

Furthermore, the fluctuation at the component E is affected by not only those at the upward components A, B, and C, but also by that at the com-

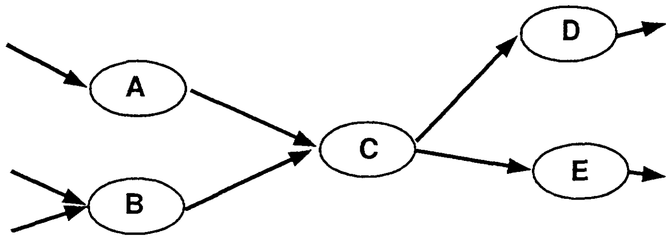


Figure 3.11: A network propagating fluctuations of data.

ponent D. Consider an attribute of stuff that represent a contaminant load at components C, D, and E. Let c_i be an attribute at component i, where $i = \mathrm{C}, \mathrm{D}, \mathrm{E}$. There is a relationship,

$$c_{\mathrm{E}} = c_{\mathrm{C}} - c_{\mathrm{D}}.$$

In general, If variable Y is a function of random variables $X_1, X_2, \ldots, X_n$ as follows,

$$Y = f(X_1, X_2, \ldots, X_n),$$

and if the function $f(\cdot)$ is continuous and partially differentiable, then variance of the variable Y, σ_Y^2, can be estimated by the following form:

$$\begin{aligned}\sigma_Y^2 \approx\ & \frac{\partial f}{\partial X_1}^2 \sigma_1^2 + \frac{\partial f}{\partial X_2}^2 \sigma_2^2 + \cdots + \frac{\partial f}{\partial X_n}^2 \sigma_n^2 \\ & + 2\frac{\partial f}{\partial X_1}\frac{\partial f}{\partial X_2}\sigma_{12} + 2\frac{\partial f}{\partial X_1}\frac{\partial f}{\partial X_3}\sigma_{13} + \cdots + 2\frac{\partial f}{\partial X_n}\frac{\partial f}{\partial X_{n-1}}\sigma_{n(n-1)}\end{aligned}$$

where $\sigma_1^2, \ldots, \sigma_n^2$ denote variances of the variables $X_1, \ldots, X_n$ and σ_{ij} denotes the covariance between variables X_i and $X_j (i, j = 1, \ldots, n)$ [Mood74]. This is called the form of propagation of errors, which is usually used for treating errors in physical and chemical experiments and surveyings.

According to the form, if σ_k^2 is variance, which is regarded as an index of fluctuation of the variable, of the attribute at the component k, and σ_{kl}

is covariance between the attributes at components k and l, $(k, l = \mathrm{C}, \mathrm{D}, \mathrm{E})$, then the variance σ_{E}^2 can be estimated by the following form:

$$\sigma_{\mathrm{E}}^2 = \sigma_{\mathrm{C}}^2 - \sigma_{\mathrm{D}}^2 - 2\sigma_{\mathrm{CD}}.$$

This form shows that variance of the variable c_{E} is affected by variance of the variable f_{D}. In water networks in the real world, it suggests that increase of water use in upper reach of a river may possibly affect uncertainty of water quality in lower reach of the river. In this sense, it is important to estimate fluctuations of data associated with the risk.

There are two methods to estimate fluctuations of data, methods of using 'variances' and *Monte Carlo simulation.*

The former is a method that estimate fluctuations of data by using variance. As shown above, variance can be regarded as index of fluctuations. Estimate of variance therefore, corresponds to estimates of range of fluctuations. The method has an advantage, when there is an unknown variance, the variance can be estimated by the form of propagation of errors, given other variances that are required to estimate the target variance. However, there are two disadvantages to it. The first, is that the method can estimate only range of fluctuations, but cannot estimate other aspects of fluctuations; such as probability distributions and maximum/minimum value in fluctuations. The second, is that the method needs too many numbers of data to estimate variance of all variables in the model. Practically, it is difficult to obtain sufficient data from urban water network in the real world to estimate variances of all attributes at all components in the model. This is one of the biggest obstacle to do the method.

The latter, Monte Carlo simulation[Rubin81], is a method that estimate fluctuations of variables from iterate of simulations of the variables under randomly generated conditions. The method has a disadvantage, it consumes a long computing time and a lot of computer resources. However, the method has an advantage, it can deal with not only range of fluctuations but also probability distribution associated with the fluctuations, which the former method cannot deal with.

Consequently design of the RARCOM employs the latter method to estimate fluctuations of attributes at components. Algorithm used for implementing the method is shown below:

Algorithm 3.3: Monte Carlo simulation on the RARCOM Fluctuation of a variable, say x, is estimated by the following procedure.

1. Estimate probablistic distributions of all variables related to the variable x in advance.
2. Generate value of those variables randomly according to the distributions.
3. Do simulation by algorithm `simCompo` on the basis of variables generated by the step 2. Consequence of the simulation is stored in the system.
4. Iterate the steps 2 and 3 as many times as necessary.
5. All consequences of iterated simulation represents distribution of the variable x.

The obtained distribution is considered to reflect fluctuation of the variable.

Chapter 4

Composite Object Model: a Modeling Framework to Deal with Multiple Domain Knowledge

As mentioned before, multiple "weltanshauungen" or multiple domain knowledge is required to understand environmental risk problems such as THM problem. However, possibly there may be several inconsistencies or gaps between concepts involved in this multiple domain knowledge. For example, as shown in Section 3.2.2, water quality indices of organic contamination used in rivers and filtration plants are different in spite that these are used to indicate the same water quality.

In a knowledge engineering point of view, it can be said that problem description of such an interdisciplinary problem is structured as a system of involving multiple axiom sets. Such a knowledge representation is called *multiple world model*[Chika84, Nakas83]. A system of multiple world model has the following characteristics.

- There are several axiom sets, which involve different axioms respectively, in the system. Logical consequence from each axiom set is thus possible to differ from each other.
- Scope of the inference mechanism is bounded in the range of each axiom set. When a predicate is called in an axiom set, the inference mechanism searches the definition of the predicate to validate it only in the axiom set. Even if there exists another definition of predicate of the same name in another axiom set, the inference mechanism does not access the other definition, as far as the program does not command explicitly to access the other definition.
- Whole behavior of the system is defined as interactions between axiom sets.

It is a contrast to the Prolog language, in which a program can be regarded as a single axiom set, and the inference mechanism for the language works as a searching procedure in the whole of the program.

In this paper, the knowledge system employs a programming method that is combined with object-oriented programming and logic programming on the basis of multiple world model. In this programming method, behavior of *objects* is described by using axioms of first order predicate logic in their class definitions. Namely, a class in the programming method represents an axiom set. Passing a message to an *object* corresponds to calling a predicate in the *object*; i.e. *methods* of *objects* are implemented as axioms(clauses) of predicate logic. Furthermore, *inheritance* in the programming method means that a sub-class holds implicitly all axioms defined in its super-class.

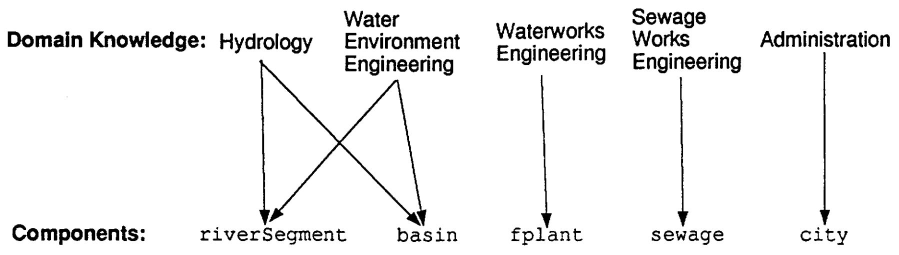


Figure 4.1: Relation between domain knowledge bases and model components.

On the basis of the idea, I developed the *composite object model*, which is a modeling framework to construct object models of the world[Maeda93]. The chapter shows contents of the composite object model.

The framework of composite object model makes models by using three parts of knowledge bases; domain knowledge bases, relation between components, and bridging rules over multiple domain knowledge.

4.1 Domain Knowledge Bases

The first one shows a set of knowledge bases in which disciplinary knowledge in respective domains are described. Each domain knowledge base is represented as an *object* in the knowledge system. In other words, knowledge in a domain is interpreted as a set of axioms of predicate logic, and the set is regarded as a class representing the domain knowledge, namely a *domain knowledge base.*

The system uses five domain knowledge bases shown in Figure 4.1 to deal with the THM problem in the urban water network. Components consisting of the model, such as `riverSegment` and `fplant`, are defined as sub-classes of the domain knowledge bases of 'hydrology', 'water environment engineering', and 'waterworks engineering', and they inherit axioms defined in respective domain knowledge bases.

In a class definition of a component of the model, axioms that determine behavior of the component are defined. In particular, there is an axiom that defines predicate `process(Obj,In,Out)`, which denotes that if stuff `In` is inputted into *object* `Obj`, the *object* outputs stuff `Out`. This predicate corresponds to the function $G(\cdot)$ explained in Section 3.1. Namely, the method corresponding to the function $G(\cdot)$ is implemented as the axiom defining the predicate `process(Obj,In,Out)`. For example, consider that an axiom,

```
process(Obj,In,Out) :- A,B,C;
```

is defined in a class of a component. In procedural sense, this form means "If the predicate `process(Obj,In,Out)` is called, the predicate calls a sequence of predicates `A`, `B` and `C` in this order." In other words, it can be said that the procedure `process(Obj,In,Out)` consists of sub-procedures `A`, `B`, and `C`. While this procedure `process(Obj,In,Out)` is for describing behavior of the component, procedures `A`, `B`, and `C` may be used for other components of models. Namely, procedures `A`, `B`, and `C` may be reusable primitives to describe behavior of components. The domain knowledge bases are considered as the collections of such primitives, that is, terminology and forms used in the domains.

It is important that the contents of each domain knowledge base are fundamentally independent of the specific arguments associated with the target object. In other words, domain knowledge bases can be described regardless of what modeling schema is used or what object in the world is modeled. Because these arguments are meaningful only in the modeling process that follows definitions of components, which are sub-classes of domain knowledge bases. Simultaneously they have little significance for construction of domain knowledge bases, which are super-classes of the components. Namely, domain knowledge bases in the composite object model are highly portable and reusable. They can be reused in models for other purposes or models based on other modeling schemata.

Figure 4.1 shows relation between domain knowledge bases and model components. For instance, a `riverSegment` component inherits domain knowledge bases of **hydrology** and **water environment engineering**, while a `fplant` component does domain knowledge base of **waterworks engineering**. Nature of each component is described on the basis of axioms inherited from domain knowledge bases. For example, in Table 3.1 behavior of `fplant` *object* is represented by using terms, such as 'filtration' or 'disinfection', which are defined in the super-class domain knowledge base of **waterworks engineering**. Behavior of components is represented by using terms and forms defined in domain knowledge bases that are inherited by the components. Appendix B.2 shows an example of a part of a domain knowledge base `waterWorks`, and Appendix B.5.1 shows a definition of the component `fplant`, which inherits the domain knowledge base.

It must be noted that both `riverSegment` and `basin` have multiple super-classes. These relationships are called *multiple inheritance*. In such relationships, it is needed to keep consistencies of axioms in the sub-classes, that axioms in the super-classes are not inconsistent with each other, or that some inconsistency solver is introduced. In this case, the former reason prevents the sub-classes having inconsistencies. Axiom sets in `riverSegment` and `basin` involve no inconsistencies, because the domain knowledge base of **hydrology**, one of the super-classes, involves only knowledge associated with rate of water flow, excluding knowledge associated with water quality, while the domain knowledge base of **water environment engineering**, the other super-class, have only knowledge associated with water quality.

Making such a set of modules of domain knowledge differs from making modules of subroutines in procedural programming languages. While the former consists of conceptual clustering of domain knowledge and specification of the domain knowledge bases to the model components, the latter makes a structure of modules of the procedures. In other words, while the former makes modules of "what", the latter makes ones of "how".

For example, let a procedural program be a model of behavior of a filtration plant, and consist of three modules; initializing one, main one, and exiting one. Making a program consisting of such modules is appropriate in structured programming in procedural languages. When a part of behavior of the model, for instance the process of filtration, is changed, however, what modules should be modified? This change of the filtration process model may require not only modification of the main module, but also that of the initializing one or the exiting one, because the change may affect the initializing process of data or the output of the results. In such procedural programming, premises for the program are implicitly scattered and embedded in the procedural program, thus, when the premises are changed, the change may affect the whole of the program. If the system becomes more complex, it becomes more difficult to specify modules to be modified. This is a kind of so-called *frame problem*[McCar69]. The composite object model copes with the frame problem at the following two points.

First, as shown in Section 3.2.3, the composite object model is based on the modeling schema of the component-conduit model. It means that each model of the component is needed to represent only the internal behavior of each component and the interface via conduits. It implies that changes in descriptions of behavior of a component will scarcely affect the other components in the model.

Secondly, In the composite object model, knowledge related to the modeling is modularized on the basis not of order of the procedures, but of disciplines of knowledge. For example, in the model of the urban water network, if the model of filtration process must be changed, modification of the model can be done by only modifying descriptions associated with the filtration in the domain knowledge base `waterWorks`, because both the axiom representing the filtration and the definitions of data used in the axiom are in the knowledge base. In other words, in the composite object model, some data and procedures to manipulate the data are stored in a few restricted modules; that is, they are not scattered in modules in the system. Consequently, the frame problem related to the modeling is inhibited as possible by the composite object model.

4.2 Relation between Components

The second part involves two kinds of information associated with relation between components of the model. These provided a topology of the component-conduit network and quantitative constraints between components in the network.

The *object* `simModel`, which is a knowledge base holding axioms representing the topology and the quantitative constraints of the network model, is introduced to store and manipulate the information. Axioms shown in Figure 3.9 are described in `simModel`.

Information held in `simModel` involves the following axioms.

1. definition of components

   A few predicates are used for definition of *object* instances representing model components. For example, predicate `modelComponent`, which is shown in Figure 3.9, is used to define membership of instances. Predicate `objectAttrs` defines value of slots in *objects*. For instance;

   ```
   objectAttrs(ed1, [{name,"Edogawa-1"},{length,3.0}]);
   ```

   defines that *object* `ed1` has slots, `name` and `length`, which have values, `"Edogawa-1"` and `3.0` respectively.

2. definition of device topology

   As shown in Figure 3.9, predicate `connect` is used for defining topology of the model.

3. definition of quantitative constraints

   Another few predicates are used for definition of quantitative constraints existing in relation between flows in the model. For example;

   ```
   fixedFlow( {kosuge_s,ar3}, 107.0 );
   ```

   represents that rate of flow from *object* `kosuge_s` to *object* `ar3` is constant 107.0 $10^3 m^3/day$, while axiom,

   ```
   flowRate( {nk3,nk4}, {nk3,sn1}, 1.0 );
   ```

   defines that rate of flow from *object* `nk3` to *object* `nk4` is equal to that from *object* `nk3` to *object* `sn1`.

The *object* `simModel` also involves methods for knowledge-based simulations, which invoke algorithms discussed in the previous chapter. These methods are implemented as a rule set corresponding to the algorithms. As mentioned in Section 3.4, the axiom set representing relation between components is equivalent to a set of simultaneous equations representing the model. Namely, the execution of simulation on the object model is done by combining of the axiom set representing both the topology and quantitative constraints of the model, and the rule sets for the knowledge-based simulation. Appendix B.3 shows examples of axioms and methods held in the *object* `simModel`.

4.3 Bridging Rules over Multiple Domain Knowledge

The third, is bridging rules over concepts in multiple domain knowledge. As mentioned in Section 3.2.2, there are a number of conceptual inconsistencies between multiple domain knowledge. Therefore, the knowledge system based on multiple domain knowledge has to deal with rules to eliminate such inconsistencies and bridge over gaps between concepts in domain knowledge. The *object* `bridge`, which is a knowledge base storing such bridging rules, is introduced to deal with such inconsistencies. Appendix B.4 shows examples of axioms involved in the *object* `bridge`.

The most important concepts that should be bridged over domains are the ones associated with stuff attributes. One of premises for the composite object model framework, which is shown in Section 3.2.3, is the constraint that all components treat the same stuff. It means that the stuff is the only information carrier that is transmitted between multiple domains. Accordingly, when stuff is transmitted between components, concepts associated with the stuff, e.g., water quality indices, have to be dealt with by the bridging capability.

In the RARCOM, two kinds of bridging rules shown below are prepared to deal with water quality in the urban water network.

1. **Interpretation rules** If there exists a rule that interprets a concept in one domain that also corresponds to another one in the different domain, it will be stored in the *object* `bridge`. For example, consider that a `riverSegment` *object* transmits stuff to an `fplant` *object*. Since, as shown in Section 3.2.2, organic contamination is indicated by BOD in rivers, while it is done by COD-Mn in filtration plants, it is required that BOD observed in the `riverSegment` *object* is interpreted as COD-Mn. Accordingly, when stuff is transmitted between two *objects* based on different domains, corresponding bridging rule is called and interpretation as shown above is executed.

2. **Default rules** If, when transmitting information between two domains, a concept in one domain has no correspondence with the other, default rules are called. In the system, the default rules deal with information transmission associated with water quality indices between concepts in the urban water network model. If there exists a water quality index that is defined in one domain inherited by the source component and is simultaneously not defined in the domain inherited by the destination, the default rule adds the index to the destination as a new attribute of stuff and set its value as equal to it in the source. In other words, the index value is not changed in the default bridging procedure.

For more complex problems than the problem currently dealt with by the RARCOM, more complex bridging rules will be installed. For instance, consider interpretation rules between BOD and COD-Mn in conditions such

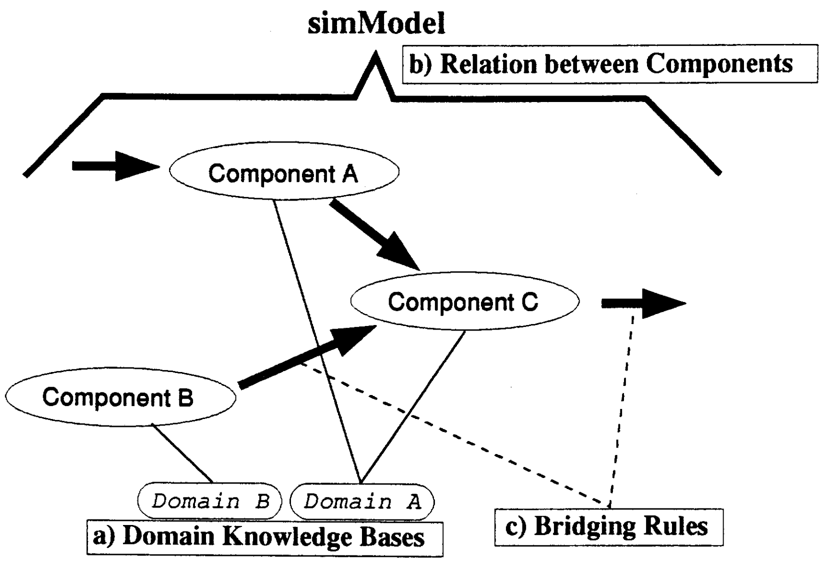


Figure 4.2: The composite object model.

as floods or droughts. The rules corresponding to the two conditions are different each other, because nature of organic compounds in water in the two conditions is significantly different. Accordingly, changes in flow condition require a variety of interpretation rules for the indices. Moreover, it must be noted that such requirements for various bridging rules are dependent on nature of problems and objects to be dealt with. Modelers will have to set more rules according to nature or complexity of the problems.

Consequently, framework of the composite object model is summarized as shown in Figure 4.2. Namely,

a) each component is documented by using knowledge of its corresponding domain,

b) relation between those components is stored and manipulated by the *object* `simModel`, and

c) gaps between concepts in components are bridged by the *object* `bridge` on the conduits connecting the components.

This modeling framework has advantages as listed below.

- Each domain knowledge can be represented in knowledge bases independently of a particular modeling schema or concerned problems.
- Each part of object in the real world is described as a model component by using domain knowledge that is appropriate to represent it.
- The knowledge system can infer across multiple domain knowledge.
- Users can examine the bridging rules, which have so far not been explicitly represented in models, as important elements constructing the model.

This type of problem solving system such as the RARCOM based on multiple domain knowledge has so far not been developed. Miki et al.[Miki93] proposed another framework shown in Figure 4.3. The system based on Miki et al.'s framework involves a community of agents, which are autonomous and mutually independent knowledge-based systems for various problem solving, and a terminology agent, which deals with all vocabularies associated with all agents and support inferences related to multiple agents. The community of agents and the terminology agent correspond respectively to domain knowledge bases and bridging rules in the composite object model. Their framework is considered a more general method for problem solving than the

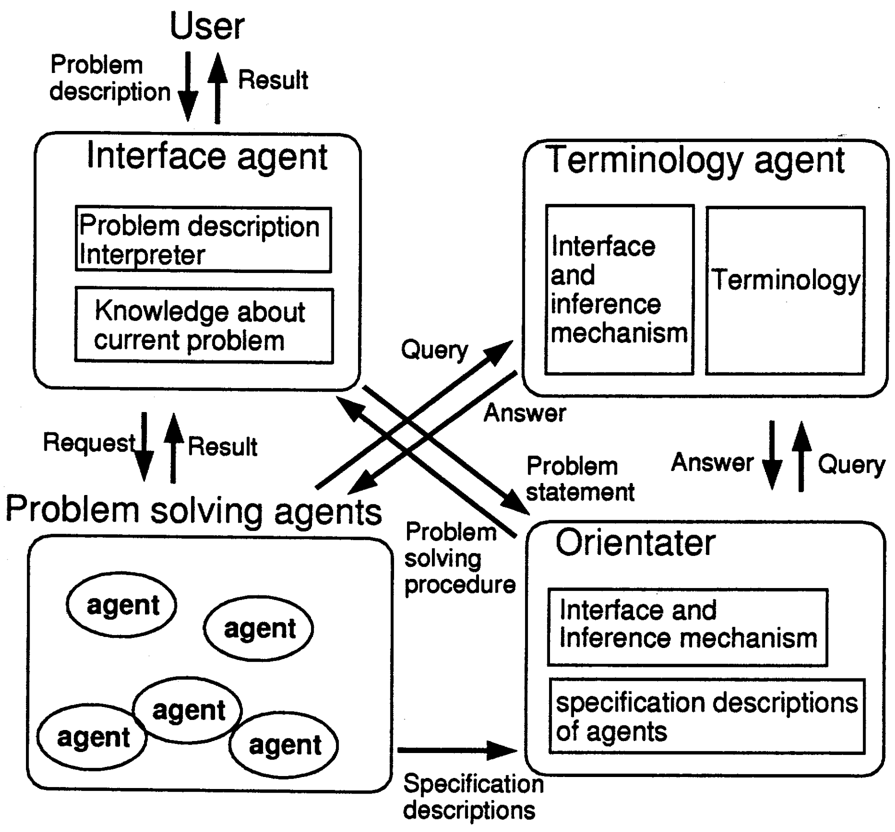

User
Problem description
Result
Interface agent
Problem description Interpreter
Knowledge about current problem
Terminology agent
Interface and inference mechanism
Terminology
Query
Answer
Request
Result
Problem statement
Answer
Query
Problem solving agents
agent
agent
agent
agent
agent
Problem solving procedure
Orientater
Interface and Inference mechanism
specification descriptions of agents
Specification descriptions

Figure 4.3: Miki et al.'s modeling framework[Miki93].

composite object model. Because the former framework has an orientater agent that can generate problem solving procedures from various problem statements, while the latter can deal with only a kind of constraint-solving problem on component-conduit models. However, Miki et al.'s framework has not yet been implemented as a working knowledge-based systems. It seems because of difficulty of the implementation of the orientater. The RARCOM is the first attempt to construct a problem solving system based on multiple domain knowledge.

Chapter 5

Implementation of RARCOM, a Risk Assessment and Risk CommunicationSupport System

5.1 Construction of the RARCOM system

The risk assessment and risk communication support system RARCOM, is constructed as a system having the following features.

1. Knowledge engineering features

    - The system is constructed on a workstation.

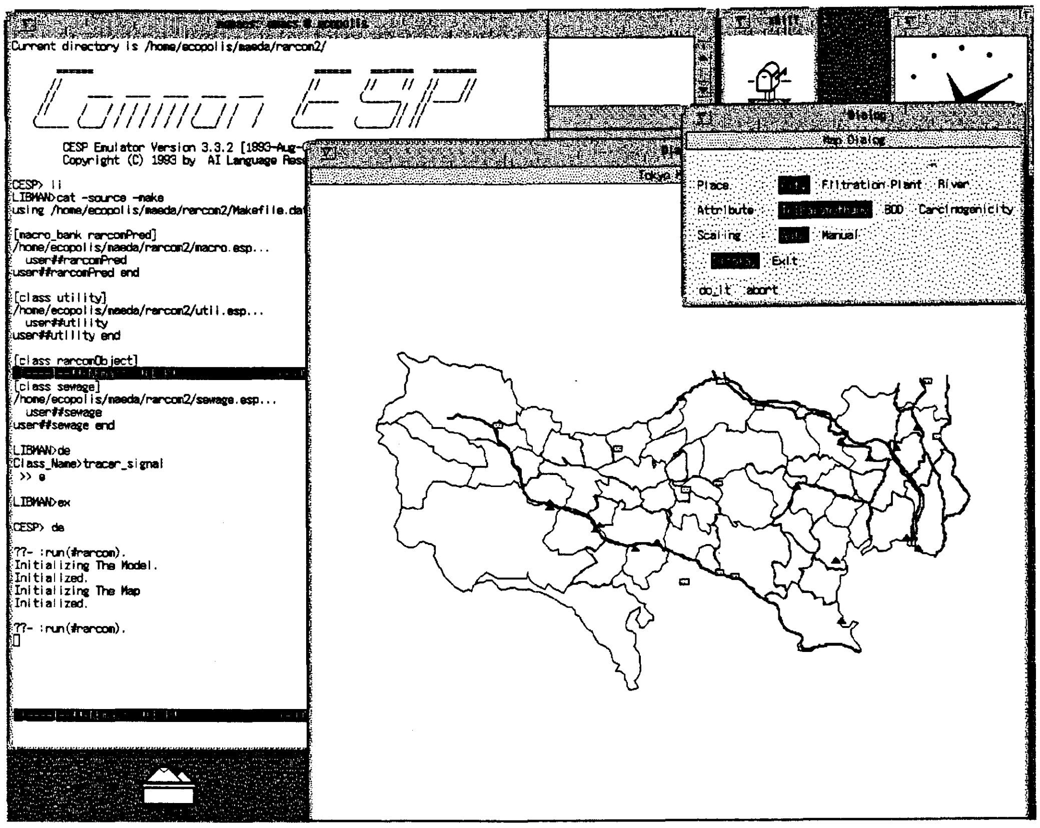
      The system is constructed on SPARCstation 2 workstation made by Sun Microsystems[Sun92], which has a UNIX-based operating system and graphical user interface based on X window system version 11(X11)[Schei88]. The workstation presents a platform by which the system can have a high performance reasoning capability and interactive graphical user interfaces.

- The system has graphical user interfaces.

  As mentioned in Section 2.3, a system supporting risk communication is required to have interactive and graphical user interface. The RARCOM complies with the requirement and has interactive graphical user interfaces. Dialogs between the system and users are done by using X11 window system and Emacs screen editor[Stall86]. Users can input messages or commands through pop-up menus, dialogbox windows, and Emacs editor, while the system outputs consequences of inference on the map of Tokyo metropolitan area or graphs displayed on the windows(Figure 5.1).

- Common ESP is used for constructing the system.

  ESP is a language developed in The Institute for New Generation Computer Technology(ICOT) for the personal sequential inference machine PSI[Chika84]. Common ESP (CESP) is a transplanted version of ESP on UNIX OS[AIR91]. Both ESP and CESP are declarative programming languages based on object-oriented programming and logic programming. The languages merge object-oriented programming and logic programming on the basis of multiple world model. Namely, a class in ESP/CESP is regarded as an axiom set represented by the logic programming



```
Current directory is /home/ecopolis/maeda/rarcom2/

       CESP Emulator Version 3.3.2 [1993-Aug-
       Copyright (C) 1993 by  AI Language Res

CESP> li
LIBMAN>cat -source -make
using /home/ecopolis/maeda/rarcom2/Makefile.dat

[macro_bank rarcomPred]
/home/ecopolis/maeda/rarcom2/macro.esp...
  user##rarcomPred
user##rarcomPred end

[class utility]
/home/ecopolis/maeda/rarcom2/util.esp...
  user##utility
user##utility end

[class rarcomObject]

[class sewage]
/home/ecopolis/maeda/rarcom2/sewage.esp...
  user##sewage
user##sewage end

LIBMAN>de
Class_Name>tracer_signal
 >> e

LIBMAN>ex

CESP> de

??- :run(#rarcom).
Initializing The Model.
Initialized.
Initializing The Map
Initialized.

??- :run(#rarcom).
```

Map Dialog

Place [✓ [illegible]] Filtration Plant River

Attribute [✓ Trihalomethane] BOD Carcinogenicity

Scaling [✓ Auto] Manual

[✓ Display] Exit

do_it abort

Figure 5.1: Screen image of executing RARCOM.

language, simultaneously, a program of ESP/CESP works as interactions in the society of *objects,* behavior of which are determined by axioms defined in classes involving the *objects.* In this language, classes are also considered as *objects,* that is, classes in CESP have slots and methods to represent the nature of them. The system employs the CESP language for programming the knowledge bases. Basic syntax of CESP is explained in Appendix A.

2. Environmental science features

   - The system deals with an environmental risk problem.
   - The system supports risk assessment and risk communication.

     Functions to support risk assessment are shown in Chapter 2. In particular, the system has *scenario analysis* function that enable users to simulate exposure situations in various conditions based on scenarios. Detailed discussion about it is shown in Section 5.2.1. On the other hand, the system has graphical and interactive user interface and *consultation* function to support risk communication. The former is as discussed before. Detailed discussion about the latter is shown in Section 5.2.2.
   - Knowledge base in the system holds knowledge that is based on the HI step and the DRA step in risk assessment, and holds the object model of urban water network.
   - The object model is based on Akiyama[Akiya89].
   - The system can simulate behavior of the network on the object model.

The system is constructed by *objects* shown in Figure 5.2. Detailed account of main *objects* is given below.

1. Class *object* `rarcom`

   The *object* manages all *objects* in the RARCOM. This involves a data base and a knowledge base to manage those *objects*. Users of the RARCOM interact primarily with the *object* or interact with the other *objects* through the *object*. Behavior of the system is determined by messages that are sent to the *objects* from it. Appendix B.1 shows a part of definition of the class *object* `rarcom`.

2. Class *object* `simModel`

   The module manages the object model of urban water network. This involves two kinds of knowledge, relation between components of the model and methods to execute simulation.

   relation between components This is one of three parts of knowledge of the composite object model in the system. The knowledge involves information representing topology of the model and knowledge about quantitative constraints between components. While the knowledge represent structure of the model, the knowledge also works as an interface between model components, domain knowledge bases, and bridging rules.

   methods for simulation There are two set of rules to execute simulations on the object model.

   (a) rules for constraint-based simulation procedure

   The first is a rule set representing the procedure to

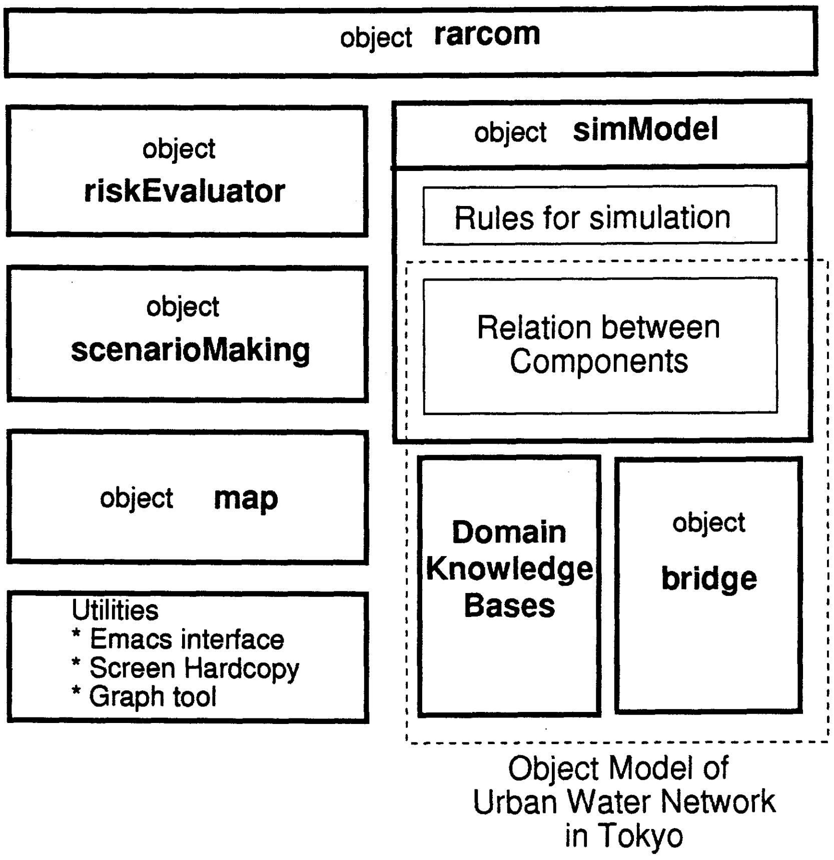


Figure 5.2: Overview of the RARCOM.

simulate by solving quantitative constraints associated with the model. The procedure to solve such constraints is, as shown in Section 3.4, equivalent to solve a set of simultaneous equations representing the model.

(b) rules for Monte Carlo simulation
The second is for Monte Carlo simulation. In the system, Monte Carlo simulation is done by iterate simulation procedures explained in (a). This procedure is shown in Section 3.5.

Appendix B.3 shows a part of definition of the class *object* `simModel`.

3. Object model of the urban water network

This is the object model of water network in Tokyo metropolitan area. The model is represented as a network, which is shown in Figure 3.8, of over 170 instances of `compo`, the class of model components. Data described in the model is based on observed data in Tokyo in 1987[Tokyo88a].
All components hold an axiom defining a predicate of common name, `process`, which representing distinct behavior of each component. Using the predicate `process`, each component represents respective characteristics as shown below.

riverSegment An instance of `riverSegment` represents behavior of a segment in river channel. When water flows in the segment from the upward terminal, It runs the flow through itself to the downward and varies attributes of the water flow by self-purification reaction in the river channel. In the model the self-purification reaction is modeled by Streeter-Phelps equation[Hall70].

fplant An instance of `fplant` represents behavior of a filtration plant. The *object* intakes a constant quantity of water flow from a `riverSegment` and supplies water to a number of `city` instances after purification treatment. Treatment capability of each fplant *object* is defined on the bases of actual treatment record in 1987 for corresponding filtration plant reported by Tokyo Metropolitan Government Waterworks Bureau[Tokyo88b]. The treatment involves processes of organic contaminants reduction by filtration, disinfection by chlorination, and THM generation brought by chlorination, which is modeled by equation of Urano, Wada and Takemasa[Urano83].

sewage An instance of `sewage` represents behavior of a sewage treatment plant. The *object* intakes a constant quantity of waste water from `city` *objects* and discharge treated water to a `riverSegment`. Treatment capability of each `sewage` *object* is defined on the bases of actual treatment record in 1987 for corresponding sewage treatment plants reported by Tokyo Metropolitan Government Sewerage Bureau[Tokyo88c].

city An instance of `city` corresponds to a city, a town or a ward in Tokyo metropolitan area. Each *object* has two kinds of information, list of `fplant` *objects* that supply water to the `city` and population in the city in 1987.

basin An instance of basin represents an area in which rainfall discharges into a corresponding `riverSegment`. The *object* brings to the corresponding `riverSegment` water and contaminant loads that do not pass through sewage plants, such as rainfall or domestic waste water that is not treated by sewage plants. Data described in the model is based on observed data in Tokyo in 1987[Tokyo88a].

Appendix B.5 shows parts of definitions of these classes.

4. Domain knowledge bases

The domain knowledge bases are another part of the composite object model. These involve the following five knowledge bases and are referred to by components in the object model.

hydrology The knowledge base holds knowledge in hydrology.

waterEnvEngineering The knowledge base holds knowledge in water environment engineering.

waterWorks The knowledge base holds knowledge in waterworks engineering.

sewageWorks The knowledge base holds knowledge in sewage works engineering.

administration The knowledge base holds knowledge related to municipal administration.

Each domain knowledge base contains definition of predicates and symbols used in itself. Definition of predicates are given as axioms. For example, predicate `filtration(Ins,F1,F2)`, which represents that if stuff `F1` is inputted into *object* `Ins`, the *object* outputs stuff `F2` after filtration, is defined by the following axiom in the domain knowledge base `waterWorks`.

```
filtration( Ins, F1, F2 ) :-
    :getAttr( Ins, F1, COD, 'CODL' ),
    :getSlot( Ins, treatmentRate, R ),
    :setAttr( Ins, F1, COD*R, 'CODL', F2 );
```

It means that the contaminant load of COD-Mn, which is represented by symbol `'CODL'`, in the output stuff `F2` is obtained by multiplying the load of COD-Mn in the input stuff `F1`, which is stored in variable `COD`, and the rate of filtration `R`, which is given as the value of slot `treatmentRate` of the *object* `Ins`.

Definition of symbols are given by axioms using predicate named `dictionary`. Each axiom defines symbol used in domain knowledge bases and explains its semantics. For example, the symbol `'CODL'` is defined in the domain knowledge base `waterWorks` as follows.

```
dictionary( 'CODL', "kg/day",
    "potassium permanganate consumption load per day",
    [flow, 'COD-Mn'], [Flow, COD], CODL,
    multiply( COD, Flow, CODL) );
```

The first argument represents the symbol, the second does the unit for the attribute represented by the symbol, and the third does an explanation of the attribute. If the attribute can be obtained from calculation using other attributes represented by other symbols, the rest of the arguments are used. In this example, it means that if value of the attribute `flow` is set in variable `Flow`, and the attribute `COD-Mn` is done in variable `COD`, the attribute `'CODL'` is obtained as value of the variable `CODL` by invoking the predicate `multiply( COD, Flow, CODL)`.

Appendix B.2 shows an example of the domain knowledge bases, a part of definition of the class `waterWorks`.

5. Class *object* `bridge`

The *object* `bridge` is the last part of the composite object model. This *object* is implemented as a class, i.e. an axiom set. These are six sub-classes of class `bridge`, which actually hold bridging rules. Each sub-class holds bridging rules particularly between two of the domain knowledge bases. The following shows the sub-classes.

admin_sewag The *object* holds bridging rules between administration and sewage works engineering.

waterE_waterW The *object* holds bridging rules between water environment engineering and waterworks engineering.

sewag_waterE The *object* holds bridging rules between sewage works engineering and water environment engineering.

admin_waterW The *object* holds bridging rules between administration and waterworks engineering.

hydro_waterW The *object* holds bridging rules between hydrology and waterworks engineering.

hydro_sewag The *object* holds bridging rules between hydrology and sewage works engineering.

Appendix B.4 shows parts of definitions of the class `bridge` and a sub-class `waterE_waterW`.

6. Risk evaluator: class *object* `riskEvaluator`

   The *object* `riskEvaluator` is a knowledge base that holds rules to evaluate health effects of carcinogen THM. The rules are based on consequences of the DRA step in risk assessment. The rules held in the RARCOM are based on WHO[WHO84].

7. Scenario making subsystem: class *object* `scenarioMaking`

   Scenario making subsystem executes simulation on the object model in conditions determined by various scenarios. The *object* holds templates of scenarios, which are displayed in the window system for users to select and customize one of those. The *object* determines a scenario through interaction between users, then send a message that requests *object* `simModel` to simulate along the scenario. Appendix B.7 shows a part of definition of the class *object* `scenarioMaking`. Detailed explanation about the *object* is shown in the next section.

8. Map subsystem: class *object* `map`

   The *object* `map` displays graphically distribution of carcinogens, contaminants, and carcinogenicity in Tokyo on the window system. Consequences of simulations or risk evaluations can be displayed on the *object.*

These *objects* are organized in a class hierarchy in consistent manner with relation between domain knowledge bases and components in the composite object model. Namely, while super classes give primitive axioms that is inherited by its sub-classes, sub-classes define their behavior by using the inherited axioms. Figure 5.3 shows the class hierarchy of these *objects.*

5.2 Capabilities of the RARCOM

5.2.1 Scenario analysis

This paper defines the *scenario analysis* as a process to check consequences of scenarios that determine conditions or alternatives affecting the concerned *object* by simulating behavior of the *object* in the conditions. The RARCOM has a capability of scenario analysis to support the EA step in risk assessment. In this paper, scenarios determine conditions for the object model, such as the boundary condition for the model, parameters of components, and topological relationship between components.

The scenario analysis is executed by messages sent by the *object* `rarcom` to the following three *objects,* `scenarioMaking`, `simModel`, and `riskEvaluator`. Figure 5.4 shows processes involved in scenario analysis.

At first, the *object* `rarcom` sends a message to the *object* `scenarioMaking`. The *object* `scenarioMaking` interacts with users and lets them determine a condition for the simulation on the basis of scenario templates held in the *object.* Based on the determined condition, `scenarioMaking` generates messages that request `simModel` to set or to change parameters of components,

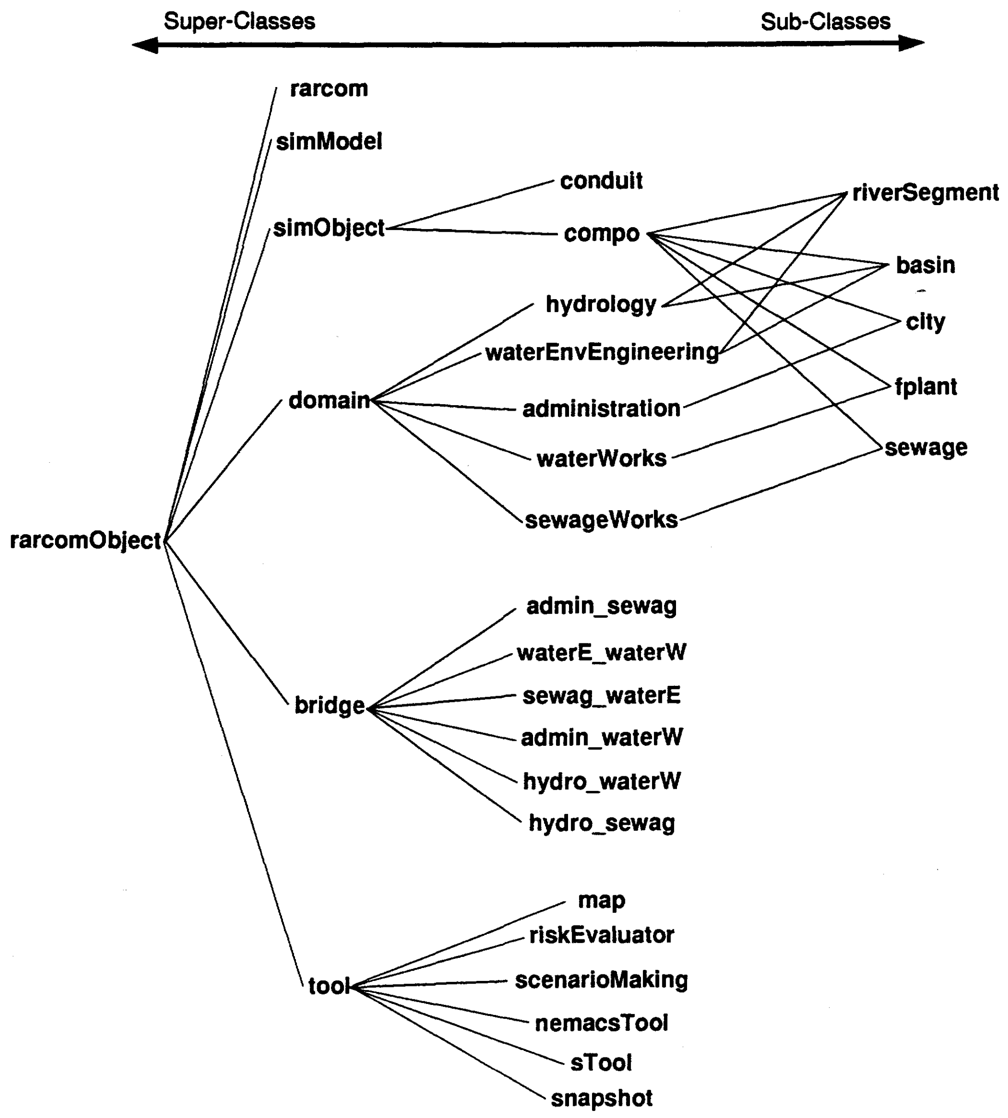


Figure 5.3: Class hierarchy of the RARCOM.

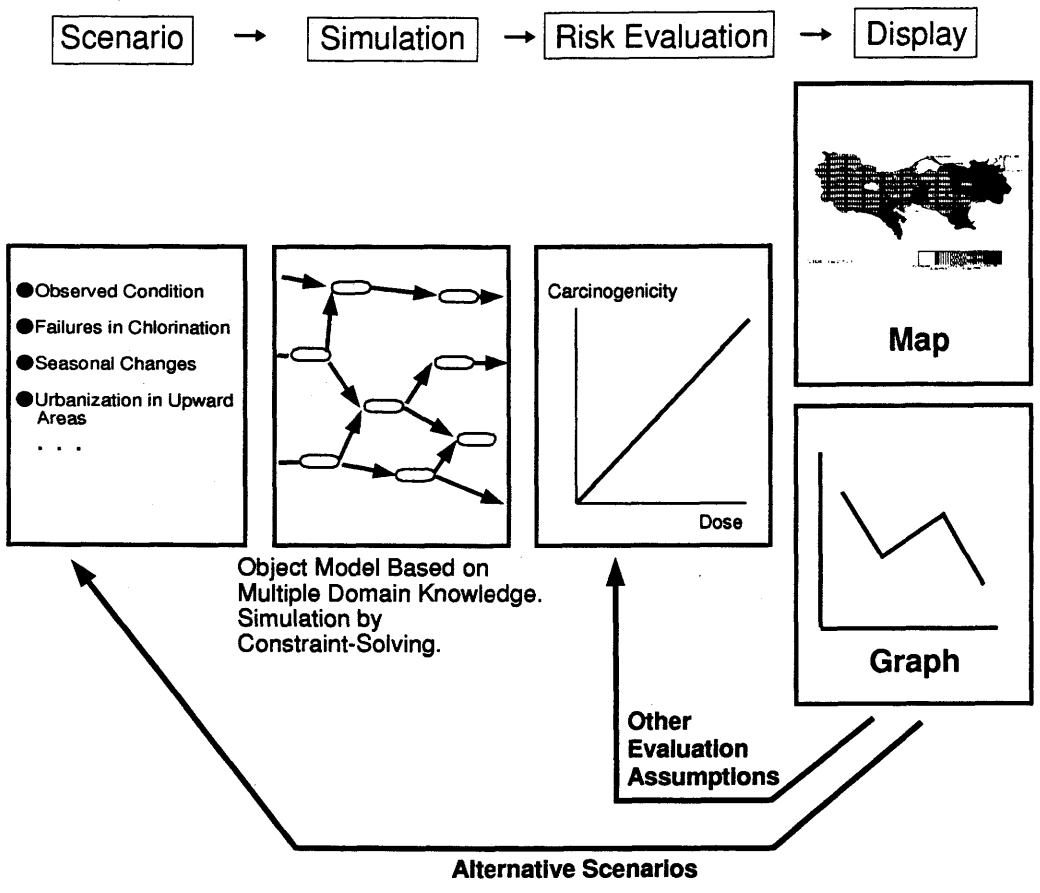

Scenario
Simulation
Risk Evaluation
Display
●Observed Condition
●Failures in Chlorination
●Seasonal Changes
●Urbanization in Upward Areas
. . .
Carcinogenicity
Dose
Map
Graph
Object Model Based on Multiple Domain Knowledge. Simulation by Constraint-Solving.
Other Evaluation Assumptions
Alternative Scenarios

Figure 5.4: Life cycle of scenario analysis.

variables determining the boundary condition, or variables determining relation between components. This messages are for defining conditions of the *object* before simulation. After conditions are set in the model by the messages, `scenarioMaking` sends to `simModel` another message that request to execute simulation. The *object* `simModel` receiving the message executes a simulation procedure under the determined condition. The simulation procedure is based on the algorithm `simCompo`, which is discussed in Section 3.4. This algorithm is implemented as shown in Appendix B.6. Output of the simulation is sent to the *object* `riskEvaluator`, which evaluates distribution of carcinogenicity on the basis of the output. Finally the consequence of the evaluation is stored in the system knowledge base. Users can display the consequence on the window system by using the *object* `map` or the *object* for data processing. Furthermore, if another scenario or another evaluation assumption is considered, then another cycle of scenario analysis will be executed.

The RARCOM currently holds the following scenario templates, by which simulation conditions are determined.

1. Observed condition in 1987.

   The scenario presents a condition that is actually observed in Tokyo in 1987. That is considered as the control condition for the other scenarios.

2. Scenario of summer and winter.

   Seasonal changes of condition, such as water temperature or rate of flow in rivers, causes changes in THM generation. To estimate the changes, the RARCOM holds scenario template representing conditions in summer and in winter.

3. Accidents in filtration plants.

   Any accidents in filtration plants may affect THM generation. The *object* `scenarioMaking` holds the scenario of excess chlorine input in the disinfection process as one of such accidents.

4. Urbanization in upward areas.

   Effect of urbanization in upward areas of filtration plants is one of the most controversial problems related with THM risks. To estimate the effect, scenario holds scenarios representing increase of contamination in upper reach of rivers.

5.2.2 Consultation

It is the most different point of knowledge-based simulations with conventional simulations that simulation models can involve not only description of behavior and structure of the modeled object, but also theories or assumptions underlying in the model description. It makes knowledge-based simulation systems answer questions like the following; "Why the model does such a behavior?" or "How the consequences are derived?" This paper calls the capability *consultation.*

In the RARCOM, All *objects* in the model have two methods, `:whyDoneIt` and `:howDoneIt`, and a slot, `reference`, as implemented for the consultation capability. When the message `:whyDoneIt` is sent to an *object* in the model, the method returns why the *object* does the previous behavior, namely, information of event that causes the previous behavior of the *object.* At the same time, when the message `:howDoneIt` is sent to an *object,* the method returns how the *object* does the previous behavior, namely, information related to procedure used for the previous behavior of the *object*. The slot `reference` holds the reference information such as sources information for modeling the *object* and related literatures. Users can examine validity of behavior of the model or premises underlying the model by using the methods or accessing the slot.

Methods `:howDoneIt` and `:whyDoneIt` is implemented as follows. First, there is a premise to implement the two methods in the system.

Premise for :howDoneIt and :whyDoneIt The knowledge system must have a log file in which the system records history of messages passing during a simulation. The history involves messages sent to *objects,* name of *objects* that receive the messages, and name of *objects* that send the messages.

Second, those methods are implemented by the following algorithms on the basis of the premise.

Algorithm 5.1: :whyDoneIt Method `:whyDoneIt(Obj,Source,Mes)` determines the value of arguments `Source` and `Mes` from given `Obj` representing an *object* by the following procedure.

1. Retrieve from the log file the newest message that was sent to *object* `Obj`, and the name of *object* that sent the message.
2. Let `Source` denote the retrieved *object* name and let `Mes` does the message.

Algorithm 5.2: :howDoneIt(Obj,Axiom) Method `:howDoneIt(Obj,Axiom)` determines value of argument `Axiom` from given `Obj` representing an *object* by the following procedure.

1. Retrieve from the log file the newest message that was sent to *object* `Obj`, and Let `Mes` denote the message.

2. Call method `:howDoneIt(Obj,Mes,Axiom)`.

Algorithm 5.3: :howDoneIt(Obj,Mes,Axiom) Method

`:howDoneIt(Obj,Mes,Axiom)` shows an axiom associated with message `Mes`, which is defined in *object* `Obj`, as value of `Axiom`.

1. Search an axiom that corresponds message `Mes` from the source file of *object* `Obj`.

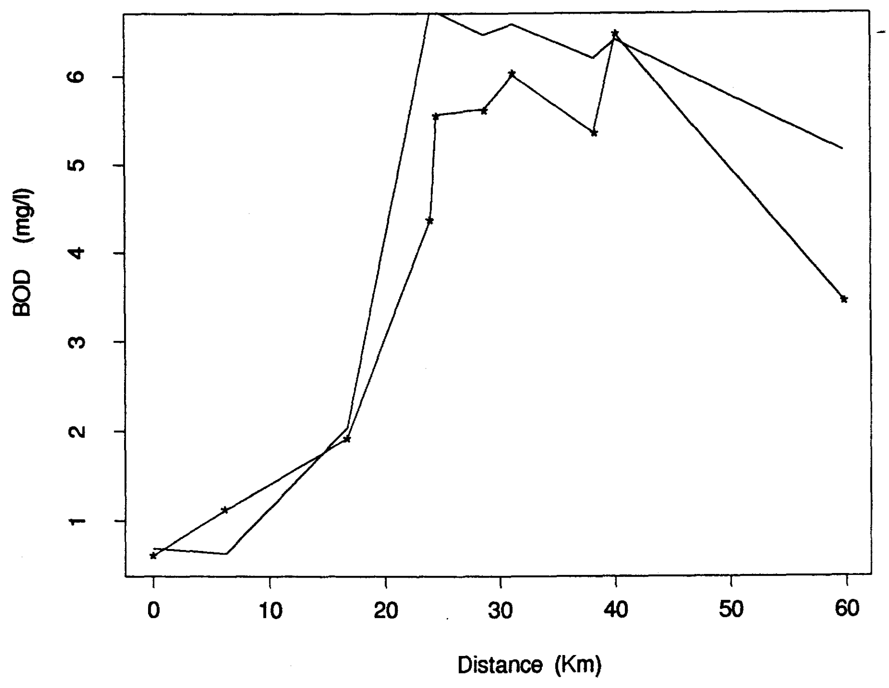
2. Let `Axiom` denote the axiom.

Furthermore, all *objects* in the model have method `:getSlot(Obj,Slot,Val)`, which tells value of slot `Slot` in *object* `Obj` as returned value of variable `Val`. By using the method, users can obtain information stored in the model. For example, methods `:getSlot(Obj,upward,Val)` and `:getSlot(Obj,downward,Val)` tell users about *objects* connecting upward side and downward side of *object* `Obj` respectively. Users can follow the device topology by using those methods. Simultaneously, method `:getSlot(Obj,reference,Val)` tells literatures or other information referenced by *object* `Obj`. Users can find source information underlying the modeling by using the method.

5.3 Examples of Execution

5.3.1 Examples of simulation

The model in the RARCOM is set on the basis of observed data in Tokyo in 1987. While almost all parameters in the components are, as mentioned in Section 5.1, set as behavior of the components fit the observed data, the



The x-axis represents the distance from the Chofu Bridge (the most upper reach in the model) along Tamagawa.
The solid line with "*" represents BOD observed in 1987.
The other line represents BOD estimated by the model.

Figure 5.5: BOD in Tamagawa river.

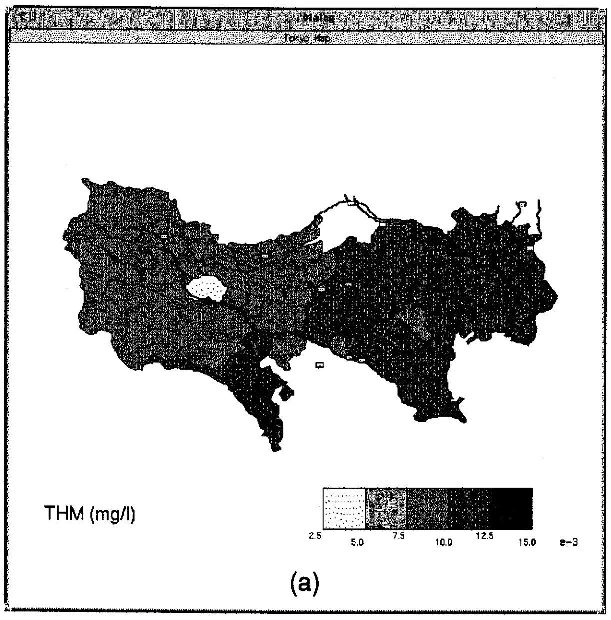
coefficient of self-purification in rivers, which is a parameter of the class `riverSegment` and difficult to be estimated from the observed data, is determined as possibly as the behavior of the whole model fits the data observed in river systems. Namely, the whole model is calibrated by using the observed data in rivers. Figure 5.5 shows a comparison between the observed data[Tokyo88a] and estimated data by the model in Tamagawa river system. Mean squared-error of these data is 1.17; goodness of fit of the estimated data for the observed data can be considered as acceptable.

Simulations in the RARCOM are executed on the model that is set as shown above. Figure 5.6 (a) shows estimated distribution of THM in the scenario in which the boundary condition for simulations, for example, water flow at the most upper reach of the rivers and waste water discharged from the basins is set by using observed data in 1987. Table 5.1 shows a comparison between the estimated THM and the observed one in several wards[Tokyo88b]. Evaluation of the comparison is difficult because of this small number of observed THM data, however, it may suggest that the estimate of THM by the model is relatively lower than the real value. It is considered that this lower estimation is possibly because of lack of data about background THM level. The model has no data of the background THM level, that is, concentration level of naturally occured THM, because there are no available data of the background THM level in Tokyo. However, this background level data will be needed to estimate the real concentration of THM distribution.

Another simulation example in which contamination in upper reach of Arakawa river, one of the source of water service in Tokyo, increases is shown below. This contamination in the upper reach of the source water causes increase of carcinogenicity in drinking water in the area. The causal chain

Table 5.1: Observed and estimated THM in wards.

ward	observed THM(mg/l)	estimated THM(mg/l)
adachi	0.032	0.012
	0.035	
arakawa	0.033	0.012
bunkyou	0.041	0.010
	0.031	
chuuou	0.042	0.012
	0.042	
edogawa	0.037	0.012
	0.041	
itabashi	0.033	0.011
kita	0.037	0.012
koutou	0.038	0.012
	0.038	
minato	0.032	0.011
	0.036	
nerima	0.028	0.010
	0.026	
setagaya	0.018	0.010
	0.034	
	0.028	
	0.034	
	0.038	
shibuya	0.025	0.010
shinagawa	0.036	0.011
shinjuku	0.044	0.011
suginami	0.035	0.011
ohta	0.024	0.011
	0.028	



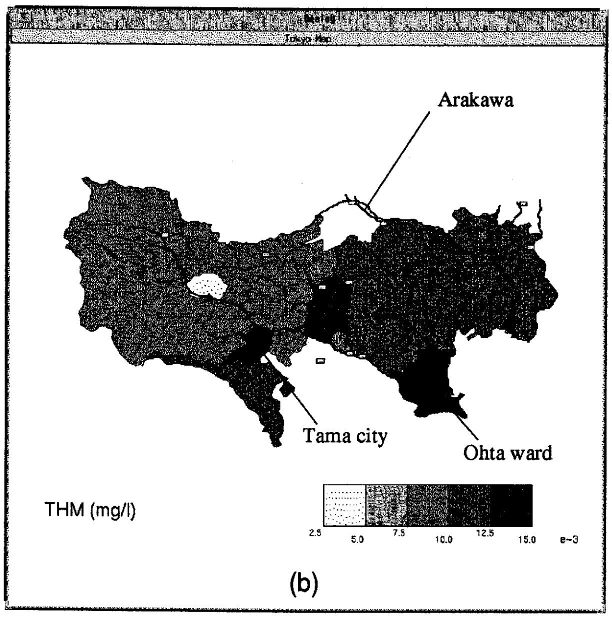


Figure 5.6: THM distributions in (a) observed condition in 1987 and (b) condition in which contamination in upper reach of Arakawa river increases two times.

from contamination in the upper reach to the carcinogenicity involves several processes associated with different domain knowledge, e.g.,

- increase of sewage in the upper reach(sewage works engineering),
- increase of river contamination(water environment engineering),
- flow from the upper reach to intake point for the filtration plant (hydrology),
- filtration process and generation of THM(waterworks engineering),
- carcinogen distribution by water supply in Tokyo(administration), and
- helth effect in each city in Tokyo(social medicine).

The system can cope with such problems. Figure 5.6 shows the consequence of exposure assessment based on the condition in which organic contamination in the upper reach of Arakawa river increases two times. The figure shows that there is high concentration of THM in Tama city or Ohta ward, which are far from Arakawa river. The consequence, which may be difficult to point out even for water service experts, is regarded because of complex water supply network in Tokyo. According to the guideline of THM proposed by WHO[WHO84], THM in Tama city or Ohta ward in this case can be considered as not a problem. However, this case shows sensitivity of these cities to water quality of Arakawa river. It suggests that administrators of drinking-water in these cities should take care of water quality of the river.

As shown in Figure 5.4, based on the consequence, the system can estimate carcinogenicity distribution in Tokyo. That is shown in Figure 5.7.

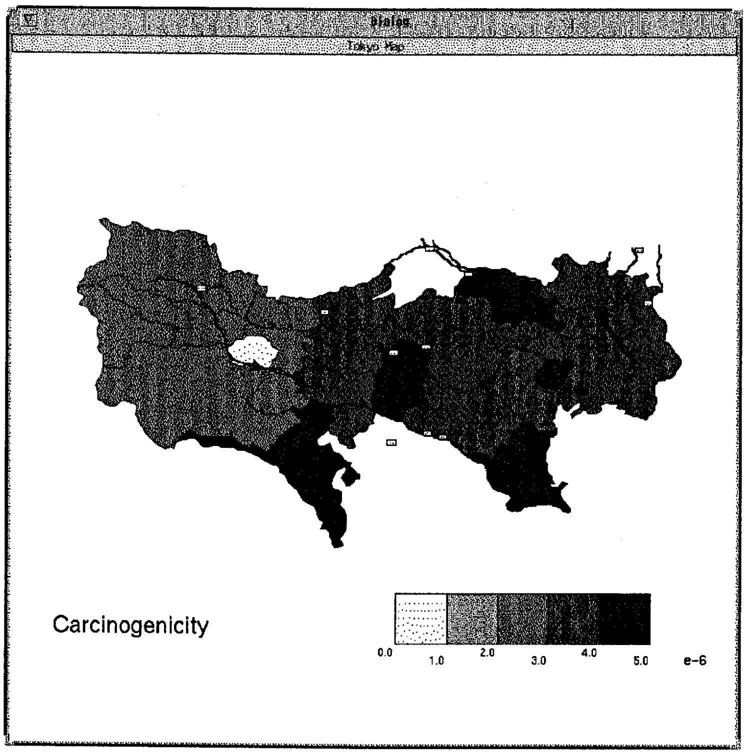


Figure 5.7: Carcinogenicity distribution in condition shown in Figure 5.6.

5.3.2 An example of consultation

Furthermore, the system can help users to understand processes involved in the simulation by the consultation capability. For example, if a user sends to the *object* representing Ohta ward a message,

```
:getSlot( ohta, upward, UP ),
```

where the symbol **ohta** represents *object* of Ohta ward, the message will return,

```
UP = [{asaka_f,ohta}].
```

That represents that the *object* ohta connects a conduit *object* {asaka_f,ohta}, which is a conduit connecting between the *object* asaka_f and the *object* ohta. Moreover, if the user sends the following messages to the *object* asaka_f;

```
:getSlot( asaka_f, name, Name ),
:getSlot( asaka_f, upward, Intake ),
```

then the user obtains results;

```
Name   = "Asaka Filtration Plant",
Intake = [{ar1,asaka_f}].
```

The results represent that the *object* asaka_f is a model of the Asaka Filtration Plant and the *object* connects the conduit *object* {ar1,asaka_f}, which connects ar1 representing a segment of Arakawa river and the *object* asaka_f, at its upward side. Namely, It means that Ohta ward, which is outside of the basin of Arakawa river, is directly supplied with drinking water from the Asaka Filtration Plant that intakes source water from Arakawa river.

Simultaneously, a message

```
:getSlot( setagaya, upward, UP2 ),
```

returns,

```
UP2 = [{asaka_f,setagaya},{kinutashita_f,setagaya},
       {kinutaue_f,setagaya},{misato_f,setagaya},{sakai_f,setagaya}].
```

It means that Setagaya ward, which is a neighbor of Ohta ward and is not significantly affected by the contamination of the case, is supplied drinking-water from Asaka Filtration Plant and other four filtration plants.- It suggests that impact of the contamination of supplied water from Asaka Filtration Plant is comparatively small in the ward because concentration of the contaminant is decreased by confluence of water supplied from the other filtration plants. It may be considered that complexity of the urban water network emerges the difference of THM level between Ohta and Setagaya. The user can understand such situations by using these methods.

Moreover, if the user sends a message,

```
:howDoneIt( asaka_f,process(asaka_f,U,D),Axiom ),
```

the method tells that the axiom associated with the message `process(asaka_f,U,D)` is;

```
process( asaka_f,U,D ) :-
    filtration( asaka_f,U,D0 ),
    disinfection( asaka_f,D0,D1 ),
    thm_generation( asaka_f,D1,D );
```

which means that water treatment processes of filtration plants involve filtration, disinfection, and THM generation. Moreover, the user also can know detailed process involved in `thm_generation` by using method,

```
:howDoneIt( asaka_f, thm_generation(asaka_f,F1,F2), Axiom2 ).
```

This method will return the following result.

```
thm_generation( asaka_f,F1,F2 ) :-
    :getAttr( asaka_f,F1,Humin,humin ),
    !,
    :getSlot( asaka_f,fe_Cl,Cl ),
    :'THM_from_humin'( #waterEnvEngineering,Humin,Cl,THM ),
    :setAttr( asaka_f,F1,THM,'THM',F2 );
thm_generation( asaka_f,F1,F2 ) :-
    :getAttr( asaka_f,F1,TOC,'TOC' ),
    !,
    :getSlot( asaka_f,fe_Cl,Cl ),
    :getSlot( asaka_f,pH,PH ),
    :getSlot( asaka_f,tmp,TMP ),
    :getSlot( asaka_f,period,T ),
    :'Urano_Takemasa'(#waterEnvEngineering,TOC,Cl,PH,TMP,T,THM),
    :setAttr( asaka_f,F1,THM,'THM',F2 );
thm_generation( asaka_f,F1,F2 ) :-
    :getAttr( asaka_f,F1,COD,'COD-Mn' ),
    :bridge( #waterE_waterW,#waterEnvEngineering,
             F1,'COD-Mn',COD,'TOC',TOC ),
    !,
    :getSlot( asaka_f,fe_Cl,Cl ),
    :getSlot( asaka_f,pH,PH ),
    :getSlot( asaka_f,tmp,TMP ),
    :getSlot( asaka_f,period,T ),
    :'Urano_Takemasa'( #waterEnvEngineering,TOC,Cl,PH,TMP,T,THM ),
    :setAttr( asaka_f,F1,THM,'THM',F2 );
```

It means that there are three axioms in the knowledge base to determine concentration of generated THM. In the execution of simulation, the inference mechanism will select one of them according to the run-time condition in the knowledge base.

It must be noted that these three axioms are defined not in the class `fplant`, but in the class `waterWorks`, which is a super-class of `fplant`. The reason why these axioms are accessed by `:howDoneIt` message sent to `asaka_f`, is the inheritance relationship between `waterWorks` and `fplant`. The method `:howDoneIt` can access these axioms because these are inherited by the class `fplant`.

As shown above, users of the system can explore contents of the knowledge bases as deeply as they need.

5.3.3 An example of the bridging rule

In the example shown above, a bridging rule is used on the conduit between the *object* `ar1` and `asaka_f`. By using the following message, the users can know the *object* holding the bridging rule(see Appendix B.4:

```
:bridgeName(#bridge,ar1,asaka_f,Bridge).
```

This message will return the result:

```
Bridge = #waterE_waterW,
```

where `#waterE_waterW` is the *object* of the class `waterE_waterW`, which is the knowledge base of the bridging rules between water environment engineering and waterworks engineering.

Then, the message;

```
:howDoneIt( #waterE_waterW,
 bridge(#waterWorks,Flux,'BOD',Data,'COD-Mn',COD), A3 ),
```

returns the following bridging rule, which used in the conduit connecting between the *object* ar1 and the *object* asaka_f.

```
bridge( #waterWorks,Flux,'BOD',Data,'COD-Mn',COD ) :-
    COD is 0.887 * Data + 5.073;
```

Moreover, the following message:

```
:getSlot(#waterE_waterW,reference,Ref),
```

returns,

```
Ref =
"Tokyo Metropolitan Government Environmental Conservation Bureau.
   {Showa 62 Nendo Kokyo You Suiiki No Suishitsu Sokutei Kekka.}
   [Quality of Public Water in 1987.] Tokyo Metropolitan Government.
   1988.
Tokyo Metropolitan Government Waterworks Bureau.
   {Jigyou Nenpo Showa 62 Nendo.}
   [Annual Report of the Water Service, 1987.]
   Tokyo Metropolitan Government.
   1988.",
```

which suggest that the bridging rule shown above is based on data involved in those two literatures. Namely, user can easily know not only how the gaps over different domains are bridged, but the source data of the bridging rules. In other words, users can examine validity of the rules. If users consider that the rule should be changed to the other rule based on different data, operation the user have to do is only the modification of the rule in the *object* `waterE_waterW`. As shown above, the explicit representation of the bridging rules based on the composite object model enables user to access and to examine the rules, which have so far been embedded in conventional simulation programs in environmental science.

Chapter 6

Conclusions

I have developed the RARCOM, a knowledge system to support risk assessment and risk communication for the THM problem. In order to design the RARCOM, requirements for knowledge systems to support risk assessment and risk communication have been discussed. It has been pointed out that knowledge-based simulation can be significantly effective to support the EA step and the RC step in risk assessment.

In particular, to cope with interdisciplinary nature of the THM problem, I have developed the composite object model, a modeling framework to deal with multiple domain knowledge. This framework enables a knowledge system, not only to construct an object model on the basis of multiple domain knowledge, but to bridge over gaps between concepts involved in the multiple domains. By using the composite object model, the RARCOM can execute knowledge-based simulation to support the EA step and the RC step in risk assessment. This framework also provides a foundation of a user interface to examine meta-information for the modeling; such as assumptions used for the modeling and interpretation rules for water quality indices between multiple domains.

There have been several researches, for instance; Fedra and Winkelbauer [Fedra91], Fiksel and Covello [Fikse87], and Kainuma, Nakamori and Morita[Kainu91], that aim to construct knowledge systems to support environmental impact assessment or decision making for environmental policies. However, only a few researches discuss the importance of the simulation functions in environmental impact assessments by utilizing knowledge systems to understand phenomena in the environment and estimated results. The model to estimate impacts of Tokyo Bay Development programs, which is constructed by Kainuma, Nakamori and Morita[Kainu91], is one of such systems. This model, which aims to predict or estimate impacts of various scenarios, differs from the RARCOM in its modeling approach. The RARCOM employs the postulated, namely top-down, approach in which the modeler uses the established or given hypothetical knowledge of hazard identification, dose-response assessment, and model structure of the urban water network, as the foundation to model a set of objects in the real world. The model of Kainuma, Nakamori and Morita employs the discovery, bottom-up, approach in which the modeler constructs the model in dialog with the system on the basis of observed data. RARCOM did not employ the latter approach because sufficient knowledge to construct the model could be obtained in advance. There are however several cases in risk phenomena in which mechanisms and structures of those are hard to identify. To deal with such cases, the discovery approach is probably required to construct models representing the risk phenomena, because it is difficult to obtain premise knowledge to construct models in such cases.

There are a few problems to be solved to utilize the system in risk assessment and risk communication in the real world. First, the system has too small amount of data and scenarios to be used in actual conditions. The system will have to store more scenarios and data to deal with more various conditions. Moreover, validity of the model should be checked in such various conditions.

Second, the system should have a user interface to deal with natural languages rather than a specific programming language. As shown in Chapter 2, systems supporting risk communication are required to have natural language user interface. This capability is unfortunately not implemented in the system. If the system intends to be actually used by ordinary people concerned with the risk problem, the system should have natural language interface. It is regarded that the *intelligent front-end*[Bundy84], which was used in the *Eco* project[Robert91], is adequate to implement such a capability.

Third, Applicability of the system in actual situations should be examined. To do this, the system will have to be reviewed by a variety of users; such as, municipal administrators, experts of disciplines associated with the risk problem, and residents in Tokyo.

Furthermore, The following two problems should be tackled to enhance capabilities of systems supporting risk assessment and risk communication for the future. First, the modeling framework of composite object model should be applied to other types of systems. In this paper, the composite object model is developed on the premise that the system is bounded with a constraint of component-conduit model. However, the other problems associated with multiple domain knowledge may be modeled on the basis of the other modeling schemata. For instance, risk assessment of the ozone layer destruction may need multiple layers model or three-dimensional model representing the atmosphere, which are differ from component-conduit model. The framework to deal with multiple domain knowledge should, therefore, be also extended to be applied to such systems.

Second, knowledge compiling[Yamag92] from declarative representation of knowledge to procedural one should be explored. The RARCOM deals with only the constraint solving in the static model. Models used in environmental science need sometimes to deal with dynamic behavior of various phenomena. If domain knowledge bases hold a set of ordinary differential equations describing such behavior, the system must compile the declarative knowledge representing these equations into the procedural knowledge for numerical analysis to compute behavior of variables. Such knowledge compiling may include processes of formula manipulation.

It is considered that there will be two future directions following this research. The first is organization and integration of environmental domain knowledge bases. The second is to develop an approach to *very large knowledge bases*(VLKB) from the real world problems.

The first is associated with an environmental scientific aspect of the future directions. As shown in this paper, domain knowledge bases in the composite object model can work as repositories of model-independent and problem-independent knowledge in various domains. Therefore, environmental scientists can organize and integrate knowledge in disciplines in environmental science by using the modeling framework. Such environmental knowledge bases can be utilized to solve various environmental problems in different conditions of the real world.

The other is related to the knowledge engineering aspect. VLKB research is a progressing field in knowledge engineering. This aims to construct multipurpose and reusable knowledge bases holding large number of and a variety of ordinary knowledge, like encyclopedias, and to make the knowledge bases foundations of various problem solving. Though several researches are being done related to VLKB on the basis of seed technologies in knowledge engineering[Lenat90, Yokoi90, Miki93], there are few researches based on needs to solve actual problems by using VLKB. On the other hand, the RARCOM, which has not stored "very large" knowledge bases, is a system to solve a real world problem by using multipurpose and problem-independent knowledge bases, which can be regarded as parts of VLKB. It is considered that both approaches to VLKB from the seeds and the needs will complement each other. Therefore, such researches as the RARCOM, which take the latter approaches, will become significant factors for a future development of VLKBs.

Acknowledgements

This dissertation is based on collaborated researches with my supervisor Professor Saburo Ikeda. I wish to express my gratitude to him for introducing me to the research about environmental risks and encouraging me in my study.

RARCOM is constructed on the bases of data shown in literatures [Tokyo88a, Tokyo88b, Tokyo88c], which are provided by Tokyo Metropolitan Government Environmental Conservation Bureau, Waterworks Bureau, and Sewerage Bureau. The system is implemented by using the Common ESP language, which is produced and provided by AI Language Research Institute. I wish to acknowledge their cooperation.

I would like to thank to Hayatake Tsubokawa for discussion with him that became the foundation of the model construction for RARCOM, and to the following people for their valuable comments; Professor Susumu Kunifuji and Assistant Professor Teruo Matsuzawa in Japan Advanced Institute of Science and Technology, Hokuriku, Assistant Professor Noriyuki Matsuda, Assistant Professor Yutaka Nakamura, Lecturer Mitsuru Ohta, and Lucturer Xiao-Ping Zheng in the University of Tsukuba.

I would also like to thank Juan José who proofread this paper, and Michael Björn who introduced him to me.

Finally, I wish to thank my wife Mana for her encouragement and understanding.

Bibliography

[AIR91] AI Language Research Institute. *CESP Gengo.*[CESP Language.] Tokyo: AIR, 1991.

[Akiya89] Akiyama, Toshiko. "Toshiiki No Mizu No Junkan To Risk No Hassei."[Water Cycle and Water-borne Risks in Urban Areas.] *Juuten Ryouiki Kenkyuu* Ningen-Kankyou Kei *Kenkyuu Houkoku Syuu* [Research Report of the Research Project on a Priority Area "Man-Environment Systems"]. G009. Tokyo: Ministry of Education, 1989. 21-52.

[Akiya91] Akiyama, Toshiko. "Fluctuations in Environmental Qualities and Public Services." *Japanese Jour. of Risk Analysis.* 3(1) (1991): 90-95.

[Bundy84] Bundy, Alan. "Intelligent Front Ends." In ed. M. Bramer. *Research and Development in Expert Systems.* Cambridge: Cambridge University P., 1984. 193-203.

[Check81] Checkland, Peter B. *Systems Thinking, Systems Practice.* New York: John Wiley, 1981.

[Chika84] Chikayama, Takashi. "Unique Features of ESP." *Proc. of The International Conference on Fifth Generation Computer Systems*

1984. Tokyo: Institute for New Generation Computer Technology, 1984. 292-298.

[Colli87] Collins, J. and K. Forbus. "Reasoning About Fluid Via Molecular Collections." *Proceedings, AAAI-87: Sixth National Conference on Artificial Intelligence.* Menlo Park, California: American Association for Artificial Intelligence, 1987. 590-594.

[Covel87] Covello, Vincent T., Detlof von Winterfeldt, and Paul Slovic. "Communicating Scientific Information about Health and Environmental Risks: Problems and Opportunities from a Social and Behavioral Perspective." In ed. V. Covello, A. Moghissi, and V.R.R. Uppuluri. *Uncertainty in Risk Assessment, Risk Management, and Decision Making.* New York: Plenum, 1987. 221-239.

[deKle83] de Kleer, J. and J. Brown. "Assumption and Ambiguities in Mechanistic Mental Models." In ed. D. Genter and A.L. Stevens. *Mental Models.* Hillsdale, New Jersey: Lawrence Erlbaum, 1983. 155-190.

[Fedra91] Fedra,K. and Winkelbauer,L. *Expert Systems for Environmental Screening.* Laxenburg, Austria: International Institute for Applied Systems Analysis, 1991.

[Fikse87] Fiksel, Joseph and Vincent T. Covello. "Expert Systems and Knowledge Systems for Risk Communication." *The Environmental Professional.* 9 (1987): 144-152.

[Fox90] Fox, M. et al. "Knowledge-Based Simulation: an Artificial Intelligence Approach to System Modeling and Automating the Simulation Life Cycle." In ed. L.E. Widman et al. *Artificial Intelligence, Simulation, and Modeling.* New York: John Wiley, 1989. 447-486.

[Haggi90] Haggith, Mandy. "Sheltering Under a Tree." *Proc. of Pacific Rim International Conference on Artificial Intelligence '90.* Tokyo: Japanese Society for Artificial Intelligence, 1990. 51-56.

[Hall70] Hall, Warren A. and John A. Dracup. *Water Resources Systems Engineering.* New York: McGraw-Hill, 1970. 341-356.

[Hopgo93] Hopgood, Adrian A. *Knowledge-Based Systems for Engineers and Scientists.* Boca Raton, Florida: CRC, 1993. 202-217.

[Hori90] Hori, Koichi and Setsuo Ohsuga. "Articulation Problem – a Basic Problem for Information Modelling." In ed. Hannu Kangassalo et al. *Information Modelling and Knowledge Bases.* Amsterdam: IOS, 1990. 36-44.

[Ikeda89] Ikeda, Saburo. "Risk Hyouka To Kanri No Tameno System-Ronteki Wakugumi." [The System Theory Framework for Risk Assessment and Management.] *Juuten Ryouiki Kenkyuu* Ningen-Kankyou Kei *Kenkyuu Houkoku Syuu.* G009. Tokyo: Ministry of Education, 1989. 93-122.

[Kainu91] Kainuma, M., Y. Nakamori, and T. Morita. "Development of an Intelligent Decision Support System for Environmental Modeling and Planning." *Research Report from the National Institute for Environmental Studies, Japan.* 128 (1991): 107pp.

[Kimur62] Kimura, Toshiaki. "Choryu Kansu Hou (II)"[Storage Function Method (II)]. *Doboku Gijutsu Shiryou*[Research Data in Civil Engineering]. 4(1) (1962): 41-51.

[Kiriy91] Kiriyama, Takashi, Tetsuo Tomiyama and Hiroyuki Yoshikawa. "Sekkei Taishou Model Tougou-Ka No Tameno Metamodel No

Kenkyuu." [Research on the Metamodel toward Integrated Design Object Model.] *Jour. of Japanese Society for Artificial Intelligence.* 6(3) (1991): 116-124..

[Klir91] Klir, George J. "Aspect of Uncertainty in Qualitative Systems Modeling." In ed. P. Fishwick and P. Luker. *Qualitative Simulation Modeling and Analysis.* New York: Springer-Verlag, 1991. 24-50.

[Lenat90] Lenat, Douglas B. et al. "CYC: Toward Progress with Common Sense." *Communications of the ACM.* 33(8) (1990): 30-49.

[Liu90] Liu, Z. and A.M. Farley. "Shifting Ontological Perspective in Reasoning about Physical Systems." *AAAI-90.* Menlo Park, California: AAAI, 1990. 395-400.

[Lloyd87] Lloyd, J.W. *Foundations of Logic Programming.* Second, Extended ed. Berlin: Springer-Verlag, 1987.

[Maeda90] Maeda, Yasunobu and Saburo Ikeda. "Toshi Suikei No Kankyou Risk Hyouka To Risk Taiwa Shien System." [A Risk Information System for Risk Assessment and Risk Communication in An Urban Water Network.] *Proc. of the 3rd Annual Conference, Society for Risk Analysis, Japan-Section.* Tsukuba: SRA-J, 1990. 40-45.

[Maeda91] Maeda, Yasunobu and Saburo Ikeda. "Ta-Ryouiki No Chishiki Ni Motoduku Taishou Model No Kouchiku." [Construction of the Object Model Based on Multiple Domain Knowledge.] *Proc. of the 5th Annual Conference of JSAI.* Tokyo: JSAI, 1991. 321-324.

[Maeda92] Maeda, Yasunobu and Saburo Ikeda. "Toshi Suikei Network No Taishou Model Wo Motta Risk Hyouka Shien Chishiki System."

[A Knowledge System for Supporting Risk Assessment in Urban Water Network with Its Object Model.] *Jour. of JSAI.* 7(2) (1992): 271-279.

[Maeda93] Maeda, Yasunobu and Saburo Ikeda. "Gakusaisei Wo Kouryo Shita Chishiki Base Model Niyoru Kankyou Risk Hyouka Shien System."[Environmental Risk Assessment Support System Based on Interdisciplinary Knowledge-Based Model.] *Environmental Science.* 6(4) (1993): 287-296.

[McCar69] McCarthy, J. and P. J. Hayes. "Some Philosophical Problems from the standpoint of artificial intelligence." *Machine Intelligence.* 4 (1969): 463-502.

[Miki93] Miki, Kiyokazu et al. "Kyouchou-Gata Chisiki Base System No Tameno Terminology." [Terminology for Cooperative Knowledge-Based System.] *Proc. of the 7th Annual Conference of JSAI.* Tokyo: JSAI, 1993. 291-294.

[Minsk75] Minsky, Marvin. "A Framework for Representing Knowledge." In ed. P.H. Winston. *The Psychology of Computer Vision.* New York: McGraw-Hill, 1975. 211-277.

[Mood74] Mood, Alexander M., Franklin A. Graybill, and Duane C. Boes. *Introduction to the Theory of Statistics*, Third Edition. Singapore: McGraw-Hill, 1974. 180-181.

[Motod93] Motoda, Hiroshi, Ri-ichiro Mizoguchi and Toyoaki Nishida. "Kaigi Houkoku: Chishiki No Kyouyuu To Sai-Riyou No Workshop Houkoku."[Conference Report: Workshop for Sharing and Reuse of Knowledge.] *Jour. of JSAI.* 8(5) (1993): 666-671.

[Nakas83] Nakashima, Hideyuki. "A Knowledge Representation System Prolog/KR." *Mathematical Engineering Technical Report.*[Dept. of Math. Eng. and Inst. Phys., University of Tokyo.] METR 83-5 (1983).

[NRC83] National Research Council. *Risk Assessment in the Federal Government: Managing the Process.* Washington,D.C.: National Academy P., 1983.

[Ramon92] Ramoni, Marco et al. "An Epistemological Framework for Medical Knowledge-Based Systems." *IEEE Transactions on Systems, Man, and Cybernetics.* 22(6) (1992): 1361-1375.

[Robert91] Robertson, D. et al. *Eco-Logic.* Cambridge, Massachusetts: MIT P., 1991.

[Rook74] Rook, J.J. "Formation of Haloform During Chlorination of Natural Water". *Water Treatment and Examination.* 23, Part 2 (1974): 234-243.

[Rubin81] Rubinstein, Reuven Y. *Simulation and the Monte Carlo Method.* New York: John Wiley, 1981.

[Rumba91] Rumbaugh, James et al. *Object-Oriented Modeling and Design.* Englewood Cliffs, New Jersey: Prentice Hall, 1991.

[Schei88] Scheifler, R., I. Gettys and R. Newman. *X Window System.* Bedford, Massachusetts: Digital P., 1988.

[Stall86] Stallman, Richard. *GNU Emacs Manual.* Fifth ed., Emacs Version 18 for Unix Users. Cambridge, Massachusetts: Free Software Foundation, 1986.

[Sugaw72] Sugawara, Masami. *Ryuushutsu Kaiseki Hou.*[Method for Run-off Analysis.] Tokyo: Kyoritsu, 1972.

[Sun92] Sun Mycrosystems. *Sun System User's Guide.* Mountain View, California: Sun Microsystems, 1992.

[Tanbo83] Tanbo, Norihito. *Suidou To Trihalomethane.*[Waterworks and Trihalomethanes.] Tokyo: Gihodo, 1983.

[Tokyo88a] Tokyo Metropolitan Government Environmental Conservation Bureau. *Showa 62 Nendo Kokyo You Suiiki No Suishitsu Sokutei Kekka.* [Quality of Public Water in 1987.] Tokyo Metropolitan Government. 1988.

[Tokyo88b] Tokyo Metropolitan Government Waterworks Bureau. *Jigyou Nenpo Showa 62 Nendo.*[Annual Report of the Water Service, 1987.] Tokyo Metropolitan Government. 1988.

[Tokyo88c] Tokyo Metropolitan Government Sewerage Bureau. *Tokyo To Gesuidou Jigyou Nenpo Showa 62 Nendo.* [Annual Report of the Sewerage Service in Tokyo, 1987.] Tokyo Metropolitan Government. 1988.

[Ueno85] Ueno, Haruki. *Chishiki Kougaku Nyuumon.*[Introduction to Knowledge Engineering.] Tokyo: Ohm-Sha, 1985. 160-187.

[Urano83] Urano, Kohei, Hiroshi Wada, and Takao Takemasa. "Empirical Rate Equation for Trihalomethane Formation with Chlorination of Humic Substances in Water." *Water Research.* 17(12) (1983): 1797-1802.

[WHO84] World Health Organization. "Health-Related Constituents." In *Guidelines for Drinking-Water Quality* . Vol.2 Health Criteria and Other Supporting Information. Geneva: WHO, 1984. 155-246.

[Whyte80] Whyte,A. I. Burton. *Environmental Risk Asessment*, SCOPE Report 15. New York: John Wiley, 1980.

[Winst84] Winston, Patrick Henry. *Artificial Intelligence.* Second ed. Reading, Massachusetts: Addison-Wesley, 1984. 163-208.

[Yamag92] Yamaguchi, Takahira et al. "Taishou Model To Koshou Model Ni Motoduku Chishiki Compiler II No Kouchiku To Hyouka." [Knowledge Compiler II Based on Domain Model and Failure Model.] *Jour. of JSAI.* 7(4) (1992): 663-674.

[Yokoi90] Yokoi, Toshio. *Nihongo No Jouhou Ka.*[Making Information of Japanese.] Tokyo: Kyoritsu, 1990.

[Zeigl84] Zeigler, Bernard P. *Multifacetted Modelling and Discrete Event Simulation.* London: Academic P., 1984.

Appendix A

Basic syntax of Common ESP

This appendix shows the basic syntax of the Common ESP language.

- Programs in Common ESP is defined as a set of class definitions.
- Basic part of class definition is described by the following form:

```
class "class name"
has "inheritance declaration"

"class slot definitions"
"class method definitions"

instance

"instance slot definitions"
"instance method definitions"

local

"local predicate definitions"

end.
```

  where

  - "class name" is the identifier of the class,
  - "inheritance declaration" declares super-classes of the class,
  - "class slot definitions" and "class method definitions" define slots and methods of the class as an *object,*

  - "instance slot definitions" and "instance method definitions" define slots and methods of instances of the class, and
  - "local predicate definitions" defines local predicates that are used by only methods and predicates defined in this class definition.

- "Inheritance declaration" is described as the following:

  nature "super-classes";

  where "super-classes" is the identifier of super-class or a sequence of the identifiers of super-classes of the class.

- Both "class slot definitions" and "instance slot definitions" are described by the same form:

  attribute $A_1, A_2, A_3, \ldots$;
  component $C_1, C_2, C_3, \ldots$;

  where A_i is a definition of an attribute slot that can be refered by *objects* of other classes, and C_i is a definition of a component slot that is accessed by only *objects* in the class. Both A_i and C_i are described by the following form:

  "slot name"
  or
  "slot name" ::= "initial value"

  where "slot name" is the identifier of the slot, and "initial value" is initial value of the slot that is set at the start-up time of this program.

- Both "class method definitions" and "instance method definitions" consist of a set of definition of methods based on Prolog-like syntax. Definition of a method is described as the following form:

  :"method name"$(Arg_1, Arg_2, Arg_3, \ldots)$:- "body";

  where "method name" is the functor to identify the method, Arg_i is the argument of the method, and "body" is a sequence of messages to call methods and predicate calls. The first argument Arg_1 represents the destination *object* to which the message to call the method is sent.

- "Local predicate definitions" also consist of a set of definition of clauses based on Prolog-like syntax. Definition of a clause is described as the following form:

  "predicate name"$(Arg_1, Arg_2, Arg_3, \ldots)$:- "body";

  where "predicate name" is the functor to identify the predicate, Arg_i is the argument of the predicate, and "body" is a sequence of messages to call methods and predicate calls.

- Arguments of methods and predicates of which the first character is a capital letter are variables.
- A class *object* that is identified by *ID* is represented as #*ID*.
- Contents put between '/*' and '*/', and ones appearing after '%' in lines are regarded as comments, namely, these are not involved in the program.

This explanation omits details of the syntax. See [AIR91] to know precise syntax of the Common ESP language.

Appendix B

Source code of the RARCOM

This appendix shows the source code of main parts of the RARCOM, which are described by using the Common ESP language. Contents of this appendix is as listed below:

- class `rarcom`
- a class of domain knowledge base: `waterWorks`
- class `simModel`
- classes related to the bridging rules
  - class `bridge`
  - a sub-class of `bridge`: `waterE_waterW`
- classes of model components
  - class `fplant`
  - class `sewage`
  - class `city`
  - class `basin`
  - class `riverSegment`
- classes `compo` and `conduit`
  - class `compo`
  - class `conduit`
- class `scenarioMaking`

B.1 Class rarcom

```
/*
        Class rarcom (part)
This class object works as an interface between users and the RARCOM.
This manages all objects involved in the RARCOM.
 */
class rarcom with_macro rarcomPred has
nature rarcomObject;

attribute                         % Slots
    comment :=
"This class object works as an interface between users and the RARCOM.
This manages all objects involved in the RARCOM.",
    objectDomain := "Environmental Risk Assessment",
    objectArea :=   "Tokyo Metropolitan Area",
    mapData := "/home/ecopolis/maeda/rarcom/map_data/",
    simulationModel := simModel,
                                  % IDs of objects involved in the system
    scenarioMakingTool := scenarioMaking,
    mapOfArea := map,
    database := storage,
%       ...
    (tools := L :-
            :create(#list_index, L),
            :add_at( L, sTool, statistics ),
            :add_at( L, nemacsTool, editor ),
            :add_at( L, snapshot, hardcopy ));
                                  % Methods
% Initializing routine
:init( Cl ) :-
    :fprintf( Cl, stdout, "Initializing The Model.\n", [] ),
    initializeModel( classObj( Cl!simulationModel ) ),
    :fprintf( Cl, stdout, "Initializing The Map\n", [] ),
    initializeMap( classObj( Cl!mapOfArea ) ),
    :fprintf( Cl, stdout, "Initialized.\n", [] ),!;

% Main procedure
:run( Cl ) :-
    checkTheModel( classObj( Cl!simulationModel ) ),
    checkTheMap( classObj( Cl!mapOfArea ) ),
    :run( classObj( Cl!scenarioMakingTool ) );

:exit( Cl ) :-
    :exit( classObj( Cl!scenarioMakingTool ) ),
    :exit( classObj( Cl!mapOfArea ) ),
    :close( Cl!stdout ),
    :close( Cl!stderr ),!;

:initializeModel( Cl ) :-
    :fprintf( Cl, stdout, "Initializing The Model.\n", [] ),
    initializeModel( classObj( Cl!simulationModel ) ),
    :fprintf( Cl, stdout, "Initialized.\n", [] ),!;

:checkTheModel( Cl ) :-
    checkTheModel( classObj( Cl!simulationModel ) );

:bridgingRules( Cl, Bridge ) :-
    bridge( BID ),
    :get_class_object( #library, BID, Bridge );
%      ...
local

initializeModel( Model ) :-
    not( :empty( Model!modelComponents ) ),!,
    :initializeModel( Model );
initializeModel( Model ):-
    :generateModel( Model ),!;

initializeMap( Map ) :-
    :initializeCoordinates( Map ),!;

% classes of Bridging Rules
bridge( admin_sewag );
bridge( waterE_waterW );
bridge( sewag_waterE );
bridge( admin_waterW );
bridge( hydro_waterW );
bridge( hydro_sewag );
%      ...
end.
```

B.2 A Domain Knowledge Base: waterWorks

This section shows a part of definition of a domain knowledge base waterWorks, which holds knowledge of waterworks engineering.

```
/*
class of Waterworks Engineering (part)
 */

class waterWorks has
nature domain;

instance

attribute                              % Slots for instances of the class.
    comment := "Class of waterworks engineering",
    reference := "",
    treatmentRate;                     % Rate of filtration.
%      ...
%                                        Methods.
:filtration( Ins, F1, F2 ) :-
    filtration( Ins, F1, F2 );

:disinfection( Ins, F1, F2 ) :-
    disinfection( Ins, F1, F2 );

:thm_generation( Ins, F1, F2 ) :-
    thm_generation( Ins, F1, F2 );

                                       % Terminology interface.
:dictionary( Ins, Name, Unit, Explanation, FromNames, FromList, Data, Pred ):-
    dictionary( Name, Unit, Explanation, FromNames, FromList, Data, Pred );
                                       % Aliases.
:alias( Ins, Alias, Name ) :-
    alias( Alias, Name );
                                       % Water quality indices used in the domain.
:waterQuality( Ins, Attr ) :-
    waterQuality( Attr );
%      ...

local                                  % Local predicates.

% filtration( Instance of fplant, Input, Output )
%    A model of filtration process.
filtration( Ins, F1, F2 ) :-
    :getAttr( Ins, F1, COD, 'CODL' ),
    :getSlot( Ins, treatmentRate, R ),
    :setAttr( Ins, F1, COD*R, 'CODL', F2 );

% disinfection( Instance, Input, Output )
%    A model of disinfection process.
disinfection( Ins, F1, F2 ) :-
    chlorination( Ins, F1, F2 );

% thm_generation( Instance, Input, Output )
%     A model of trihalomethane generation.
thm_generation( Ins, F1, F2 ) :-
    :getAttr( Ins, F1, Humin, humin ),
    !,
    :getSlot( Ins, fe_Cl, Cl ),
    :'THM_from_humin'( #waterEnvEngineering, Humin, Cl, THM ),
    :setAttr( Ins, F1, THM, 'THM', F2 );
thm_generation( Ins, F1, F2 ) :-
    :getAttr( Ins, F1, TOC, 'TOC' ),
    !,
    :getSlot( Ins, fe_Cl, Cl ),
    :getSlot( Ins, pH, PH ),
    :getSlot( Ins, tmp, TMP ),
    :getSlot( Ins, period, T ),
    :'Urano_Takemasa'( #waterEnvEngineering, TOC, Cl, PH, TMP, T, THM ),
    :setAttr( Ins, F1, THM, 'THM', F2 );
thm_generation( Ins, F1, F2 ) :-
    :getAttr( Ins, F1, COD, 'COD-Mn' ),
    :bridge( #waterE_waterW, #waterEnvEngineering,
     F1, 'COD-Mn', COD, 'TOC', TOC ),
    !,
    :getSlot( Ins, fe_Cl, Cl ),
    :getSlot( Ins, pH, PH ),
    :getSlot( Ins, tmp, TMP ),
    :getSlot( Ins, period, T ),
    :'Urano_Takemasa'( #waterEnvEngineering, TOC, Cl, PH, TMP, T, THM ),
    :setAttr( Ins, F1, THM, 'THM', F2 );

% dictionary( Name, Unit, Explanation, FromNames, FromList, Data, Pred )
%    If value of attribute 'Name' is determined by other attributes,
%    'Pred' represents form determinig the value, 'FromNames' does
%    list of attributes used in the form, 'FromList' does list of variables
%    corresponding to the attributes, and 'Data' does the obtained value.
dictionary( flow, "10e3 m^3/day", "rate of flow", [], [], F, F );
dictionary( 'COD-Mn', "mg/l", "potassium permanganate consumption",
    [flow, 'CODL'], [Flow, CODL], COD, divide( CODL, Flow, COD ) );
```

```
dictionary( 'CODL', "kg/day", "potassium permanganate consumption load per day",
    [flow, 'COD-Mn'], [Flow, COD], CODL, multiply( COD, Flow, CODL) );
dictionary( treatmentRate, "", "average filtration rate", [], [], Data, Data );
%   ...

% alias( Alias, Name )
alias( organic, 'COD-Mn' );

% waterQuality( Name )
waterQuality( 'CODL' );
%   ...
end.
```

B.3 Class simModel

```
/*
        class simModel (part)

 */
class simModel has
nature rarcomObject;

attribute
    comment :=
"This class object involves knowledge representing:
        (1) relation between components, and
        (2) methods for simulation.",
                % Database for model components.
    (modelComponents := DB :- :create(#list_index, DB ));

                % Methods.
%     sim( Cl )
%                 Simulating exposure situation, Evaluating carcinogenicity,
%                 and storing consequences of simulation and evaluation in
%                 the knowledge base.
:sim( Cl ) :-
    :clear( #conduit ),
    sim( all ),
    evalRisk( allCities ),
    store( all ),
    Cl!simStatus := done,!;

:sim( Cl, Where ) :-
    :clear( #conduit ),
    :newTable( #storage, N ),
    sim( Where ),
    evalRisk( Where ),
    store( all ),
    Cl!simStatus := done,!;

:modelComponent( Cl, CompoID, Class, Obj ) :-
    modelComponent( CompoID, Class, Obj );

:componentID( Cl, ID, Obj ) :-
    componentID( ID, Obj );

:objectAttrs( Cl, Id, Attrs ) :-
    objectAttrs( Id, Attrs );
```

```
:connect(Cl, Upper, Lower ) :-
    connect( Upper, Lower );

:flowRate( Cl, F1, F2, F1overF2 ) :-
    flowRate( F1, F2, F1overF2 );

:fixedFlow( Cl, {U,D}, Flow ) :-
    fixedFlow( {U,D}, Flow );

:confluenceRate(Cl,Ux,D,Rate) :-
    confluenceRate(Ux,D,Rate);

:rel( Cl, Relation ) :- rel( Relation );

local

% procedures
sim( all ) :-
    repeat,
    (
    modelComponent( Component, Class, Obj ),
    :simCompo( Obj, Out ),
    fail;
    true );
sim( allCities ) :-
    repeat,
    (
    modelComponent( Component, city, Obj ),
    :simCompo( Obj, Out ),
    fail;
    true );
sim( Where ) :-
    componentID( Where, Obj ),
    :simCompo( Obj, OWhere ),!;

evalRisk( all ) :-
    repeat,
    (
    modelComponent( Component, Class, Obj ),
    :evalRisk( #riskEvaluator, Obj, Carcinogenicity ),
    fail;
    true );
evalRisk( allCities ) :-
    repeat,
    (
    modelComponent( Component, city, Obj ),
    :evalRisk( #riskEvaluator, Obj, Carcinogenicity ),
    fail;
    true );
evalRisk( Where ) :-
    :evalRisk( #riskEvaluator, Where, Carcinogenicity ),!;

store( all ) :-
    repeat,
    (
    modelComponent( Component, city, Obj ),
    :store( Obj ),
    fail;
    true );

modelComponent( CompoID, Class, Obj ) :-
    modelComponent( CompoID, Class ),
    componentID( CompoID, Obj );

componentID( ID, Obj ) :-
    objectID( #simModel!modelComponents, ID, Obj ),!;

objectID( DB, ID, Obj ) :-
    :get_at( DB, Obj, ID ),!;
objectID( DB, ID, Obj ) :-
    :get_contents_with_key( DB, DBList ),
    :member( #utility, {Obj,ID}, DBList ),!;

setBounderyConditions :-
    repeat,
    ( bounderyCondition( Id, F, BC ),
      componentID( Id, Obj ),
      :setConditions( Obj, F, BC ),
      fail ;
      true ),
    !;

initializeParameters :-
    repeat,
    ( objectAttrs( Id, AttrList ),
      componentID( Id, Obj ),
      :setSlot( Obj, status, yet ),
      :setObjectAttrs( Obj, AttrList ),
      fail ;
      true ),
    !;

% Component Database
%
modelComponent( ar0, riverSegment );     % Arakawa
modelComponent( ar1, riverSegment );
modelComponent( ar2, riverSegment );
%   ...
modelComponent( b_ar1, basin );          % Arakawa
modelComponent( b_ar2, basin );
modelComponent( b_ar3, basin );
%   ...
modelComponent( asaka_f, fplant );
modelComponent( misono_f, fplant );
%   ...
modelComponent( kosuge_s, sewage );
modelComponent( kitatama1_s, sewage );
modelComponent( tamagawaj_s, sewage );
%
modelComponent( chiyoda, city );
modelComponent( ohta, city );
modelComponent( tama, city );
%   ...
%
% Object Attributes
% objectAttrs( Id, Attrs );
%
objectAttrs( ar0, [{name, "Arakawa-0"}, {status, done}] );
objectAttrs( ar1, [{name, "Arakawa-1"}, {number, 326}, {length, 8.0}] );
objectAttrs( ar2, [{name, "Arakawa-2"}, {number, 327}, {length, 24.1}] );
objectAttrs( b_ar1,
     [{name, "Arakawa-1 Basin"}, {treatrate, 0.423},
      {population, 65549}] );
objectAttrs( b_ar2,
     [{name, "Arakawa-2 Basin"}, {treatrate, 0.758},
      {population, 374473}] );
objectAttrs( asaka_f,
[{name, "Asaka Filtration Plant"}, {number, 101},
{hasFixedFlow, yes}, {fixedFlow, 1091.9},
{fe_Cl, 1.1}, {tmp, 16.2}, {pH, 7.0}, {period, 7.4}] );
objectAttrs( chiyoda, [{name, "Chiyoda"}, {number, 1},
       {population, 53349}, {area, 11.52}] );
%   ...

% connect( Upper, Lower )
```

```
connect( ar0, ar1 );
connect( ar1, ar2 );
connect( ar1, asaka_f );
connect( ar1, misono_f );
connect( asaka_f, chiyoda );
connect( asaka_f, choufu );
connect( asaka_f, ohta );
connect( b_ar1, ar1 );
connect( b_ar2, ar2 );
connect( kosuge_s, ar3 );
connect( misono_f, itabashi );
connect( misono_f, kita );
%   ...

% fixedFlow( {U,D}, Flow );
fixedFlow( {kosuge_s, ar3}, 107.0 );
fixedFlow( {kosuge_s, ay3}, 62.1 );
%   ...
fixedFlow( {U,D}, Flow ) :-
    componentID( D, Obj ),
    :getSlot( Obj, hasFixedFlow, yes ),!,
    :getSlot( Obj, fixedFlow, Flow );

% boundery Conditions for simulations
bounderyCondition( ed0, dFlux, [{5166.7,flow}, {2.600,'BOD'}] );
bounderyCondition( nk0, dFlux, [{2586.7,flow}, {5.700,'BOD'}] );
bounderyCondition( ay0, dFlux, [{593.1,flow}, {26.00,'BOD'}] );
%   ...

% confluenceRate(Upper,Confluent point,Rate);
confluenceRate( asaka_f, chiyoda, 0.8 );
confluenceRate( asaka_f, ohta, 0.8 );
confluenceRate( asaka_f, setagaya, 0.2 );
confluenceRate( asaka_f, shibuya, 0.2 );
confluenceRate( asaka_f, suginami, 0.2 );
confluenceRate( asaka_f, choufu, 1.0 );
confluenceRate( asaka_f, tama, 1.0 );
confluenceRate( misono_f, kita, 0.5 );
confluenceRate( misono_f, itabashi, 0.8 );
%   ...

% flowRate( F1, F2, F1overF2 )
flowRate( {nk3, nk4}, {nk3, sn1}, 1.0 );
%   ...
% rel( Relation )
rel( riverSystem( 'Edogawa', [ed0, ed1, ed2, ed3, ed4, ed5] ) );
rel( riverSystem( 'Arakawa', [ar0, ar1, ar2, ar3, ar4] ) );
rel( riverSystem( 'Tamagawa', [tm0,tm1,tm2,tm3,tm4,tm5,tm6,tm7,tm8,tm9] ) );
%   ...
end.
```

B.4 Classes related to the bridging rules

This section shows a part of the class `bridge` and the class `waterE_waterW`, which is a sub-class of `bridge`.

B.4.1 Class bridge

```
/*
        class bridge (part): knowledge base of bridging rules.

 */
class bridge has
nature rarcomObject;

attribute                                    % Slots.
    commnets :=
"This class object manages all bridging rules used in the system.
When this object receives a message to call a bridging rule,
it selects a sub-class of it that actually holds an appropriate
bridging rule, and requests the sub-class to bridge.",

    domain1, domain2;                        % ID of domain knowledge bases.

                                             % Methods.
:bridgeGap( Cl, Up, Down, UFlux, Solved ) :- % Bridging a flow data.
    solveAttr( Up, Down, UFlux, UFlux, Solved );

                                           % Bridging by interpretation rules.
:bridge( Cl, Destination, Flux, Attr, Data, A2, D2 ) :-
    bridge( Cl, Destination, Flux, Attr, Data, A2, D2 );

:bridgeName(Cl, Obj1, Obj2, Bridge) :-     % Specifying a sub-class of
    bridgeName(Cl, Obj1, Obj2, Bridge);    % the class bridge
%   ...

local

solveAttr( Up, Down, UFlux, [{Data,Attr} |Attrs], [{Data,Attr} |Solved] ) :-
    :dictionary( Down, Attr, _,_,_,_,_ ), !,
    solveAttr( Up, Down, UFlux, Attrs, Solved );
solveAttr( Up, Down, UFlux, [{Data,Attr} |Attrs], [{Data2,Attr2} |Solved] ) :-
    :domainName( #domain, Up, UDom ),            % Using
    :domainName( #domain, Down, DDom ),          % interpretation rules.
    bridgeOverDomains( #bridge, UDom, DDom, BNam ),
    bridge( BNam, DDom, UFlux, Attr, Data, Attr2, Data2 ),!,
    solveAttr( Up, Down, UFlux, Attrs, Solved );
solveAttr( Up, Down, UFlux, [{Data,Attr} |Attrs], [{Data,Attr} |Solved] ) :-
    solveAttr( Up, Down, UFlux, Attrs, Solved ),!;     % Default rule.
solveAttr( Up, Down, UFlux, [], [] );

bridgeName(Cl, Obj1, Obj2, Bridge) :-                % Searching
    domain_class( Obj1, Dom1 ),                      % a sub-class of bridge.
    domain_class( Obj2, Dom2 ),
    bridgeOverDomains(Cl, Dom1, Dom2, Bridge);

bridgeOverDomains(Cl, Dom1, Dom2, Bridge) :-
    ( :is_class( Dom1 ), Obj1 = Dom1 ;               % Searching
      :get_class_object( #library, Dom1, Obj1 ) ), % a sub-class of bridge
    ( :is_class( Dom2 ), Obj2 = Dom2 ;               % from given two domain
      :get_class_object( #library, Dom2, Obj2 ) ), % objects.
    :bridgingRules( #rarcom, Bridge ),
    bridgeOfDomains( Bridge, Obj1, Obj2 );

domain_class( Obj, Dom ) :-                    % If Obj is an object,
    object( Obj, Size, ObjDesc),!,
    :domainName( Obj, Dom );
domain_class( CID, Dom ) :-                    % else if CID is a component ID,
    :componentID( #simModel, ID, Obj ),!,
    :domainName( Obj, Dom );
domain_class( DID, Dom ) :-                    % else if DID is a class ID,,,
    :get_class_object( #library, DID, Cls ),!,
    :domainName( Cls, Dom );

bridgeOfDomains( Bridge, Dom1, Dom2 ) :-
    :getSlot( Bridge, domain1, DID1 ),
    :class_name( Dom1, DID1 ),
    :getSlot( Bridge, domain2, DID2 ),
    :class_name( Dom2, DID2 );
bridgeOfDomains( Bridge, Dom1, Dom2 ) :-
    :getSlot( Bridge, domain2, DID1 ),
    :class_name( Dom1, DID1 ),
    :getSlot( Bridge, domain1, DID2 ),
    :class_name( Dom2, DID2 );
%   ...

end.
```

B.4.2 Class waterE_waterW

```
%
%     water environment engineering -- waterworks engineering
%
```

```
class waterE_waterW has
nature bridge;

attribute
    comment :=
"Class object of bridging between
 water environment engineering and waterworks engineering",

    reference :=
"Tokyo Metropolitan Government Environmental Conservation Bureau.
   {Showa 62 Nendo Kokyo You Suiiki No Suishitsu Sokutei Kekka.}
   [Quality of Public Water in 1987.] Tokyo Metropolitan Government. 1988.
Tokyo Metropolitan Government Waterworks Bureau.
   {Jigyou Nenpo Showa 62 Nendo.}
   [Annual Report of the Water Service, 1987.] Tokyo Metropolitan Government.
   1988.",

    domain1 := waterEnvEngineering,
    domain2 := waterWorks;

:bridge( C1, Destination, Flux, Attr, Data, A2, D2 ) :-
    bridge( Destination, Flux, Attr, Data, A2, D2 );

local

bridge( #waterWorks, Flux, 'BODL', Data, 'CODL', D2 ) :-
    :getAttr( #waterEnvEngineering, Flux, BOD, 'BOD' ),
    bridge( #waterWorks, Flux, 'BOD', BOD, 'COD-Mn', COD ),
    :member( #utility, {F, flow}, Flux ),
    D2 is COD * F;

bridge( #waterWorks, Flux, 'BOD', Data, 'COD-Mn', COD ) :-
    COD is 0.887 * Data + 5.073;

bridge( #waterEnvEngineering, Flux, 'CODL', Data, 'BODL', D2 ) :-
    :getAttr( #waterWorks, Flux, COD, 'COD-Mn' ),
    bridge( #waterEnvEngineering, Flux, 'COD-Mn', COD, 'BOD', BOD ),
    :member( #utility, {F, flow}, Flux ),
    D2 is BOD * F;

bridge( #waterEnvEngineering, Flux, 'COD-Mn', Data, 'BOD', BOD ) :-
    BOD0 is 0.810 * Data - 1.794,
    ( BOD0 =< 0.0, BOD = 0.0 ; BOD = BOD0 );

bridge( #waterEnvEngineering, Flux, 'CODL', Data, 'TOCL', D2 ) :-
```

```
    :getAttr( #waterWorks, Flux, COD, 'COD-Mn' ),
    bridge( #waterEnvEngineering, Flux, 'COD-Mn', COD, 'TOC', TOC ),
    :member( #utility, {F, flow}, Flux ),
    D2 is TOC * F;

bridge( #waterEnvEngineering, Flux, 'COD-Mn', Data, 'TOC', TOC ) :-
    TOC is 0.545 * Data + 1.227;

end.
```

B.5 Model components

This section shows five sub-classes of **compo**, that is, **fplant**, **sewage**, **city**, **basin**, and **riverSegment**, which define behavior of components of the urban water network model.

B.5.1 Class fplant

```
/*
 * Class of Filtration Plant (part)
 */

class fplant has
nature compo, waterWorks;

instance
attribute                    % Slots for instances.
    comment := "Model of filtration plants.",
    reference :=
"Tokyo Metropolitan Government Waterworks Bureau.
   {Jigyou Nenpo Showa 62 Nendo.}[Annual Report of the Water Service, 1987.]
    Tokyo Metropolitan Government. 1988.",

    fe_Cl   := 0,           %    free efficient chlorine (mg/l)
    pH      := 7.0,         %    pH
    tmp     := 15.0,        %    temperature (degs Celsius)
    period  := 0,           %    filtration period (hour)
    treatmentRate := 0.5; %    treatment rate for organic matters(BOD)
%   ...

:process(Ins) :-             % Methods.
    getFlux( Ins, uFlux, U ),
    process(Ins,U,D),
    setFlux( Ins, dFlux, D );
%   ...

local                        % Local predicates.

process( Ins, U, D2 ) :-
    :filtration( Ins, U, D0 ),
    :disinfection( Ins, D0, D1 ),
    :thm_generation( Ins, D1, D2 );
%   ...

end.
```

B.5.2 Class sewage

```
/*
 * Class of Sewage Works (part)
 */
class sewage with_macro rarcomPred has
nature compo, sewageWorks;

instance
attribute
    comment := "Model of sewage treatment plants.",
    reference :=
"Tokyo Metropolitan Government Sewerage Bureau.
   {Tokyo To Gesuidou Jigyou Nenpo Showa 62 Nendo.}
   [Annual Report of the Sewerage Service in Tokyo, 1987.]
   Tokyo Metropolitan Government. 1988.",

    maxflow := 0.0,        % maximum flow (10e3m^3/day)
    maxload := 0.0,        % maximum load (BOD)(kg/day)
    treatmentRate := 0.0,  % mean treatment rate
    cities;              % treatment cities

:process(Ins) :-
    getFlux( Ins, uFlux, U ),
    process(Ins,U,D),
    setFlux( Ins, dFlux, D );
                                    % Data saved in the storage object.
:savingData( Ins, Attr ) :-
    savingData( Attr );
%       ...

local

process( Ins, U, D ) :-
    :getAttr( Ins, U, BODL, 'BODL' ),
    :getAttr( Ins, U, Q, flow ),
    :getSlot( Ins, maxflow, MF ),
    :checkOverflow( Ins, MF, BODL, Q, RealLoad, OverLoad ),
    :getSlot( Ins, maxload, ML ),
    :getSlot( Ins, treatmentRate, MTR ),
    :treatment( Ins, ML, MTR, RealLoad, Treated ),
    :setAttr( Ins, U, Treated+OverLoad, 'BODL', D );

% Data saved in the storage.
savingData( flow );
savingData( 'BOD' );
savingData( 'THM' );

end.
```

B.5.3 Class city

```
/*
 * Class of cities, towns and wards (part)
 */
class city with_macro rarcomPred has
nature compo, administration;

instance

attribute
    comment := "Model of cities, towns and wards.",
    reference :=
"Tokyo Metropolitan Government Waterworks Bureau.
   {Jigyou Nenpo Showa 62 Nendo.}[Annual Report of the Water Service, 1987.]
    Tokyo Metropolitan Government. 1988.",

    area,                           % unit: Km^2
    population;

:process(Ins) :-
    getFlux( Ins, uFlux, U ),
    process(Ins,U,D),
    setFlux( Ins, dFlux, D );
                                    % Data saved in the storage object.
:savingData( Ins, Attr ) :-
    savingData( Attr );
%       ...

local

process( Ins, U, U );

% Data saved in the storage.
savingData( flow );
savingData( 'COD-Mn' );
savingData( 'THM' );
%      ...

end.
```

B.5.4 Class basin

```
/*
 * Class of Basin (part)
 */
class basin with_macro rarcomPred has
nature compo, hydrology, waterEnvEngineering;

instance
attribute
    comment := "Model of river basins."
    reference :=
"This data is read from 1:50,000 Scale Topographic Maps published by
The Geographical Survey Institute, Ministry of Construction, Japan.",

    intake := 0.0,    % Intake quantity(10e3m^3/day)
    domesticLoad := 0.0,   % Domestic load(kg/day)
    treatrate := 0.0;    % Treatment rate

:setSlot( Ins, population, Data ) :-
    set_slot( Ins, population, Data ),
    integer_to_floating_point( Data, DF ),
    :getSlot( Ins, treatrate, R ),
    :getSlot( Ins, unitload , UL ),
    :getSlot( Ins, runoff   , ROFF ),
    L is DF * ( 1.0 - R ) * UL * ROFF,
    set_slot( Ins, domesticLoad, L );
:setSlot( Ins, treatrate, Data ) :-
    set_slot( Ins, treatrate, Data ),
    :getSlot( Ins, population, P ),
    integer_to_floating_point( P, PF ),
    :getSlot( Ins, unitload  , UL ),
    :getSlot( Ins, runoff    , ROFF ),
    L is PF * ( 1.0 - Data ) * UL * ROFF,
    set_slot( Ins, domesticLoad, L );

:process(Ins) :-
    getFlux( Ins, uFlux, U ),
    process(Ins,U,D),
    setFlux( Ins, dFlux, D );
                                  % Data saved in the storage object.
:savingData( Ins, Attr ) :-
    savingData( Attr );
%       ...
local
```

```
process( Ins, U, D ) :-
    :getSlot( Ins, domesticLoad, DL ),
    addBodLoad( Ins, U, DL, D0 ),
    :getSlot( Ins, intake, Q1 ),
    :getAttr( Ins, D0, Q0, flow ),
    :setAttr( Ins, D0, Q0-Q1, flow, D );

addBodLoad( Ins, U, DL, D ):-
    :getAttr( Ins, U, BODL, 'BODL' ),
    :setAttr( Ins, U, BODL+DL, 'BODL', D );
addBodLoad( Ins, U, DL, D ):-
    :setAttr( Ins, U, DL, 'BODL', D );

savingData( flow );
savingData( 'BODL' );
%savingData( 'THM' );

end.
```

B.5.5 Class riverSegment

```
/*
    Class of segments of river channels.
 */
class riverSegment with_macro rarcomPred has
nature compo, hydrology, waterEnvEngineering;

instance

attribute
    comment := "Model of segments of river channels.",
    reference :=
"Hall, Warren A. and John A. Dracup.
   {Water Resources Systems Engineering.}
   New York: McGraw-Hill, 1970. 341-356.
Tokyo Metropolitan Government Environmental Conservation Bureau.
   {Showa 62 Nendo Kokyo You Suiiki No Suishitsu Sokutei Kekka.}
   [Quality of Public Water in 1987.] Tokyo Metropolitan Government. 1988.";

:process(Ins) :-
    getFlux( Ins, uFlux, U ),
    process(Ins,U,D),
    setFlux( Ins, dFlux, D );
                                        % Data saved in the storage object
:savingData( Ins, Attr ) :-
```

```
    savingData( Attr );
%       ...
local

process( Ins, U, D ):-
    :getSlot( Ins, length, L ),
    :selfPurification( Ins, L, U, D );

% Data saved in the storage
savingData( flow );
savingData( 'BOD' );
savingData( 'THM' );
%       ...
end.
```

B.6 Classes compo and conduit

B.6.1 Class compo

```
%
%     Class of component
%
class compo with_macro rarcomPred has
nature simObject;

instance
attribute
    comment :=
"Class of component.
This class defines the method simCompo,
the simulation procedure based on the constraint solving algorithm.",
    dataSource,             % Data source(Name of the observaiotn point)
    hasFixedFlow := no,     % If the component has a fixed flow,
                            % it becomes 'yes'.
    fixedFlow,              % Fixed rate of flow.
    number,                 % Serial number
    carcinogenicity := 0.0,       % Estimated carcinogenicity
    status := yet;

:simCompo( Ins, Out ) :-
                        % This method determines the value of OUT,
                        % which represents the stuff at the downward terminal.
    simCompo( Ins, Out );

:setConditions( Ins, Flux, BC ) :-
    setConditions( Ins, Flux, BC ),!;
%      ...
local

simCompo( Ins, Out ) :-
    :getSlot( Ins, status, yet ),
    !,
    :getSlot( Ins, upward, Up ),
    simUpward( Ins, Up, In ),
    :process( Ins, In, Out ),
    :setSlot( Ins, status, done  ),!;
simCompo( Ins, Out ) :-
    :getSlot( Ins, status, done ),
    !,
    getFlux( Ins, dFlux, Out ),!;
```

```
simUpward( Ins, null, Input ) :-
    getFlux( Ins, uFlux, Input );
simUpward( Ins, List, Input ) :-
    simConduitList( List ),
    :confluence( #conduit, List, Input );

simConduitList( [Conduit | Rest] ) :-
    :simConduit( Conduit ),
    simConduitList( Rest );
simConduitList( [] );
%       ...
end.
```

B.6.2 Class conduit

```
%
%     Class conduit
%  An instance of conduit connects a component with another component.
%

class conduit with_macro rarcomPred has
nature simObject;

attribute
    comment :=
"Class of conduit.
An instance of conduit connects a component with another component.";

component
    fluxDB := DB :- :create( #list_index, DB );

:confluence( Cl, List, Out ) :-
    confluence( List, Out );
:divide(Cl, In, OutList ) :-
    divideFlow( In, OutList );

:makeInstance( Cl, {Up, Dw}, Ins ) :-
    :new( Cl, Ins ),
    :setSlot( Ins, id, {Up, Dw} ),
    :componentID( #simModel, Up, UIns ),
    :componentID( #simModel, Dw, DIns ),
    :setSlot( Ins, upward,   UIns ),
    :setSlot( Ins, downward, DIns ),!;

instance
```

```
attribute
    comment :=
"Class of conduit.
An instance of conduit connects a component with another component.",
    status := yet;

:solve( Ins, Up, Dw, UFlux ) :-
    solve( Ins, Up, Dw, UFlux );

% simConduit
% It is called by the method simCompo in the class 'compo'.
:simConduit( Ins ) :-
    simConduit( Ins );

local

/* Predicate solve
It translates the given UFlux by calling a bridging rule,
and sets the translated value in the slot dFlux */
solve( Self, Up, Down, UFlux ) :-
    :class_object( Up,   Cls ),
    :class_object( Down, Cls ), !,
    setFlux( Self, dFlux, UFlux );
solve( Self, Up, Down, UFlux ) :-
    :bridgeGap( #bridge, Up, Down, UFlux, Solved ),
    setFlux( Self, dFlux, Solved ),!;

confluence( [Conduit |Rest], Out ) :-
    addConfluence( Conduit, Rest, Out );

addConfluence( Original, [Add | Rest], Out ) :-
    addConfluence( Original, Rest, Sum ),
    getFlux( Add, dFlux, AddFlux ),
    addFlux( Sum, AddFlux, Out ),!;
addConfluence( Original, [], Sum ) :-
    getFlux( Original, dFlux, Sum ),!;

addFlux( [{Data,Key} | Rest], Add, [{Sum,Key} |Result] ) :-
    :delete(#utility, {AddData,Key}, Add, AddRest ),
    !,
    Sum is Data + AddData,
    addFlux( Rest, AddRest, Result );
addFlux( [{Data,Key} | Rest], Add, [{Data,Key} |Result] ) :-
    addFlux( Rest, Add, Result );
addFlux( [], [A | R], Result ) :-
```

```
    addFlux( [A | R], [], Result );
addFlux( [], [], [] );

simConduit( Ins ) :-
    :getSlot( Ins, status, yet ),
    !,
    :getSlot( Ins, upward, UPWARD ),
    :getSlot( Ins, downward, DOWNWARD ),
    :simCompo( UPWARD, Flux ),
    :getSlot( Ins, id, ID ),
                % Computing the value to be stored in the slot uFlux.
    upwardFlux( ID, UPWARD, Flux, UFlux ),
    setFlux( Ins, uFlux, UFlux ),
    solve( Ins, UPWARD, DOWNWARD, UFlux ),!;
simConduit( Ins ) :-
    :getSlot( Ins, status, done ),
    !;

% Computing the value to be stored in the slot uFlux.
upwardFlux( {UP,DW}, UPWARD, Flux, UFlux ) :-
    divideFlow( UPWARD, DividedList ),
    :member( #utility, {DW, UFlux}, DividedList ),!;

% There may be more than one conduits connecting the upward component.
% This predicate divides the output from the upward component to
% these conduits.
divideFlow( Upward, List ) :-
    :getSlot( Upward, id , UName ),
    getFlux( Upward, dFlux, Flux ),
    :setof( #sets, X, connect(#simModel,UName,X), NameList ),
    divideListedFlow( Flux, UName, NameList, List );

divideListedFlow( Flux, Up, [], [] );
divideListedFlow( Flux, Up, [Name], [{Name,Flux}] );
            % If there are conduits having a fixed rate of flow, then...,
divideListedFlow( Flux, Up, Names, Result ):-
    haveFixedFlow( Up, Names, WeHave ),
    WeHave =\= [],
    !,
    calcFixedFlow( Flux, Names, WeHave, FixedFluxList, RestName, RestFlux ),
    divideListedFlow( RestFlux, Up, RestName, List ),
    :append( #utility, FixedFluxList, List, Result );
            % or else, the constraint between the conduits must be solved.
divideListedFlow_body( Flux, Up, Names, Result ):-
    haveEquation( Up, Names, Eqs ),
    Eqs =\= [],
    !,
    solveEquation( Flux, Names, Eqs, Result );

solveEquation( Flux, Names, Eqs, Result ) :-
    getAttr( Flux, Full, flow ),
                      % Generating a set of linear simultaneous equations
                      % from description of the relation between the conduits
                      % that is stored in the object simModel.
    generateVarEqs( Full, Names, Eqs, VarNames, VarEqs ),
                      % Calling the solver for linear simultaneous equations.
                      % It returns a set of forms to obtain solutions.
    :solve( #cSolver, VarNames, VarEqs, R ),
    weHave( Names, R, Forms ),
                      % Computing the returned froms.
    calcListedFlow( Flux, Forms, Result, Rest );
%      ...
end.
```

B.7 Class scenarioMaking

```
/*
        class scenarioMaking

 */

class scenarioMaking with_macro rarcomPred has
nature tool;

attribute
    comment :=
"This object makes scenarios on the basis of scenario templates
stored in the object, and execute simulations by sending the scenarios
to the object simModel. Procedures to execute simulations are as follows:
1) Displaying the dialogbox,
2) Let users to input the scenario setting on the dialogbox,
3) Based on the inputted data, sending to the object simModel
   messages to set the condition for the simulation, and,
4) Sending to the object simModel the message to execute the simulation.";

component
    dialogbox   := null,      % Main dialogbox.
    subbox      := null,      % List of sub-dialogboxes.
    item_number := 1;

:run( Cl ) :- run( Cl );
:exit( Cl ) :- exit( Cl );
:deactivate( Cl ) :- deactivate( Cl );

local

run( Cl ) :-
    make_Dialogbox,
    listen_to_Dialogbox( Selection, Var_Switch, Map_Switch ),
    scenario( Selection, Var_Switch ),
    (:member( #utility, [message, MES], Selection ) ; MES = null ),
    :termString( #utility, MES, Name ),  % Extracting the scenario name.
    :newTable( #storage, Name ),         % Initializing the storage.
    :sim( #simModel ),                   % Simulation.
    display_Map( Map_Switch );
run( Cl );

%   listen_to_Dialogbox( Selection, Var_Switch, Map_Switch )
%       Displaying the dialogbox and get the input data.
%       Selection:  Selected scenario by the user
%       Var_Switch: Dummy
%       Map_Switch: Flag to determine displaying the object map or not
listen_to_Dialogbox( Selection, Var_Switch, Map_Switch ):- !,
    #scenarioMaking!dialogbox = Main,
    :activate( Main ),
    scenario_Template( TLIST ),
    read_Dialog_Selection( Main, TLIST, Selection, Var_Switch, Map_Switch, Dialog ),
    :deactivate( #scenarioMaking ),!,
    Dialog \== abort;

%   read_Dialog_Selection( Main, TLIST, Selection, Var_Switch, Map_Switch, Dialog )
%       Getting the input data from the dialogbox
%       Main:       The main dialogbox
%       TLIST:      List of scenario templates
%       Selection:  Selected scenario by the user(an item of TLIST)
%       Var_Switch: Dummy
%       Map_Switch: Flag to determine displaying the object map or not
%       Dialog:     User's input: 'do_it' or 'abort'
read_Dialog_Selection(Main, TLIST, Selection, false, Map_Switch, do_it) :-
    :read( Main, INPUT ),
    :member( #utility, selection(ATOM_N), INPUT ),
    :get_atom_string( #symbolizer, ATOM_N, STR_N ),
    :get_number( #symbolizer, N, STR_N ),
    :nth( #utility, N, TLIST, Selection ),
    listen_to_SubDialog( Selection ),
    :member( #utility, map(Map_Switch), INPUT );
read_Dialog_Selection( Main, TLIST, Selection, Var_Switch, Map_Switch, abort );

%   scenario( Selection, Var_Switch )
%           Sending to the object simModel the message to set the simulation
%           condition according to the selected scenario.
%           Selection: Selected scenario by the user
%             form: [[name, 'name string'], [message, 'message to send'], ...]
%           Var_Switch: Dummy
scenario( Selection, Var_Switch ):-
    (:member( #utility, [message, MES], Selection ) ; MES = null ),
    :resetSimModel(#simModel),
    initializeScenario( MES ),
    :set_Var_Mode( #simModel, Var_Switch );

scenario_Template(
    [[[name, "1987 mean value"]],
     [[name, "Case of Summer"], [message, case_of_Summer]],
     [[name, "Case of Winter"], [message, case_of_Winter]],
     [[name, "Excess Input of Chlorine"],
       [message, ex_Chlor( Fplant, Cl )],
       [parameter, Fplant, Cl],
       [subbox, ex_Chlor],
       [param_attr, [[name, "Where(Filtration Plant)?"],
     [type, choose,
    ["Asaka FP",
     "Misono FP",
     "Higashimurayama FP",
     "Sakai FP",
     "Kinuta-Ue FP",
     "Kinuta-Shita FP",
     "Kanamachi FP",
     "Misato FP",
     "Kosaku FP"]],
     [id, fplant]],
                    [[name, "Chlorine (mg/l)"],
     [type, string],
     [id, 'Cl']]]],
      [[name, "Increase Contamination at Upward Area"],
       [message, inc_C_at_Upper( River, C )],
       [parameter, River, C],
       [subbox, inc_C_at_Upper],
       [param_attr, [[name, "Where(River Name)?"],
     [type, choose,
    ["Edogawa",
     "Ayasegawa",
     "Nakagawa",
     "Arakawa",
     "Shingashigawa",
     "Tamagawa"]],
     [id, river]],
                    [[name, "BOD (mg/l)"],
     [type, string],
     [id, c]]]]] );

initializeScenario( null );
initializeScenario( case_of_Summer ) :-
    e_setSlot( asaka_f, tmp, 28.2 ),
    e_setSlot( misono_f, tmp, 29.2 ),
    e_setSlot( higashimura_f, tmp, 28.1 ),
    e_setSlot( sakai_f, tmp, 24.6 ),
    e_setSlot( kinutakami_f, tmp, 25.6 ),
    e_setSlot( kinutashimo_f, tmp, 20.9 ),
    e_setSlot( kanamachi_f, tmp, 30.9 ),
    e_setSlot( misato_f, tmp, 30.2 ),
    e_setSlot( kosaku_f, tmp, 23.8 ),!;
initializeScenario( case_of_Winter ) :-
    e_setSlot( asaka_f, tmp, 5.5 ),
    e_setSlot( misono_f, tmp, 4.8 ),
    e_setSlot( higashimura_f, tmp, 5.1 ),
    e_setSlot( sakai_f, tmp, 5.3 ),
    e_setSlot( kinutaue_f, tmp, 11.4 ),
    e_setSlot( kinutashita_f, tmp, 14.0 ),
    e_setSlot( kanamachi_f, tmp, 5.8 ),
    e_setSlot( misato_f, tmp, 5.0 ),
    e_setSlot( kosaku_f, tmp, 5.1 ),!;
initializeScenario( ex_Chlor( FplantName1, Cl ) ) :-
    purify_string( FplantName1, PureString ),
    :objectAttrs( #simModel, Fplant, [{name,FplantName2} | R] ),
    purify_string( FplantName2, PureString ),
    :get_number( #symbolizer, Data, Cl ),
    e_setSlot( Fplant, fe_Cl, Data ),!;
initializeScenario( inc_C_at_Upper( RiverString, C ) ) :-
    ( string(RiverString,_,_),
      :get_atom( #symbolizer, River, RiverString );
      atom(RiverString), RiverString = River ),
    :rel( #simModel, riverSystem( River, [R0 | R] ) ),
    :get_number( #symbolizer, Data, C ),
    e_setConc(R0, 'BOD', Data ),!;
%       ...
end.
```