

# Compositional Model-Based Design

Prasanta Bose and Shankar A. Rajamoney

Computer Science Department  
University of Southern California  
Los Angeles, CA 90089-0781

## Abstract

Generative model-based design of physical devices is difficult because the space of candidate design solutions is typically very large, and the cost of testing each candidate design to verify that it satisfies the desired behavior is substantial. In this paper, we present *compositional model-based design* (CMD), which addresses these problems by adopting a divide-and-conquer strategy and by interleaving design generation and testing. The CMD method divides the specified behavior into separate fragments and constructs locally correct *design fragments* for each. The design fragments are then incrementally composed until a design for the entire specified behavior is obtained. We show how the CMD may be used for the conceptual design of a steam-boiler control system.

## 1 Introduction

We present *compositional model-based design* (CMD), a divide-and-conquer strategy that facilitates the generative design of physical devices. The CMD method divides the specified behavior into separate fragments and constructs *design fragments* for each. The design fragments are then incrementally composed until a design for the entire specified behavior is obtained. The composition of independently designed fragments may result in a design which, due to adverse interactions between the fragments, violates the behavior established by one or more of them. The CMD method detects such interactions and revises the composite design to re-establish the desired behavior.

Consider the design of a boiler control system for a steam plant that is required to supply steam at a steady, fixed pressure despite variations in the demand of steam at the load. Figure 1a shows a candidate design from [Babcock and Wilcox, 1975] which establishes this regulatory behavior as follows:

Fluctuations in the load are sensed by a governor control mechanism which adjusts the outflow valve to maintain the steam flow rate to the load steady. To maintain the pressure at the drum steady, the change in the outflow rate is sensed and used to adjust the flow rate of the fuel/air mixture to the furnace. Consequently, the steam generation rate and the rate of steam flow from the boiler to the drum change to compensate for the outflow rate change, thereby, maintaining the steam pressures at the boiler and the drum steady.

To design such a device, compositional model-based design, given the basic input specifications<sup>1</sup> shown in

<sup>1</sup>We specify device behavior in terms of a *region diagram*

Figure 1b<sup>2</sup>, constructs independent design fragments (Figure 1c) to establish behavior fragments such as: a) Maintaining the water level in the drum steady, (established by assuming the level is uninfluenced). b) Maintaining the steam pressure in the drum steady. Next, it composes these two design fragments to establish the combined behavior. During composition, it may decide that the water supplied to the boiler for steam generation is from the drum (Figure 1d). In this case, the composite design is incorrect since it does not maintain the water level in the drum steady; instead, the water level decreases as the water is used to generate steam in the boiler. The CMD method detects this violation and revises the composite design to introduce an inflow of water into the drum from a tank to compensate for this loss (Figure 1e).

## 2 Design Fragments

A design fragment (DF) describes how a partial design establishes a behavior fragment. It consists of four parts: **Behavior B(DF)**: The behavior fragment to be established. Figure 2 illustrates a design fragment for maintaining the steam pressure in the drum steady (region  $R_1$ , Figure 1b).

**Design D(DF)**: The design that establishes the behavior fragment. Figure 2 shows a portion of the design stating that the drum and boiler must be containers with working fluids water and steam, and  $p_1$  must be a path connecting the drum and the boiler.

**Causal-relations network CRN(DF)**: A causal-relations<sup>3</sup> network (in the form of a data-dependency graph) that describes how the design establishes the behavior fragment. Figure 2 illustrates the causal-relations network that maintains the steam pressure in the drum steady. In particular, the network shows that the steam pressure in the drum is steady because it is

[Joskowicz, 1989] consisting of a set of non-overlapping regions and a set of transitions between them.

<sup>2</sup>Throughout the paper, we will use a simple notation for describing quantities: the first few letters describe the quantity, the last letter describes the object, and the subscript describes the container or the path. For example,  $FRS_{d,l}$  denotes the flow rate of steam from the drum to the load, and  $LW_d$  denotes the level of water in the drum.

<sup>3</sup>We use Forbus' Qualitative Process theory [Falkenhainer and Forbus, 1991] to represent domain models and causal relations.

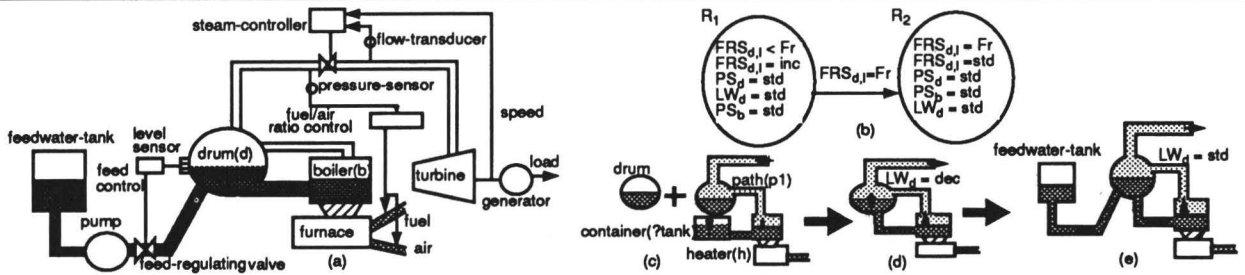


Figure 1: a) A boiler control system in a steam plant. b) Behavior input to CMD. c) Design fragments for maintaining the water level and steam pressure in the drum steady. d) Composing the two fragments to achieve the combined behavior. In this case, if the drum serves as a water source for the boiler, the water level in the drum will change, violating the previously established behavior. e) The composite design fragment is revised by introducing a compensating inflow of water.

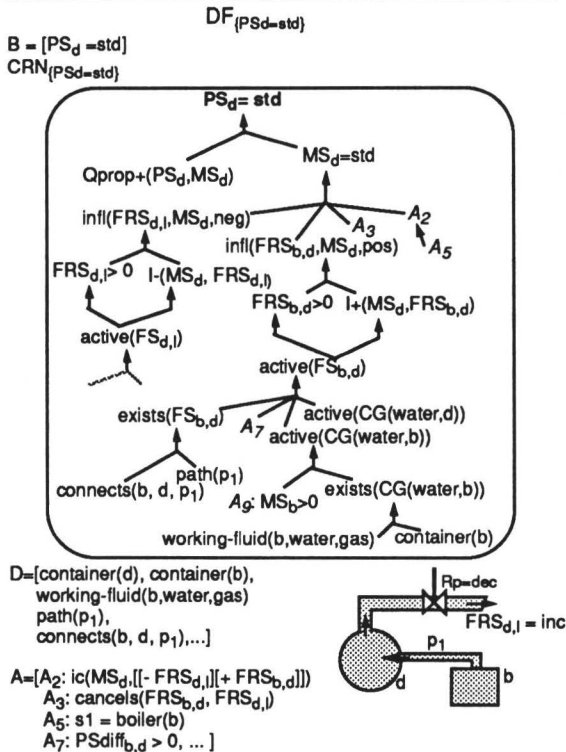


Figure 2: A design fragment.

qualitatively proportional to the steam mass which is steady since the partial influences on it (based on direct influences from the outflow and inflow) cancel.

**Assumptions A(DF):** A set of auxiliary assumptions under which the designed device works. There are four types of assumptions: (a) *Codesignation assumptions*. In extending a partial design, additional objects may be required. Often, instead of considering new objects, some of the existing objects may fulfill the role (a form of function-sharing). Codesignation [Chapman, 1988] explicitly considers the possibility that two objects may be identical. The causal-relations network in Figure 2 is based on the assumption,  $A_5$ , that the source codesignates with the boiler. (b) *Influence closure assumptions*. The computed qualitative change is valid only under the closed-world assumption that all the partial influences on the quantity are known [Falkenhainer and Forbus, 1991]. This assumption is explicitly represented

by listing all the known partial influences on the quantity. For example, in the causal-relations network shown in Figure 2, assumption  $A_2$  is an influence closure assumption specifying that the only known partial influences on the steam mass in the drum are a positive one from the steam inflow (from the boiler) and a negative one from the steam outflow (to the load). (c) *Influence dominance/cancellation assumptions*. When there are opposing partial influences on a quantity, the qualitative change to it may be ambiguous. Influence dominance/cancellation assumptions specify whether one set of partial influences dominates or cancels the other, thereby, permitting unambiguous computation of qualitative changes.<sup>4</sup> In Figure 2, the steam mass in the drum is steady based on an influence cancellation assumption  $A_3$  which states that the inflow rate of steam from the boiler equals the outflow rate of steam to the load. (d) *Quantity inequality assumptions*. Quantity inequalities that are not justified by the model fragment instances must be assumed as initial conditions or region boundary conditions. For example, the inequality assumption  $A_7$  specifies an initial condition: the steam pressure difference between source,  $s_1$ , and drum,  $d$ , must be greater than zero (to enable the steam inflow from the source).

A design fragment is locally correct if, based upon its design and auxiliary assumptions, its causal-relations network can be constructed (using the domain model axioms and qualitative inference rules) and if its causal-relations network entails all the behavior fragments specified in its behavior.

### 3 Behavioral Axioms

Two fundamental problems in composing design fragments are: 1) If the composite design is incorrect, that is, due to adverse interactions, previously established behavior is violated, how is blame assigned to individual elements of the causal-relations network? 2) How is the incorrect design fragment revised to re-establish the desired behavior or, more generally, what is the set of revision operators?

<sup>4</sup>This assumption may be refined to determine how the cancellation or dominance is achieved when the preliminary qualitative design is analyzed with quantitative information or exact equations.

To address both these problems, we declaratively specify the axioms describing the necessary and sufficient conditions for the different types of behavior fragments to hold. By examining these axioms, we derive all the *failure conditions*—violations of the necessary and sufficient conditions that may prevent the behavior fragment from being established. For each failure condition, we devise all the revision operators that will re-establish the behavior fragment by revising the failed condition(s) to satisfy the behavioral axioms. Compositional model-based design, therefore, solves the blame assignment problem by systematically searching the causal-relations network for the failure condition(s) associated with the behavior fragment. It solves the revision problem by applying one of the revision operators associated with the failure condition(s) to the incorrect design fragment.

Table 1 shows the behavioral axioms for qualitative changes and maintaining quantity inequalities.<sup>5</sup> Axiom 1.1 states that, in order for a quantity  $q$  to be increasing in a region  $R$ , it must necessarily be positively influenced by a quantity  $p$ , in  $R$ . In addition, in  $R$ , either all quantities  $q''$  that are negatively influenced do not codesignate with  $q$  or, if some do, then the set of positive influences on  $q$  must dominate the set of negative influences (Axiom 1.2), resulting in a net positive influence on  $q$  (under an influence closure assumption, supported by a set of codesignation assumptions). Axiom 2 specifies the conditions under which a quantity  $p$  partially influences a quantity  $q$ , positively or negatively, in a region  $R$ , and Axioms 3.1 and 3.2 specify the conditions under which a causal relation between  $p$  and  $q$  (direct influence or qualitative proportionality) holds in a region  $R$ . Axioms for the activity and existence of model fragment instances are obtained from their definitions in the domain model. The next set of axioms specify the necessary and sufficient conditions for a quantity  $q$  to remain greater than a constant value  $c$  in a region  $R$ .

Failure conditions are determined by systematically enumerating the reasons why a desired behavior fragment may not hold. For example, in a design fragment, if a quantity  $q$  is to be increasing in a region  $R$  and the causal-relations network does not establish it, then, from the behavioral axioms, it must be due to one of two failure conditions:

- (1) The influence closure assumption for  $q$  does not contain a positive partial influence on  $q$  (from Axiom 1.1). This failure may be revised in two ways (resulting in two distinct revision operators): a) forcing a quantity  $q'$  which is positively influenced to codesignate with  $q$ , or b) introducing a new positive influence  $p$  on  $q$ .
- (2) The influence closure assumption for  $q$  includes negative influences and the set of positive influences on  $q$  does not dominate the set of negative influences on  $q$  (from Axiom 1.2). This failure may be revised in three ways (resulting in three distinct revision operators): a) preventing the negatively influenced quantities from codesignating with  $q$ , b) forcing the set of positive influences to dominate the set of negative influences, or c) intro-

ducing new positive influences on  $q$  such that the new set of positive influences on  $q$  dominates over the set of negative influences.

## 4 Operations on a Design Fragment

The three complex operations on a design fragment are: 1) constructing design fragments for basic behavior fragments, 2) composing design fragments to establish their combined behavior, and 3) extending design fragments by posting auxiliary behavior fragments that enable or disable region transitions. Each operation consists of two phases: (a) a generative phase in which, working backwards from the behavior fragments to be established, all the necessary elements of the causal-relations network down to the design components and their structural relations are generated, and (b) a verification phase in which the complete causal-relations network based upon the design components and structural relations identified in the generative phase is constructed (as in qualitative simulation) and checked to see whether the desired behavior is established or violated (due to unwanted interactions). In each of these two phases, the basic revision operators described in the previous section are used to handle failure conditions.

### Constructing Basic Design Fragments

Qualitative changes to quantities are the basic behavior fragments. In the generative phase, the behavior of a new design fragment is initialized to the desired qualitative change and failure conditions are repeatedly detected and revised. When a revision operator introduces new objects (e.g. the source of a flow instance), all possible codesignations to the objects already in the design are considered and codesignation assumptions are formulated. In the verification phase, the complete causal-relations network is constructed based upon a set of codesignation assumptions. If the network does not establish the desired behavior, the failure conditions are identified and revised.

In the boiler control example, to establish the steam flow rate from the drum to the load to be increasing in region  $R_1$ , a new design fragment is created with its behavior initialized to this increase. This design fragment is incorrect because the necessary condition that the steam flow rate be positively influenced does not hold (Axiom 1.1; Table 1). One of the revision operators for this failure condition is applied, for example, the revision operator introducing a new positive influence on the steam flow rate. The operation involves searching the model fragments in the domain for a suitable causal relation that may introduce this partial influence. In this case, the operation finds, from the liquid flow model fragment, that the flow rate is inversely proportional to the resistance of the path. Accordingly, it attempts to recursively establish a decrease in the resistance of the path (Axioms 2, 3; Table 1). This recursive process continues until design components and structural relations are reached.

In the verification phase, the complete causal-relations network is constructed. For example, since a steam flow instance from the drum to the load is active, the mass

<sup>5</sup>Space restrictions prevent us from discussing the remaining. Also, for simplicity, we show only the axioms for the specific qualitative changes  $q = inc$  and inequalities  $q > c$ .

Table 1: Behavioral axioms for a qualitative change,  $q=inc$ , and a quantity inequality,  $q < c$ .

Let,  $\mathcal{P}+$  be the set of positive influences on a quantity  $q$  under a set of codesignation assumptions,  
 $= [p_1, \dots, p_n] \wedge [\forall p, q', \text{infl}(p, q', \text{pos}, R) \wedge \text{codesignates}(q', q) \leftrightarrow p \in \mathcal{P}+]$   
 $\mathcal{P}-$  be the set of negative influences on a quantity  $q$  under a set of codesignation assumptions,  
 $= [p_1, \dots, p_n] \wedge [\forall p, q', \text{infl}(p, q', \text{neg}, R) \wedge \text{codesignates}(q', q) \leftrightarrow p \in \mathcal{P}-]$

- A.  $\text{holds}(q = inc, R) \Leftrightarrow \exists p, \text{infl}(p, q', \text{pos}, R) \wedge \text{codesignates}(q', q)$  [1.1]  
 $\wedge [\forall p, \text{infl}(p, q', \text{neg}, R) \rightarrow \neg \text{codesignates}(q', q)] \vee [\exists \mathcal{P}-, \mathcal{P}+, \text{dominates}(\mathcal{P}+, \mathcal{P}-, q, \text{pos})]$  [1.2]  
 $\text{infl}(p, q, \text{pos}, R) \Leftrightarrow [\text{holds}(I+(q, p), R) \wedge \text{holds}(p > 0, R)] \vee [\text{holds}(I-(q, p), R) \wedge \text{holds}(p < 0, R)]$  [2]  
 $\vee [\text{holds}(Qprop+(q, p), R) \wedge \text{holds}(p = inc, R)] \vee [\text{holds}(Qprop-(q, p), R) \wedge \text{holds}(p = dec, R)]$   
 $\text{holds}(I\pm(q, p), R) \Leftrightarrow \exists M, \text{mf}(M) \wedge \text{active}(M, R) \wedge [\text{active}(M, R) \rightarrow \text{holds}(I\pm(q, p), R)]$  [3.1]  
 $\text{holds}(Qprop\pm(q, p), R) \Leftrightarrow \exists M, \text{mf}(M) \wedge \text{active}(M, R) \wedge [\text{active}(M, R) \rightarrow \text{holds}(Qprop\pm(q, p), R)]$  [3.2]
- B.  $\text{maintained}(q > c, R) \Leftrightarrow \text{holds}(q > c, R) \wedge [\forall q', \text{codesignates}(q', q) \rightarrow \neg \text{holds}(q' \leq c, R)]$  [1.1]  
 $\wedge [\text{holds}(q = std, R) \vee \text{holds}(q = inc, R)]$  [1.2]

## Composing Design Fragments

This operation composes two design fragments such that the composite design fragment establishes the combined behavior. In the generative phase, all possible codesignation assumptions across the design fragments are formulated. In the verification phase, the complete causal-relations network is generated under a set of codesignation assumptions and checked to see whether it continues to establish all the behavior fragments. New interactions may be due to two reasons: 1) A codesignation assumption may force two previously separate quantities to be the same, thereby, requiring an influence closure assumption that is the union of the two previously separate influence closure assumptions. 2) Composing design fragments may lead to new interactions between objects across the design fragments.

Figure 4b shows how the design fragments for two qualitative changes of region  $R_1$  are composed. Since all the objects are ground instances (e.g. drum, boiler), no codesignation assumptions are required. In the verification phase, upon constructing the complete causal-relations network, one of the previously established behavior fragments,  $PS_d = std$ , is violated. It was originally established under the assumption that the steam mass in the drum is uninfluenced. However, upon composition, the steam outflow from the drum (from the design fragment for increasing the outflow rate) negatively influences the steam mass. This interaction invalidates the earlier influence closure assumption, resulting in a decreasing steam pressure at the drum.

The failure condition for this violation is that the negative influence on the steam mass from the outflow is not canceled to keep the steam mass steady. The revision operator introduces a new positive influence on the steam mass in the drum. Upon searching the domain model, the fluid flow model fragment is found to have a suitable causal relation: a direct positive influence on the destination fluid mass by the flow rate. Accordingly, the operation attempts to recursively establish an active fluid flow instance (with steam as substance and the drum as the destination) whose flow rate is positive (Axioms 2, 3; Table 1). In this case, since the source and path are not specified, they may codesignate with existing objects or new objects may be introduced.

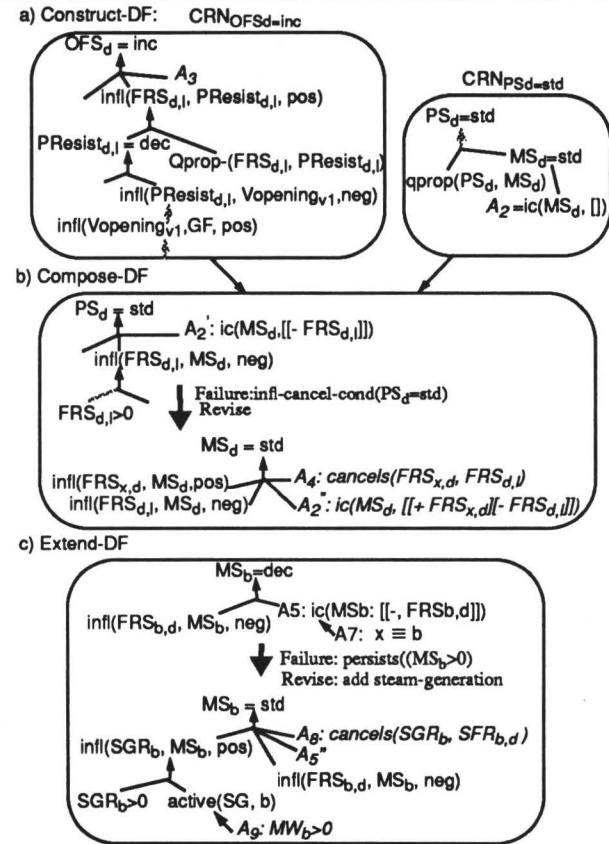


Figure 3: Construct, compose and extend operators.

of the steam in the drum is negatively influenced by the flow rate. This leads to other behavior fragments being established by the design including a decrease in the mass and pressure of the steam at the drum. In this case, the established behavior fragment, increasing of the outflow rate (by decreasing the path resistance, etc.), is not violated. Figure 4a shows relevant portions of the causal-relations networks that establish two qualitative changes of region  $R_1$  of the boiler control example.

## Extending Design Fragments

A design fragment may be extended by posting additional behavior fragments to be established in order to make a region transition when the transition condition is reached or to prevent an unwanted region transition when conditions other than the transition condition are reached. Potential transition conditions for a region are generated by determining which qualitative changes in the region will result in a limit point being reached. If the transition is not wanted, behavior fragments that prevent the transition point from being reached are added to the behavior of the design fragment to be later established using the construct and compose operations. If the transition is desirable, then nothing is done.<sup>6</sup>

In the composite design fragment of Figure 4b, if the source of the inflow to the drum codesignates with the boiler then, since the steam mass in the boiler is decreasing due to the flow, it may eventually become zero. In this case, the flow stops and the region makes an unwanted transition. Therefore, steam mass in the boiler must be maintained greater than zero, and this behavior fragment is posted as a necessary auxiliary behavior to be established by the design fragment. Figure 4c shows how this behavior fragment may be established by generating steam in the boiler at a rate equal to the rate of steam outflow from the boiler.

## 5 The CMD Method

Table 2 presents the algorithm for compositional model-based design. The algorithm accepts as input a device behavior expressed as a region diagram and a domain model, and produces as output a set of designs that generate the specified behavior. It uses a working set of design fragments and a set of design fragments for basic behavior fragments (qualitative changes to quantities). In Step 0, the working set is initialized such that each design fragment in the set corresponds to a region from the region diagram. The behavior of each design fragment is initialized to the set of qualitative changes in the corresponding region; the causal-relations network to empty; the design to the objects mentioned in the qualitative changes or quantity inequalities of the region; and, the assumptions to the quantity inequalities defining the boundary of the region.

The algorithm cycles, incrementally establishing the design fragments in the working set and the basic set. The body of the cycle consists of Steps 2 and 3. Step 2 constructs design fragments for each basic behavior fragment in the region, and incrementally composes them until the combined behavior for the region is established. Once every design fragment in the working set (corresponding to every region in the region diagram) has been established, Step 3 extends each one by posting auxiliary behavior to enable region transitions or disable unwanted transitions. These new behavior fragments are established by cycling back through the algorithm from

<sup>6</sup>This assumes that the device automatically shifts upon reaching the transition condition as in the boiler control system. [Bose and Rajamoney, 1993] discusses the design of devices in which the transition must be actively effected by a control mechanism.

Table 2: The CMD Method

|                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Method: $CMD(\mathcal{RD}, \mathcal{DM}, Maxtime)$                                                                                                                                                                                                                                                                                                                                                                                            |
| • Given: $\mathcal{RD}$ = region-diagram behavior specification consisting of a set of regions, $R$ , where each $R_i \in R$ consists of a set of qualitative changes $QC_i$ and a set of quantity inequalities (defining the region boundary) $QI$ ;<br>$\mathcal{DM}$ = Domain Model                                                                                                                                                        |
| • Output: A set of design solutions                                                                                                                                                                                                                                                                                                                                                                                                           |
| Let, $DF$ be a working set of design fragments<br>$DF_{base}$ be the set of design fragments that establish basic behavior fragments                                                                                                                                                                                                                                                                                                          |
| • Steps:                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 0. Initialize.<br>$DF = \{df \mid B(df) \leftarrow QC_i, CRN(df) \leftarrow [],$<br>$D(df) \leftarrow \text{initial design components mentioned in } R_i, A(df) \leftarrow QI_i\}$<br>$DF_{base} = \{\}$                                                                                                                                                                                                                                      |
| 1. If Exceeded( $Maxtime$ ) OR<br>$\text{Forall } df \in DF, \text{ Forall } b \in B(df), \text{ estd-in?}(b, df)$<br>$\text{Then Halt and if covers-behavior}(DF, \mathcal{RD}),$<br>$\text{output the design solution, union of all } D(df),$<br>$\text{and the assumptions, the union of all } A(df)$                                                                                                                                      |
| 2. $\text{Forall } df \in DF \text{ do}$<br>$\text{Forall } b \in B(df) \text{ and } \neg \text{established-in?}(b, df) \text{ do}$<br>(i) If $\neg \exists df_x \in DF_{base}, \text{ established-in?}(b, df_x)$<br>Then Construct- $DF(b, \mathcal{DM})$<br>and add the dfs returned to $DF_{base}$<br>ii) Let, $df_e$ be a design-fragment, such that,<br>$df_e$ is behavior-establisher( $b, DF_{base}$ )<br>iii) Compose- $DF(df, df_e)$ |
| 3. $\text{Forall } df \in DF \text{ do Extend-}DF(df)$                                                                                                                                                                                                                                                                                                                                                                                        |
| 4. Go to Step 1                                                                                                                                                                                                                                                                                                                                                                                                                               |

Step 1. Step 2 will construct (or retrieve from the basic set, if one is available) a design fragment that establishes the newly added behavior and will compose it with the design fragment.

The cycle continues until one of the end tests is met: either the user-set maximum time limit has been reached or all the behavior fragments and transitions in the region diagram have been established. The design solution is the union of the designs of the design fragments in the working set, and the assumptions underlying the design solution are the union of the assumptions of the design fragments in the working set.

### 5.1 An Example Trace

We illustrate how the CMD method designs a portion of the boiler control system shown in Figure 1. In the trace, we show only the choices made by the method that will lead to the design shown in Figure 1; if it were to make alternative choices, a different design would result. Figure 5.1a shows the construction of the design fragments for the basic behavior fragments of region  $R_1$ . The water level in the drum and the steam pressures in the drum and boiler are maintained steady by assuming they are uninfluenced. The outflow rate from the drum to the load is increased by decreasing the path resistance through a valve and lever mechanism attached to the governor sensing the steam demand changes at the load

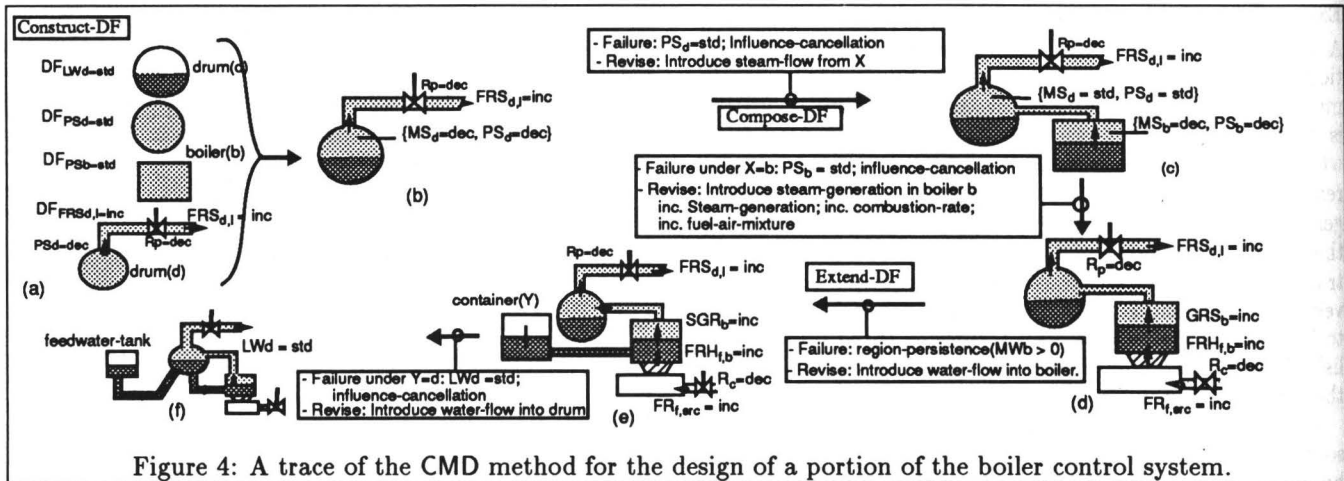


Figure 4: A trace of the CMD method for the design of a portion of the boiler control system.

(not shown in the figure). The composition of the design fragments for increasing the outflow rate and maintaining the pressure at the drum steady (Figure 5.1b), results in a violation of the latter behavior fragment since the outflow decreases the steam mass and pressure in the drum. To re-establish the steady pressure, an inflow of steam is introduced to cancel the outflow (Figure 5.1c).

Composition of the design fragment for maintaining the steam pressure in the boiler steady with the composite design fragment obtained above, under the assumption that the source of the steam inflow to the drum codesignates with the boiler, results in the steam pressure in the boiler decreasing due to the flow. Hence, the behavior fragment—maintaining the steam pressure in the boiler steady—is violated. To re-establish the steady pressure, a generation of steam is introduced to cancel the outflow (Figure 5.1d).

The composition of the design fragment for maintaining the water level in the drum steady to the composite design fragment obtained above, does not violate any of the previously established behavioral fragments. Step 3 of the method now extends the composite design fragment to prevent unwanted transitions. Since the water in the boiler is decreasing due to steam generation, it will eventually become zero, leading to an unwanted transition from region  $R_1$ . To prevent that, an auxiliary behavior fragment specifying that the mass of water in the boiler must be steady is posted. In the next iteration through the method, a design fragment for maintaining the water level in the boiler steady is constructed by introducing an inflow of water into the boiler to cancel the loss due to steam generation, and composed with the above design fragment (Figure 5.1e).

If the source of the inflow codesignates with the drum, then a previously established behavior fragment, maintaining the water level in the drum steady, is violated. To re-establish the steady water level, an inflow of water into the drum from a tank is introduced to compensate for the loss due to the outflow of water (Figure 5.1f).

## 6 Discussion

In model-based design, our work is closely related to Williams' work [Williams, 1989] on *Interaction-based Invention*. His IBIS system precompiles the domain the-

ory into an abstract space describing the interactions between quantities. It establishes a desired functional relationship between two quantities by finding a candidate path of potential interactions in this abstract space and verifying with the detailed domain model that the path establishes the desired relationship. More recently, Neville and Weld [Neville and Weld, 1992] have investigated potential problems due to the scaling up of William's approach. In contrast to IBIS's attempt to find a candidate design and then test it, the CMD method interleaves design generation and verification by constructing design fragments and verifying that they establish fragments of the specified behavior. Also, if a design fails, CMD can determine why it failed and revise it using its failure conditions and associated revision operators. Ongoing work on model abstractions and the design of devices that require explicit mechanisms to effect region transitions addresses some of the limitations of CMD.

## References

- [Babcock and Wilcox, 1975] Babcock and Wilcox. *Steam/Its Generation and Use*. Babcock and Wilcox Company, 1975.
- [Bose and Rajamoney, 1993] P. Bose and S. Rajamoney. Designing for transitions in multi-region devices. In *Proceedings of the 7th International Workshop on Qualitative Reasoning*, 1993.
- [Chapman, 1988] David Chapman. Planning for conjunctive goals. *Artificial Intelligence*, 1988.
- [Falkenhainer and Forbus, 1991] Brian Falkenhainer and K. Forbus. Compositional modelling: Finding the right model for the job. *Artificial Intelligence*, 1991.
- [Joskowicz, 1989] L. Joskowicz. Simplification and abstraction of kinematic behaviors. In *Proceedings of the IJCAI*, 1989.
- [Neville and Weld, 1992] Dorothy Neville and Daniel S. Weld. Innovative design as systematic search. In *Proceedings of the AAAI Fall Symposium on Design from Physical Principles* 1992.
- [Williams, 1989] Brain C. Williams. *Invention from First Principles via Topologies of Interaction*. PhD thesis, MIT AI Lab, Cambridge, MA, 1989.