COMODEL:
A LANGUAGE FOR THE REPRESENTATION OF TECHNICAL KNOWLEDGE

Werner Dilger and Jorg Kippe

Fraunhofer-Institut flir Informations- und Datenverarbeitung
D-7500 Karlsruhe
FR Germany

ABSTRACT

The component oriented language COMODEL for the description of technical systems is presented. It represents technical knowledge by means of functions and geometric properties of components in a uniform way based on a small number of elementary concepts. Components together with interactions between components can be composed to aggregates, and a whole technical system is viewed as a big aggregate consisting of a hierarchy of subaggregates. Functions and geometric properties of aggregates can be derived from those of the subaggregates. Thus, a technical system can be modelled by a set of COMODEL expressions, and diagnosis and prognosis in the system may be performed by means of these expressions.

1. INTRODUCTION

The first and most well-known expert systems were developed for medical and scientific applications. Only since a few years technical applications came in sight. This type of application requires in first place new types of knowledge representation. Thus, a number of papers appeared on this topic following the ideas of the basic work of Hayes [1]. They deal mainly with the modelling of physical processes going on in technical systems, cf. [2, 3, 4,5,6,7,8,9,10]. All of them reason about physical processes making use of a qualitative description of physical entities. As possible applications are considered diagnosis and prognosis in technical systems and control of technical processes.

In our approach we stress attention to the system theoretical principles of knowledge modelling and on the representation of geometric knowledge. On the basis of some principles of the modelling of technical systems (section 2), we give an introduction to the knowledge representation language COMODEL (section 3). Its main feature is the component oriented description of technical systems and processes covering geometrical and process knowledge by the same means of description. Its application will be illustrated by some examples (section 4).

2. MODELLING OF TECHNICAL SYSTEMS

The different works on modelling of technical systems mentioned above have in common that they do not describe a technical system globally, rather they start from context independent descriptions of single parts and their functions and try to infer descriptions of aggregates from those of the parts. The descriptions are based on variables, e.g. temperature, pressure, cross-section etc., values of variables and constraints. Forbus [2, 3, 4] takes the process, causing changes of physical situations, as the basic concept of his description, in de Kleer's and Brown's approach [5, 6] the physical component, whose behavior is characterized by confluences, plays this role, and Kuipers' and Kassirer's work [7, 8, 9] is based on "universal" constraints, related to different types of variables. Raulefs [10] attaches importance to different degrees of granulation in the qualitative description and to reasoning about time.

Our approach has the following basic features:

1. Component oriented descripton. A technical system is regarded as composed of a finite number of components that correspond to the parts of a real system and can be accumulated to aggregates.
2. Qualitative and quantitative representation of the properties of physical objects.
3. System theoretical taxonomy for classifying the variety of components and their connections (component types, interaction types).
4. Uniform description of the function and the geometric properties of components.

With respect to 1. and 2., our approach is related to that of de Kleer and Brown and of Kuipers and Kassirer. Component orientation allows modular descriptions of large aggregates, therefore such a description can be easily modified and adapted to other configurations. The behaviour of components and aggregates is defined by means of constraints on physical "properties" or variables. With respect to 3. and 4., we differ from the works mentioned above. As a consequence of a component oriented approach, the components can and should be classified in a systematic way according to the principles of system engineering. Geometric properties of components are of interest in so far as they are essential parts of the behaviour of the components, and for this reason they are described in the same way as the functional properties.

3. ELEMENTS OP COMODEL

In the COMponent Oriented DEscription Language (COMODEL) a technical system is viewed as a set of phenomenons, that may be observable by automatic measurement or by human senses. Phenomenons can be connected by relations. This results in a network consisting of phenomenons as edges and two types of nodes, called components and interactions (cf. figure 1). Components, interactions and phenomenons are the basic objects of our description language, they are related to the parts of a technical system, the connections of these parts and the observable physical properties in the real world.

A COMODEL-description of a real technical system consists of a number of object-definitions, defining the three basic objects and a compound object called aggregate .

object-def <- (DEFOBJ identifier basic-object) *

basic-object* phenomenon I component I
interaction I aggregate

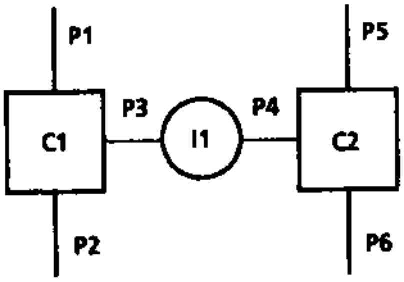

Fig. 1: A sample network.

A phenomenon is a collection of a finite number of physical properties, combined from a particular point of view.

phenomenon <- phenomenon-name I
phenomenon-descr

phenomenon-name <- identifier I PHENOMENON

phenomenon-descr <- (phenomenon-type
(phyprop...)) *

phenomenon-type <- phenomenon-name

A physical property is a part of a phenomenon. It may be observable or not.

phyprop <- (phyprop-name phyprop-type)

A component is an object with a finite number of gates and relations. To each gate a phenomenon is assigned. The gates serve as links between components and interactions. The relations describe connections between values of different physical properties in the gates' phenomenons or define values of physical properties.

component <- component-name I component-descr

component-name <- identifier I COMPONENT

component-descr <- (component-type (gate...)
(rel...))
component-type <- component-name

An interaction is an object with a finite number of gates and relations, like a component. It differs from a component by the fact that its relations cannot define values, i.e. interactions do not have characteristic or intrinsic values.

interaction <- interaction-name I
interaction-descr

interaction-name <- identifier I
INTERACTION

interaction-descr <- (interaction-type (gate...)
(rel...))

interaction-type <- interaction-name

---

* Preterminal symbols are underlined, there are no further definitions on account of clearness.

* Repetition is symbolized by '...'

Gates are parts of components and interactions and are used as links between both. The connection of a component with an interaction by corresponding gates induces identity on the corresponding phenomenons of the gates.

gate <- (gate-name phenomenon-name)

Relationships between physical properties are described by relations. Simple relations are equal, unequal, proportional etc.

```
rel ← ( relator rel-obj ...) | (rel-obj relator rel-obj)

rel-obj ← phyrop-name | value | functional-expr
```

Connection of components and interactions results in larger entities called aggregates. In the same way aggregates are combined to form bigger aggregates up to the whole technical system, that can be viewed as one aggregate consisting of a hierarchy of (sub-) aggregates.

```
aggregate ← aggregate-name | aggregate-desc
            component

aggregate-name ← identifier | AGGREGATE

aggregate-descr ← (aggregate-type
                   ((local-aggregate-name
                   aggregate)...)
                   ((local-interaction-name
                   interaction)...)
                   (gate-pair...)
                   (gate...))

aggregate-type ← aggregate-name

gate-pair ← (gate  gate)
```

In the definition of phenomenons, components, interactions and aggregates one may refer either to build-in types (PHENOMENON, COMPONENT, INTERACTION, AGGREGATE) or to user-defined objects (symbolized by the preterminal symbol identifier). These objects form a hierarchy representing the taxonomy of system theoretical terms. Each object describes some aspects of a part of a real system at a particular level of generality.

4. SAMPLE APPLICATION OF COMODEL

As a paradigm for the representation of technical systems we take the nitric acid cooler of Lapp/Powers [11], shown in figure 2. The function of this process is to cool a hot nitric acid stream before reacting it with benzene to form nitrobenzene. One top event for the system is a high temperature in the nitric acid reactor feed, since this could cause a reactor runaway.

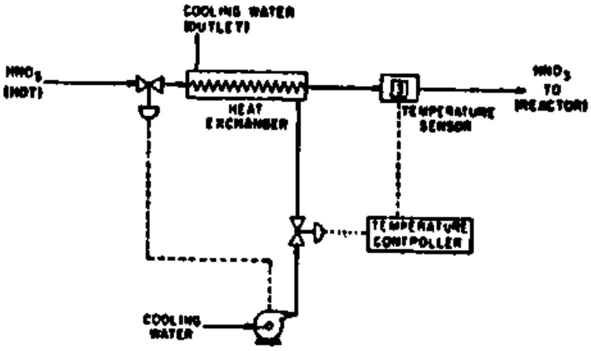

Fig. 2: The nitric acid cooler.

In order to show the applicability of COMODEL to describe a broad variety of technical mechanisms we choose two parts of the nitric acid cooler - the heat exchanger as energy flow system and a piston and cylinder, taken from the cooling water pump, which is assumed to be a piston pump, as an example for geometric interactions.

Heat Exchanger

The heat exchanger is an aggregate with four gates, two fluid streams passing through and interchanging heat energy. It can be viewed as constructed of two equal components, each a fluid stream with heat energy flowing off or flowing to (cf. figure 3). Therefore, such a component is an object with three gates, two fluid streams, and one energy stream (cf. figure 4).

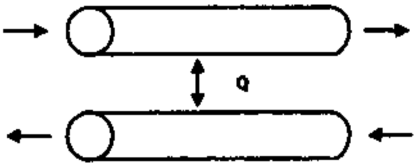

Fig. 3: The heat exchanger.

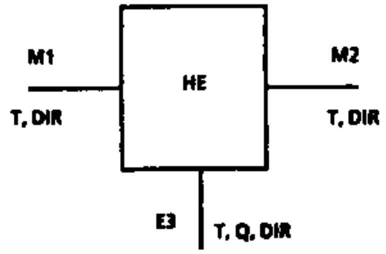

Fig. 4: The heat exchanger component.

The relevant physical properties (flow direction, temperature, heat stream) are combined to phenomenons.

```
(DEFOBJ MATERIALFLOW
       (PHENOMENON ((DIR DIRECTION)
                    (T TEMPERATURE))))

(DEFOBJ ENERGYFLOW
       (PHENOMENON  (DIR DIRECTION)
                    (T TEMPERATURE)
                    (Q HEATSTREAM))))
```

The heat exchanger component HE is

```
(DEFOBJ HE
        (COMPONENT
           ((M1 MATERIALFLOW)
            (M2 MATERIALFLOW)
            (E3 ENERGYFLOW))
           ((INV M1.DIR   M2.DIR)
            (E3.Q~|M1.T - M2.T|)
            (((M2.T >  M1.T)&(M2.DIR = IN)) v
             ((M1.T >  M2.T)&(M1.DIR = IN)) =
              (E3.DIR = OUT))
            (((M2.T <  M1.T)&(M2.DIR = IN)) v
             ((M1.T <  M2.T)&(M1.DIR = IN)) =
              (E3.DIR = IN))
            ((AVERAGE M1.T M2.T) = E3.T)))))
```

The heat exchanger component has three gates: two for the fluid stream (MI, M2) and one for the heat stream (E3). The fluid stream passes through the gates MI and M2 in opposite directions with respect to the centre of the component, i.e. at one gate it enters the component and at the other it leaves the component. This is stated by the first relation. The other relations define some relationships between the heat stream and the temperature of the fluid stream. The heat stream is proportional to the difference of temperatures, i.e. to the loss or gain of temperature of the fluid stream. If the entrance temperature is greater (less) than the exit-temperature then the heat stream leaves (enters) the component. The average temperature of the fluid stream is equal to the temperature of the heat stream.

Fluid stream and heat stream are described by interactions.

```
(DEFOBJ FLUIDSTREAM
        (INTERACTION
           ((M1  MATERIALFLOW)
            (M2  MATERIALFLOW))
           ((M1 o-o M2))))
```

The interaction FLUIDSTREAM has two gates (MI, M2). The relation states that the values of corresponding physical properties occurring at gate MI and gate M2 respectively are equal, except for the values of the flow directions.

```
(DEFOBJ HEATENERGY
        (INTERACTION
           ((E1  ENERGYFLOW)
            (E2  ENERGYFLOW))
           ((E1.Q ~|E1.T - E2.T|)
            (E1.Q = E2.Q)
            ((E1.T >  E2.T) = (E1.DIR = IN))
            ((E1.T <  E2.T) = (E1.DIR = OUT))
            (INV E1.DIR  E2.DIR))))
```

The interaction HEATENERGY has the two gates EI and E2. The relations define some relationships between the physical properties included in the phenomenon ENERGYFLOW occurring at the gates of HEATENERGY. The difference of the temperatures at both gates determines the heat stream. No heat is lost between both gates, therefore EI.Q = E2.Q. The heat flows from higher to lower temperature, i.e. it enters (leaves) the interaction at E1 if EI.T > (<) E2.T. The same holds for gate E2 because of the opposite directions.

The aggregate HEATEXCHANGER (cf. figure 5) consists of two components of type HE and one interaction of type HEATENERGY. The list of gate-pairs define how the constituents of the aggregate are composed, because a gate-pair represents an identification of two gates. For example HE1 ist composed with HF by identification of the gates HE1.E3 and HF.E1. A pair like (HE2.M1 M3) says that gate MI of HE2 is "free", i.e. not used for composition, and becomes therefore a gate of the aggregate. The description of the aggregate gates is adapted from the description of the corresponding component gates.

```
(DEFOBJ  HEATEXCHANGER
         (AGGREGATE
            ((HE1 HE)(HE2 HE))
            ((HF  HEATENERGY))
            ((HE1.M1 M1)(HE1.M2 M2)
             (HE2.M1 M3)(HE2.M2 M4)
             (HE1.E3 HF.E1)
             (HF.E2 HE2.E3))
            ((MI   MATERIALFLOW)
             (M2   MATERIALFLOW)
             (M3   MATERIALFLOW)
             (M4   MATERIALFLOW))))
```

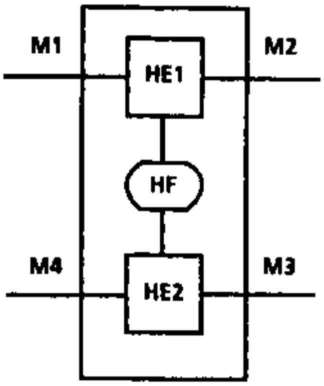

Fig. 5: The heat exchanger aggregate.

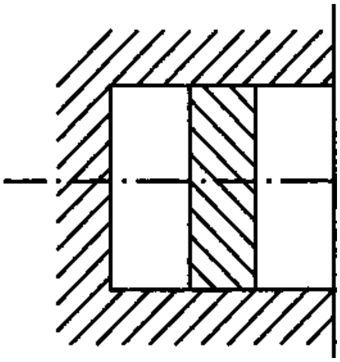

Fig. 6: Piston and cylinder.

Piston and cylinder

Piston and cylinder are the basic elements of the piston pump. They are viewed as two components and modelled by the two objects full cylinder and hollow cylinder, cf. figure 6. The phenomenons join together geometric and kinematic properties.

```
(DEFOBJ  FULLCYLINDERPROPERTIES
         (PHENOMENON
          ((R RADIUS)(L LENGTHKV VOLUME)
           (FR DEGREES-OF-FREEDOM)
           (POS POSITION))))
```

```
(DEFOBJ  HOLLOWCYLINDERPROPERTIES
         (PHENOMENON
          ((R RADIUS)(L LENGTHKV VOLUME)
           (FR DEGREES-OF-FREEDOM)
           (POS POSITION)
           (NUM-BOT  NUMBER-OF-BOTTOMS))))
```

A hollow body (e.g. a cylinder) can have no bottoms (pipe), one bottom (opened can), or two bottoms (closed can).

The two components are

```
(DEFOBJ  PISTON
          (FULLCYLINDER
           ((P  FULLCYLINDERPROPERTIES))
           ((P.R = XR)(P.L = XL)
            (P.V = TTR^L)
            (P.FR=(TRANS-FR(R) TRANS-FR(L)
                   TRANS-FR (ROT90(R))
                   ROT-FR(R)  ROT-FR(L)
                   ROT-FR (ROT90(R)))))))
```

```
(DEFOBJ  CYLINDER
          (HOLLOWCYLINDER
           ((C  HOLLOWCYLINDERPROPERTIES))
           ((C.R = XR)(C.L = XL)
            (C.V= πR^2L)
            (C.FR = (TRANS-FR(R) TRANS-FR(L)
                     TRANS-FR(ROT90(R))
                     ROT-FR(R)  ROT-FR(L)
                     ROT-FR(ROT90(R))))
            (C.NUM-BOT = 1))))
```

In the definition of piston and cylinder radius, length, degrees of freedom and number of bottoms are bound to particular values (X_R, X_L, TRANS-FR,..., 1). Every piston and cylinder has six degrees of freedom: translation along the R-axis, along the L-axis, and along the R-axis after a 90°-rotation, rotation around the R-axis, around the L-axis, and around the R-axis after a 90°-rotation. These definitions describe aspects of individual objects from the class of all fuilcylinders and all hollowcylinders respectively, only the values for the position in space are left open.

The FULLCYLINDER-IN-HOLLOWCYLINDER-interaction describes the geometric relationship, if the full cylinder is put into the hollow cylinder. It creates a new hollow cylinder, called pump chamber, with the length depending on the position of the piston. The piston retains two degrees of freedom in the resulting aggregate, cf. figure 7.

```
(DEFOBJ  FULLCYLINDER-IN-HOLLOWCYLINDER
        (INTERACTION
         ((F  FULLCYLINDERPROPERTIES)
          (H  HOLLOWCYLINDERPROPERTIES)
          (R  HOLLOWCYLINDERPROPERTIES))
         ((R.R = H.R)(R.L = F.POS(L) -H.POS(L))
          (R.NUM-BOT = 2)
          (F.FR = (TRANS-FR(L) ROT-FR(L))))))
```

The FULLCYLINDER-IN-HOLLOWCYLINDER-interaction has three gates (F, H, R). It combines the phenomenons FULLCYLINDERPROPERTIES and HOLLOWCYLINDERPROPERTIES to form a new phenomenon HOLLOWCYLINDERPROPERTIES. Gates R and H have the same radius, i.e. the radius of the interaction as a whole depends on the hollowcylinder which it is composed with. A similar relation holds for the length. The number of bottoms is 2. Only two degrees of freedom are left: translation along and rotation around the L-axis. The aggregate PUMPCHAMBER consists of two components of type CYLINDER and PISTON respectively and one interaction of type FULLCYLINDER-IN-HOLLOWCYLINDER. Here, the interaction plays a central role. Gate FIT.R is the gate of the aggregate. The composition of the three constituents is obvious.

The definition of the aggregate reflects the fact that the behaviour of a pumpchamber can be described mainly by the interaction of a piston and a cylinder.

```
(DEFOBJ  PUMPCHAMBER
     (AGGREGATE
     ((CASE  CYLINDER)(PISTON1  PISTON))
     ((FIT
        FULLCYLINDER-IN-HOLLOWCYLINDER))
            ((FIT.R  PC)(CASE.H  FIT.H)
             (PISTON1.F  FIT.F))
            ((PC  HOLLOWCYLINDERPROPERTIES))))
```

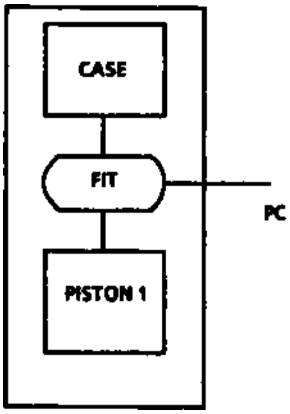

Fig. 7: The pump chamber aggregate.

S. CONCLUSION

COMODEL is a component oriented approach to the representation of knowledge about technical systems. In our description of COMODEL we have attached importance to its ability to represent function and geometric properties of components in a uniform way.

It is our intention to use COMODEL as a technical description language for expert systems for the area of technical systems, in particular for interpretation of the states of those systems and diagnosis of failures.

For this purpose a COMODEL-description represents a model of a technical-system. This model together with process signals should be processed by an inference engine. At this point additional heuristic knowledge about symptoms of defects as well as strategies for the search for interpretation and diagnosis is needed.

REFERENCES

[1] P. Hayes: The Naive Physics Manifesto. in: D.E.Michie (ed.): Expert Systems in the Microelectronic Age. Edinburgh Univ. Press, 1979.

[2] K.D. Forbus: Qualitative reasoning about physical processes. Proc. 7th IJCAI 1981, Vancouver, 326-330.

[3] K.D. Forbus: Modelling Motion with Qualitative Process Theory. Proc. AAAI-82, Pittsburgh, 1982, 205-208.

[4] K.D. Forbus: Measurement Interpretation in Qualitative Process Theory. Proc. 8th IJCAI 1983, Karlsruhe 315-320.

[5] J. de Kleer/J.S. Brown: Foundations of Envisioning. Proc. AAAI-82, Pittsburgh, 1982, 434-437.

[6] J. de Kleer/J.S. Brown: The Origin, Form and Logic of Qualitative Physical Laws. Proc. 8th IJCAI, Karlsruhe, 1158-1169

[7] B. Kuipers: Commonsense Reasoning About Causality: Deriving Behavior From Structure. Tufts University Working Papers in Cognitive Science, No. 18, May 1982.

[8] B. Kuipers: Getting the Envisionment Right. Proc. AAAI-82, Pittsburgh 1982, 209-212.

[9] B. Kuipers/J.P. Kassirer: How to Discover a Knowledge Representation for Causal Reasoning by Studying an Expert Physician. Proc. 8th IJCAI 1983, Karlsruhe, 49-56.

[10] P. Raulefs: Foundations of Expert Systems for Conceptional Design in Mechanical Engineering. Proc. 6th ECAI, Pisa, 1984.

[11] S.A. Lapp/G.J. Powers: Computer-aided Synthesis of Fault-trees. IEEE Transactions on Reliability, April 1977, 2-12.