Vol. 40 No. 1 Transactions of Information Processing Society of Japan Jan. 1999

Regular Paper

Specification of a Concurrent System Based on Propositional Logic

KAORU TAKAHASHI,† KANA SUGAWARA,†,☆ TOSHIHIKO ANDO,†
YASUSHI KATO† and NORIO SHIRATORI††

Formal Description Techniques (FDTs) are needed to achieve a highly reliable system design. In many FDTs, a system is specified by explicitly describing its behavior. However, in such a specification, the logical property of the system may be unclear and a modification of part of the specification may affect the entire specification. To cope with these problems, we propose a propositional logic based functional requirement description method for a concurrent system consisting of several subsystems. A method for synthesizing a state transition system as a formal specification from the given functional requirement specifications is also given with an application example.

1. Introduction

As information systems become large and complex, it is recognized that *FDTs* (*Formal Description Techniques*) are needed to achieve a highly reliable system design (e.g., Ref. 1)). A number of FDTs is based on the concept of state transition and the specification of a system is obtained by explicitly describing its behavior. However, in such a specification, the logical property of the system function may be unclear, and modifying a part of the specification may affect the entire specification.

To cope with these problems, a description method based on propositional logic has been proposed [2)~5)]. In that method, the following is focused on: the logical functional requirement of a system, the pre-condition for the execution of individual function, the input and output in the function execution, and the post-condition after the function execution. It is possible to describe the local functions of a system, thereby a modification of a specification can be flexibly performed. Consequently, we can think of a specification based on that method as a requirement specification of a system and a specification based on the state transition concept as a formal (behavioral) specification of a system. As a related and similar formalism, a specification method called STR [6)] has been proposed. But, that lacks logical and formal treatments.

In this paper, the above propositional logic-based method, in which *single* system is only dealt with, is extended so that a *concurrent system* consisting of a number of subsystems can be dealt with. For this purpose, we propose a functional requirement description method for each subsystem and a functional requirement description method for a concurrent system as a collection of subsystems. Furthermore, a method for synthesizing a sound and complete state transition system is given as a formal specification from the given functional requirement specifications. We also show an application example.

2. Requirement Specification and Formal Specification

As a description method for the requirement specification of a system, we use a *functional requirement* in which the specification is described as a collection of functions. The functional requirement of a concurrent system consists of the functional requirements of its subsystems. On the other hand, as a description method for the formal specification of a system, we use a *state transition system* in which the behavior of the system is explicitly represented.

In this section, first, the syntax and semantics of propositions are defined. A proposition is used as a condition of a function in the functional requirement of a system. Next, a functional requirement and a state transition system are defined.

2.1 Propositional Logic

Let $\mathcal{A}$ be a finite set of *atomic propositions* in propositional logic that depends on a target system. An atomic proposition represents a *basic* condition which enables a function of the system and/or a *basic* condition to be changed by a function of the system. A more general

† Sendai National College of Technology
☆ Presently with University of Electro-Communications
†† Research Institute of Electrical Communication, Tohoku University

or complex condition can be given by a proposition generated from the atomic propositions and logical operators.

Definition 1 The *propositions* generated from $\mathcal{A}$ are inductively defined as follows:

(1) $A \in \mathcal{A}$ is a proposition.
(2) Let f be a proposition, then $\neg f$ is a proposition.
(3) Let f and g be propositions, then $f \wedge g$, $f \vee g$ and $f \Rightarrow g$ are propositions. □

Definition 2 For an atomic proposition $A \in \mathcal{A}$, A or $\neg A$ is called a literal of A. □

Definition 3 Let $\mathcal{L}$ be the set of propositions generated from $\mathcal{A}$. Then, the semantics of a proposition is given by using an *interpretation* $\mathcal{I} : \mathcal{L} \to \{\textbf{true}, \textbf{false}\}$ where **true** and **false** are the truth values of propositions. In the following, A is an atomic proposition and f and g are propositions.

(1) $\mathcal{I}(A) = \textbf{true}$ or $\mathcal{I}(A) = \textbf{false}$
(2) Let $\mathcal{I}(f) = \textbf{true}$, then $\mathcal{I}(\neg f) = \textbf{false}$
(3) Let $\mathcal{I}(f) = \textbf{false}$, then $\mathcal{I}(\neg f) = \textbf{true}$
(4) Let $\mathcal{I}(f) = \textbf{true}$ and $\mathcal{I}(g) = \textbf{true}$, then $\mathcal{I}(f \wedge g) = \textbf{true}$ otherwise $\mathcal{I}(f \wedge g) = \textbf{false}$
(5) Let $\mathcal{I}(f) = \textbf{false}$ and $\mathcal{I}(g) = \textbf{false}$, then $\mathcal{I}(f \vee g) = \textbf{false}$ otherwise $\mathcal{I}(f \vee g) = \textbf{true}$
(6) Let $\mathcal{I}(f) = \textbf{true}$ and $\mathcal{I}(g) = \textbf{false}$, then $\mathcal{I}(f \Rightarrow g) = \textbf{false}$ otherwise $\mathcal{I}(f \Rightarrow g) = \textbf{true}$ □

Definition 4 A proposition f is said to be *true under an interpretation* $\mathcal{I}$ iff $\mathcal{I}(f) = \textbf{true}$. A proposition f is said to be *false under an interpretation* $\mathcal{I}$ iff $\mathcal{I}(f) = \textbf{false}$. If a proposition f is true under an interpretation $\mathcal{I}$, then we say that $\mathcal{I}$ *satisfies* f. □

Definition 5 Let f and g be propositions.

(1) f is *consistent* iff there is an interpretation which satisfies f.
(2) f is *inconsistent* iff f is not consistent.
(3) f is *dependent on* g iff either every interpretation satisfying g satisfies f, or every interpretation satisfying g satisfies $\neg f$.
(4) f is *independent of* g iff neither f is dependent on g nor g is dependent on f. □

From these definitions, the following proposition holds:

Proposition 1 Let γ be a consistent conjunction of literals. Then, an atomic proposition A is independent of γ iff A and $\neg A$ do not appear in γ. $\neg A$ is independent of γ iff A and $\neg A$ do not appear in γ. □

2.2 Functional Requirement

A concurrent system consists of a number of subsystems which interact with each other. We introduce the concept of a *port* to associate subsystems. That is, a subsystem performs interaction (input/output) with other subsystems (or environment) via the specified ports.

We denote a finite set of ports of a subsystem k as P_k. A finite set of inputs and a finite set of outputs are denoted as I_k and O_k, respectively. Using these, input and output actions of a subsystem are defined as follows:

Definition 6 *Input actions* Σ_k and *output actions* Δ_k of a subsystem k are the following sets:

$$\Sigma_k \subseteq P_k \times I_k$$
$$\Delta_k \subseteq P_k \times O_k$$ □

As actions performed by a subsystem, we consider the following three types of actions:

- *input* via a port from the outside
- *output* via a port to the outside
- *internal action*[⋆] independent of the outside

It can be considered that an action of a subsystem is possible to occur only if the subsystem satisfies some condition. From this point of view, the state set of a subsystem can be divided into a set in which the action is possible to occur and another set in which the action is not possible to occur. Moreover, it can be considered that there exist actions which become executable newly and actions which become unexecutable newly, in consequence of the execution of the action which may change conditions.

Definition 7 Let $\mathcal{A}_k$ ($\mathcal{A}_k \subset \mathcal{A}$) be a finite set of atomic propositions of a subsystem k, and $\mathcal{L}_k$ be the set of propositions generated from $\mathcal{A}_k$. Then, a *function* ρ of the subsystem k is a 3-tuple

$$\rho = \langle f_{in},\ a,\ f_{out} \rangle$$

where f_{in} is a *pre-condition* to be satisfied before execution of ρ ($f_{in} \in \mathcal{L}_k$), a is an input, output or internal action ($a \in \Sigma_k \cup \Delta_k \cup \{\varepsilon\}$), and f_{out} is a *post-condition* to be satisfied after execution of ρ ($f_{out} \in \mathcal{L}_k$). □

For understandability, a function ρ is represented as $\rho : f_{in} \overset{a}{\Rightarrow} f_{out}$; particularly, for an input, it is represented as $\rho : f_{in} \overset{a?}{\Rightarrow} f_{out}$ and is called an *input function*; for an output, it is represented as $\rho : f_{in} \overset{a!}{\Rightarrow} f_{out}$ and is called an *output function*; for an internal action, it is represented as $\rho : f_{in} \overset{\varepsilon}{\Rightarrow} f_{out}$ and is called an *internal function*. When a function is an input

[⋆] We denote every internal action as ε.

or output function, its action a is divided into the port part p and input/output part e, and is represented as $p : e$.

A functional requirement of the whole subsystem is defined as a collection of functions.

Definition 8 A *functional requirement* $\mathcal{R}_k$ of a subsystem k is a 5-tuple

$$\mathcal{R}_k = \langle R_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k \rangle$$

where R_k is a finite set of functions of the subsystem k, P_k is a finite set of ports of the subsystem k, Σ_k is a finite set of input actions of the subsystem k, Δ_k is a finite set of output actions of the subsystem k, and $\mathcal{A}_k$ is a finite set of atomic propositions of the subsystem k ($\mathcal{A}_k \subset \mathcal{A}$). □

We define a functional requirement of a concurrent system as a collection of functional requirements and initial conditions of subsystems.

Definition 9 A *functional requirement* $\mathcal{R}$ of a concurrent system is a set

$$\mathcal{R} = \{(\mathcal{R}_k, \gamma_{0k})\} \quad (1 \leq k \leq n)$$

where n is the number of subsystems, $\mathcal{R}_k = \langle R_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k \rangle$ is a functional requirement of the subsystem k ($\mathcal{A}_i \cap \mathcal{A}_j = \phi$, $i \neq j$, $\bigcup_{l=1}^{n} \mathcal{A}_l = \mathcal{A}$), and γ_{0k} is the initial condition of the subsystem k and is a consistent conjunction of literals of all the atomic propositions in $\mathcal{A}_k$. □

2.3 Formal Specification

A formal specification is a specification in which the behavior of a system is explicitly described. We use a state transition system as a formal specification.

Definition 10 A *state transition system* M is a 4-tuple

$$M = \langle Q, E, \longrightarrow, q_0 \rangle$$

where Q is a finite set of *states*, E is a finite set of *events* (*actions*), $\longrightarrow$ is a *transition relation* ($\longrightarrow \subseteq Q \times E \times Q$), and q_0 is the initial state ($q_0 \in Q$). □

A transition $(p, a, q) \in \longrightarrow$ is often represented as $p \xrightarrow{a} q$ which means that when the system is in a state p, if an event a occurs then the system state changes to a state q. In a transition $p \xrightarrow{a} q$, p is called the *source state* of the transition and q is called the *destination state* of the transition.

Definition 11 Let $\mathcal{X} \equiv X$ or $\mathcal{X} \equiv \neg X$ for any $X \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$)*. That is, $\mathcal{X}$ is a literal of X. By letting a state q of a formal specification $M = \langle Q, E, \longrightarrow, q_0 \rangle$ have a property as a proposition, we define q as $q = \bigwedge_{X \in \mathcal{B}} \mathcal{X}$. The property as a proposition that the state q has is called the *proposition of the state* q. □

Proposition 2 Let q be any state of a formal specification $M = \langle Q, E, \longrightarrow, q_0 \rangle$. Then, the proposition of q is consistent.

(Proof) From the above definition of a state, if we choose an interpretation $\mathcal{I}$ such that $\mathcal{I}(\mathcal{X}) =$ **true** for any $X \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$), then $\mathcal{I}(\bigwedge_{X \in \mathcal{B}} \mathcal{X}) = \mathcal{I}(q) =$ **true**. □

Proposition 3 Any atomic proposition $A \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$) is dependent on any state q of a formal specification $M = \langle Q, E, \longrightarrow, q_0 \rangle$.

(Proof) Clear from Proposition 1. □

Definition 12 Let f be the proposition of a state q of a formal specification $M = \langle Q, E, \longrightarrow, q_0 \rangle$ and $A \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$). If $\mathcal{I}(f \Rightarrow A) =$ **true** for every interpretation $\mathcal{I}$, then we say that q *satisfies* A and denoted as $q \models A$. If q does not satisfy A, then denoted as $q \models \neg A$. □

Proposition 4 Let f be the proposition of a state q of a formal specification $M = \langle Q, E, \longrightarrow, q_0 \rangle$ and $A \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$). For every interpretation $\mathcal{I}$, $\mathcal{I}(f \Rightarrow A) =$ **true** iff A appears as a literal in f.

(Proof) By definition, f is represented as $f \equiv \ldots \wedge \mathcal{X} \wedge \ldots$ ($\mathcal{X} \equiv A$ or $\mathcal{X} \equiv \neg A$). If A appears as a literal in f, i.e. $f \equiv \ldots \wedge A \wedge \ldots$, then by the definitions of $\wedge$ and $\Rightarrow$, $\mathcal{I}(f \Rightarrow A) =$ **true** for every interpretation $\mathcal{I}$. If A does not appear as a literal in f, i.e. $f \equiv \ldots \wedge \neg A \wedge \ldots$, then, by choosing an interpretation $\mathcal{J}$ such that $\mathcal{J}(l) =$ **true** for any literal l (including $\neg A$) in f, $\mathcal{J}(f \Rightarrow A) =$ **false**. □

We extend an interpretation $\mathcal{I}$ defined in Definition 3 as follows. This represents the interpretation of an atomic proposition $A \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$) in a state q.

$$\mathcal{I}(q)(A) = \begin{cases} \textbf{true} & \text{if } q \models A \\ \textbf{false} & \text{if } q \models \neg A \end{cases}$$

By this extension, we can define the *interpretation of a proposition in a state.*

Definition 13 Let q be a state of a formal specification $M = \langle Q, E, \longrightarrow, q_0 \rangle$ and g be a proposition generated from $\mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$). If g is interpreted as **true** in q, then we say that q *satisfies* g and denoted as $q \models g$. If q does not satisfy g, then denoted as $q \models \neg g$. □

3. Synthesis of Formal Specification

This section describes a method for synthesizing a formal specification from a given func-

* The symbol $\equiv$ represents that the left hand side and right hand side of $\equiv$ are syntactically equal.

tional requirement. First, a transformation method of a functional requirement of a subsystem into a canonical form is given. In a canonical form, all the propositions are represented as a conjunction of literals. Next, a method for synthesizing a formal specification from a functional requirement of a subsystem is given. Finally, we give a synthesis method of a formal specification for a concurrent system consisting of a number of subsystems.

3.1 Canonical Form

Let $\mathcal{R}_k = \langle R_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k \rangle$ be a functional requirement of a subsystem k.

First, we equivalently transform all the propositions in R_k into a disjunctive normal form $\gamma_1 \vee \ldots \vee \gamma_n$ where γ_i $(1 \leq i \leq n)$ is a conjunction of literals. For this transformation, see Refs. 8), 9) for example.

Next, apply the following rules to the transformed R_k as much as possible:

rule 1 $R_k \cup \{\gamma_1 \vee \ldots \vee \gamma_n \stackrel{a}{\Rightarrow} \gamma\} \Longrightarrow R_k \cup \{\gamma_1 \stackrel{a}{\Rightarrow} \gamma, \ldots, \gamma_n \stackrel{a}{\Rightarrow} \gamma\}$

rule 2 $R_k \cup \{\gamma_1 \wedge A \wedge \gamma_2 \stackrel{a}{\Rightarrow} \gamma\} \Longrightarrow R_k \cup \{\gamma_1 \wedge A \wedge \gamma_2 \stackrel{a}{\Rightarrow} \gamma \wedge A\}$
where neither A nor $\neg A$ $(A \in \mathcal{A}_k)$ appears in γ.

rule 3 $R_k \cup \{\gamma_1 \wedge \neg A \wedge \gamma_2 \stackrel{a}{\Rightarrow} \gamma\} \Longrightarrow R_k \cup \{\gamma_1 \wedge \neg A \wedge \gamma_2 \stackrel{a}{\Rightarrow} \gamma \wedge \neg A\}$
where neither A nor $\neg A$ $(A \in \mathcal{A}_k)$ appears in γ.

The resulting R_k is denoted as $\hat{R}_k$. We denote the transformed functional requirement as $\hat{\mathcal{R}}_k = \langle \hat{R}_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k \rangle$ and call it the *canonical form* of $\mathcal{R}_k$.

3.2 Synthesis of Formal Specification for Subsystem

Given a functional requirement $\mathcal{R}_k = \langle R_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k \rangle$ and the initial condition γ_{0k} of a subsystem k, the corresponding state transition system $\mathcal{T}_s(\mathcal{R}_k, \gamma_{0k}) = \langle Q, E, \longrightarrow, q_0 \rangle$ is synthesized by the following transformation $\mathcal{T}_s$:

Transformation $\mathcal{T}_s$

(1) Derive the canonical form $\hat{\mathcal{R}}_k = \langle \hat{R}_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k \rangle$ from $\mathcal{R}_k = \langle R_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k \rangle$.

(2) $Q = \{ \gamma \mid \gamma$ is a consistent conjunction of literals of all the atomic propositions in $\mathcal{A}_k \}$

(3) $E = \{ a \mid \rho : f_{in} \stackrel{a}{\Rightarrow} f_{out} \in \hat{R}_k \}$

(4) For each $\rho : f_{in} \stackrel{a}{\Rightarrow} f_{out} \in \hat{R}_k$, $\gamma \stackrel{a}{\longrightarrow} \gamma'$ $(\gamma, \gamma' \in Q)$ such that

(a) $\gamma \models f_{in}$

(b) $\gamma' \models f_{out}$

(c) If an atomic proposition $A \in \mathcal{A}_k$ is independent of f_{out}, then $\gamma \models A$ iff $\gamma' \models A$.

(5) $q_0 = \gamma_{0k}$ □

In the synthesized formal specification, states of the subsystem and inputs/outputs between the subsystem and the (external) environment are *explicitly* expressed. Thus, the formal specification can be regarded as an abstract implementation of the subsystem.

3.3 Synthesis of Formal Specification for Concurrent System

The synthesis of a formal specification for a concurrent system given here is based on the following notion:

Since a functional requirement of a concurrent system is a collection of functional requirements of subsystems, we make the whole of the conditions prescribing the execution of the functions of the subsystems correspond to the states of a formal specification of the concurrent system. We can think that an input/output interaction among subsystems can occur when the conditions of the related subsystems are simultaneously satisfied. This input/output can correspond to the occurrence of the event (action) of a transition in a formal specification. In order to enable a multicast communication among subsystems, a one-to-many communication mechanism is considered. Similar to process algebras such as CCS [10] and CSP [11], we assume that communication among subsystems is *synchronous*☆. Such an event is said to be *closed* within a concurrent system and is independent of the external environment of the concurrent system. An internal action in a subsystem is spontaneous and so in the whole concurrent system. We think that an input/output action not related with an interaction among subsystems, in other words, an action at a port that does not play a role of an interaction among subsystems, contributes to an interaction with the external environment of the concurrent system. Therefore, such an action is straightforwardly mapped to the event of a transition in a formal specification.

Consequently, the formal specification syn-

☆ Thus, in a formal specification, distinction between input and output of an event disappears; only the occurrence of an event is represented. This communication model makes it easy to analyze the formal specification. For implementation of synchronous communication, see Ref. 7) for example. Incidentally, *asynchronous* communication can be easily modeled by introducing a subsystem which represents a channel.

thesized from a functional requirement of a concurrent system represents not only event occurrences and state transitions inside the system but also the possibility of inputs/outputs from and to the external environment. If we identify input/output events inside the system, with internal actions ε, then the formal specification of the concurrent system can be treated equivalently with a formal specification of a subsystem.

3.3.1 Preliminary

Definition 14 We introduce the following notation for a function $\rho : f_{in} \stackrel{a}{\Longrightarrow} f_{out}$ ($a = p : e?$ or $a = p : e!$ or $a = \varepsilon$):

$pre(\rho) = f_{in}$

$post(\rho) = f_{out}$

$act(\rho) = a$

$$ev(\rho) = \begin{cases} e & \text{if } act(\rho) = p : e? \\ e & \text{if } act(\rho) = p : e! \\ \varepsilon & \text{otherwise} \end{cases}$$

$$port(\rho) = \begin{cases} p & \text{if } act(\rho) = p : e? \\ p & \text{if } act(\rho) = p : e! \\ undefined & \text{otherwise} \end{cases}$$

□

Definition 15 Let $\rho : f_{in} \stackrel{p:e!}{\Longrightarrow} f_{out} \in \hat{R}_k$ be an output function of a subsystem k in a functional requirement $\mathcal{R} = \{(\mathcal{R}_k, \gamma_{0k})\}$ ($1 \leq k \leq n$) of a concurrent system. Then, we define the set of input functions $peer(\rho)$ of the other subsystems as follows:

$$peer(\rho) = \left\{ \rho_j \mid \rho_j : f_{in_j} \stackrel{p:e?}{\Longrightarrow} f_{out_j}, \rho_j \in \bigcup_{l \neq k} \hat{R}_l \right\}$$ □

A function belonging to $peer(\rho)$ can input the event specified in ρ.

Definition 16 Let $\mathcal{R} = \{(\mathcal{R}_k, \gamma_{0k})\}$ ($1 \leq k \leq n$) be a functional requirement of a concurrent system. If there exist an output function $\rho \in \hat{R}_i$ and an input function $\rho' \in \hat{R}_j$ ($i \neq j$) such that $port(\rho) = port(\rho')$, then we call $port(\rho)$ an *internal port*. We denote the whole of functions which have internal ports as $INT(\mathcal{R})$. On the other hand, we call a port an *external port* if it is not an internal port. We denote the whole of functions which have external ports as $EXT(\mathcal{R})$*. □

3.3.2 Synthesis

Given a functional requirement $\mathcal{R} = \{(\mathcal{R}_k, \gamma_{0k})\}$ ($1 \leq k \leq n$) of a concurrent system, the corresponding state transition system $\mathcal{T}(\mathcal{R}) = \langle Q, E, \longrightarrow, q_0 \rangle$ is synthesized by the following transformation $\mathcal{T}$:

Transformation $\mathcal{T}$

(1) For each $\mathcal{R}_k = \langle R_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k \rangle$ ($1 \leq k \leq n$), derive the canonical form $\hat{\mathcal{R}}_k = \langle \hat{R}_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k \rangle$.

(2) $Q = \{ \gamma \mid \gamma$ is a consistent conjunction of literals of all the atomic propositions in $\mathcal{A} \}$

(3) $E = \{ ev(\rho) \mid \rho \in INT(\mathcal{R}) \} \cup \{ \varepsilon \} \cup \{ act(\rho) \mid \rho \in EXT(\mathcal{R}) \}$

(4) For each $\rho \in \bigcup_k \hat{R}_k$, $\gamma \stackrel{a}{\longrightarrow} \gamma'$ ($\gamma, \gamma' \in Q$) such that

(a) If ρ is an output function and $\rho \in INT(\mathcal{R})$, then for a non-empty subset $\mathcal{S} \subseteq peer(\rho)$:

(i) $\gamma \models pre(\rho)$

(ii) $\forall \rho_j \in \mathcal{S} \bullet \gamma \models pre(\rho_j)$

(iii) $\gamma' \models post(\rho)$

(iv) $\forall \rho_j \in \mathcal{S} \bullet \gamma' \models post(\rho_j)$

(v) If an atomic proposition $A \in \mathcal{A}$ is independent of $post(\rho) \wedge \bigwedge_j post(\rho_j)$, then $\gamma \models A$ iff $\gamma' \models A$.

(vi) $a = ev(\rho)$

(b) If ρ is an internal function or $\rho \in EXT(\mathcal{R})$, then:

(i) $\gamma \models pre(\rho)$

(ii) $\gamma' \models post(\rho)$

(iii) If an atomic proposition $A \in \mathcal{A}$ is independent of $post(\rho)$, then $\gamma \models A$ iff $\gamma' \models A$.

(iv) $a = act(\rho)$

(5) $q_0 = \bigwedge_k \gamma_{0k}$ □

Example 1 Let $\mathcal{R}$ be a functional requirement of a concurrent system (see **Fig. 1**):

$\mathcal{R} = \{(\mathcal{R}_1, A \wedge B), (\mathcal{R}_2, C), (\mathcal{R}_3, D)\}$

where $\mathcal{R}_1 = \langle \{\rho_1 : A \stackrel{p_a:a!}{\Longrightarrow} \neg A, \rho_2 : B \stackrel{p_b:b?}{\Longrightarrow} A\}, \{p_a, p_b\}, \{p_b : b\}, \{p_a : a\}, \{A, B\} \rangle$, $\mathcal{R}_2 = \langle \{\rho_3 : C \stackrel{p_a:a?}{\Longrightarrow} \neg C, \rho_4 : \neg C \stackrel{p_b:b!}{\Longrightarrow} C\}, \{p_a, p_b\}, \{p_a : a\}, \{p_b : b\}, \{C\} \rangle$, and $\mathcal{R}_3 = \langle \{\rho_5 : \neg D \stackrel{p_a:a?}{\Longrightarrow} \neg D, \rho_6 : D \stackrel{\varepsilon}{\Longrightarrow} D, \rho_7 : D \stackrel{p_c:c!}{\Longrightarrow} \neg D\}, \{p_a, p_c\}, \{p_a : a\}, \{p_c : c\}, \{D\} \rangle$. The canonical forms of $\mathcal{R}_1$, $\mathcal{R}_2$ and $\mathcal{R}_3$ are as follows:

$\hat{\mathcal{R}}_1 = \langle \{\rho_1 : A \stackrel{p_a:a!}{\Longrightarrow} \neg A, \rho_2 : B \stackrel{p_b:b?}{\Longrightarrow} A \wedge B\}, \{p_a, p_b\}, \{p_b : b\}, \{p_a : a\}, \{A, B\} \rangle$

$\hat{\mathcal{R}}_2 = \mathcal{R}_2$

$\hat{\mathcal{R}}_3 = \mathcal{R}_3$

The internal ports are p_a and p_b. The external port is p_c. The functions are classified into:

* Note that $INT(\mathcal{R}) \cap EXT(\mathcal{R}) = \phi$.

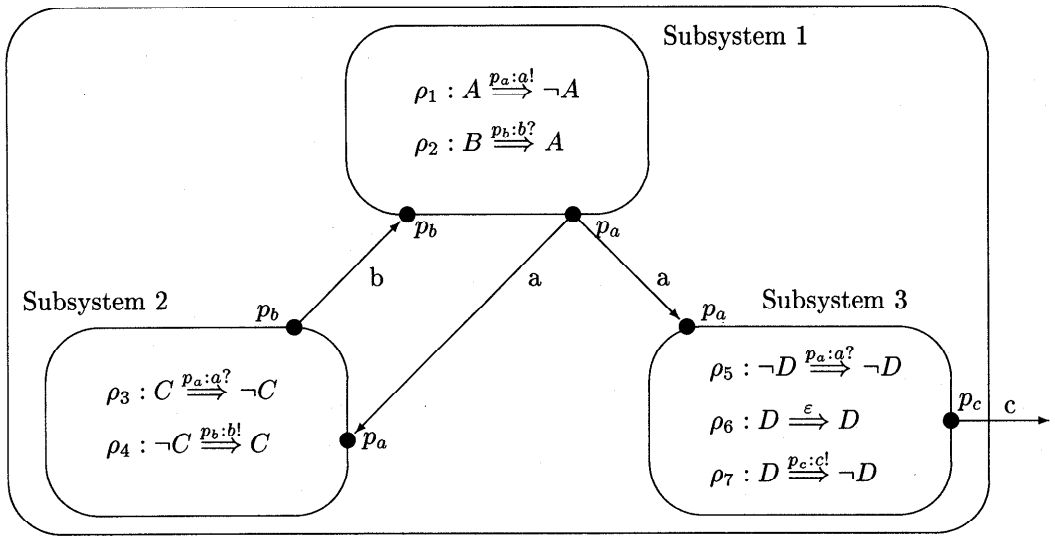


Fig. 1 An example of a functional requirement of a concurrent system.

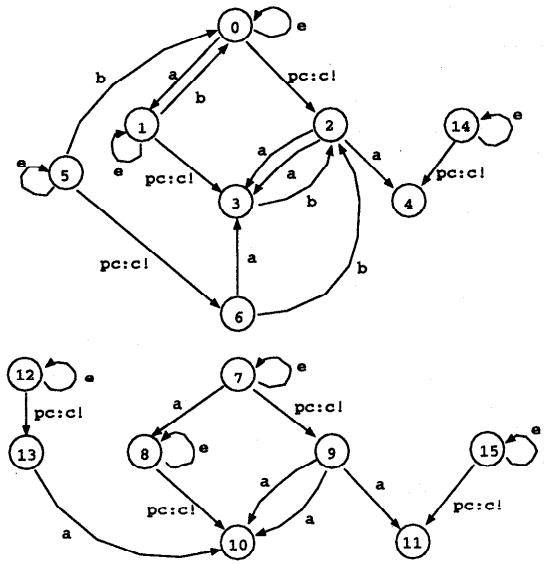


Fig. 2 The synthesized formal specification of the example concurrent system (note: e represents ε).

$INT(\mathcal{R}) = \{ \rho_1, \rho_2, \rho_3, \rho_4, \rho_5 \}$ and
$EXT(\mathcal{R}) = \{ \rho_7 \}$.[☆]

The peer functions to ρ_1 and ρ_4 are as follows:

$peer(\rho_1) = \{ \rho_3, \rho_5 \}$
$peer(\rho_4) = \{ \rho_2 \}$

The corresponding formal specification (state transition system) is synthesized as **Fig. 2**. In the formal specification, the state 0 is the initial state. The correspondence between the states and propositions is shown in **Table 1**. An action $p_c : c!$ is an output to the external environment of the concurrent system and the others are actions closed within the concurrent system. □

[☆] ρ_6 belongs to neither $INT(\mathcal{R})$ nor $EXT(\mathcal{R})$.

Table 1 Correspondence between states and propositions in the synthesized formal specification of the example concurrent system.

state	*proposition*
0	$A \wedge B \wedge C \wedge D$
1	$\neg A \wedge B \wedge \neg C \wedge D$
2	$A \wedge B \wedge C \wedge \neg D$
3	$\neg A \wedge B \wedge \neg C \wedge \neg D$
4	$\neg A \wedge B \wedge C \wedge \neg D$
5	$A \wedge B \wedge \neg C \wedge D$
6	$A \wedge B \wedge \neg C \wedge \neg D$
7	$A \wedge \neg B \wedge C \wedge D$
8	$\neg A \wedge \neg B \wedge \neg C \wedge D$
9	$A \wedge \neg B \wedge C \wedge \neg D$
10	$\neg A \wedge \neg B \wedge \neg C \wedge \neg D$
11	$\neg A \wedge \neg B \wedge C \wedge \neg D$
12	$A \wedge \neg B \wedge \neg C \wedge D$
13	$A \wedge \neg B \wedge \neg C \wedge \neg D$
14	$\neg A \wedge B \wedge C \wedge D$
15	$\neg A \wedge \neg B \wedge C \wedge D$

4. Soundness and Completeness

In this section, we discuss the soundness and completeness as relations between a functional requirement and a formal specification (state transition system). The soundness represents that for any transition in the formal specification, there exists a function in the functional requirement which is satisfied by the transition. The completeness represents that for any executable function in the functional requirement,

there exists a transition in the formal specification which reflects the function.

We deal with only functional requirements and formal specifications of *concurrent systems*; we omit those of subsystems since a similar discussion to the traditional researches [2),5)] is applicable to subsystems.

4.1 Soundness

Let $\mathcal{R} = \{(\mathcal{R}_k, \gamma_{0k})\}$ $(1 \leq k \leq n)$ be a functional requirement of a concurrent system, and $M = \langle Q, E, \longrightarrow, q_0 \rangle$ be a formal specification in which the interpretation of propositions generated from $\mathcal{A}$ is defined. An action in E is an input, output, internal or *closed* action. In particular, as described in the previous section, a closed action means an action as a result of an interaction among subsystems within the concurrent system.

Definition 17 We say that a transition $t = \langle p \xrightarrow{a} q \rangle \in \longrightarrow$ *satisfies* a function $\rho : f_{in} \overset{b}{\Longrightarrow} f_{out} \in \bigcup_k R_k$ and write $t \models \rho$ if the following conditions are satisfied:

(1) If a is an internal, input or output action, then:
  (a) $p \models f_{in}$
  (b) $q \models f_{out}$
  (c) If an atomic proposition $A \in \mathcal{A}$ is independent of f_{out}, then $p \models A$ iff $q \models A$.
  (d) $b = a$

(2) If a is closed, then for some output function $\rho_o \in \bigcup_k R_k$ and some non-empty subset $\mathcal{S} \subseteq peer(\rho_o)$:
  (a) $\rho \in \{\rho_o\} \cup \mathcal{S}$
  (b) $\forall \rho_x \in \{\rho_o\} \cup \mathcal{S} \bullet p \models pre(\rho_x)$
  (c) $\forall \rho_x \in \{\rho_o\} \cup \mathcal{S} \bullet q \models post(\rho_x)$
  (d) If an atomic proposition $A \in \mathcal{A}$ is independent of $\bigwedge_{\rho_x \in \{\rho_o\} \cup \mathcal{S}} post(\rho_x)$, then $p \models A$ iff $q \models A$.
  (e) $ev(\rho) = a$ □

Although there is a distinction between actions, the above definition states that the source state and destination state of the transition have to satisfy the pre-condition and post-condition of the function, respectively. Actions are differentiated depending on their types, i.e. actions which can result from synchronization among the related subsystems, or actions which can result from a single subsystem.

Using this definition, we define the soundness as follows:

Definition 18 A state transition system M is *sound with respect to* a functional requirement $\mathcal{R}$ of a concurrent system if the following conditions are satisfied:

(1) $q_0 \models \bigwedge_k \gamma_{0k}$

(2) For every transition $t \in \longrightarrow$, there exists a function $\rho \in \bigcup_k R_k$ such that $t \models \rho$. □

The first condition of this definition implies that for every atomic proposition $A \in \mathcal{A}$,

> A is positively (negatively) appeared in the proposition of q_0 *iff* A is positively (negatively) appeared in $\bigwedge_k \gamma_{0k}$.

4.2 Completeness

The soundness does not guarantee that states and transitions are sufficient for a functional requirement.

As a preliminary of the definition of the completeness, we introduce a homomorphism between state transition systems.

Definition 19 Let $M = \langle Q, E, \longrightarrow, q_0 \rangle$ and $M' = \langle Q', E', \longrightarrow', q_0' \rangle$ be state transition systems in which the interpretation of the propositions generated from $\mathcal{A}$ is defined and which are sound with respect to a functional requirement $\mathcal{R} = \{(\mathcal{R}_k, \gamma_{0k})\}$ $(1 \leq k \leq n)$ of a concurrent system. A mapping $\varphi : Q \longrightarrow Q'$ is said to be a *homomorphism from M into M'* if the following conditions are satisfied:

(1) $\varphi(q_0) = q_0'$

(2) $\langle p \xrightarrow{a} q \rangle \in \longrightarrow$ implies $\langle \varphi(p) \xrightarrow{a} \varphi(q) \rangle \in \longrightarrow'$

(3) For a state $p \in Q$ of M and a proposition f generated from $\mathcal{A}$, $p \models f$ iff $\varphi(p) \models f$. □

If a homomorphism φ is a bijection and the inverse function φ^{-1} is a homomorphism from M' into M, then φ is called an *isomorphism*. Then, M and M' are said to be *isomorphic*.

Using the concept of a homomorphism, we define the completeness as follows:

Definition 20 Let $M = \langle Q, E, \longrightarrow, q_0 \rangle$ be a state transition system in which the interpretation of the propositions generated from $\mathcal{A}$ is defined and which is sound with respect to a functional requirement $\mathcal{R} = \{(\mathcal{R}_k, \gamma_{0k})\}$ $(1 \leq k \leq n)$ of a concurrent system. M is *complete with respect to* $\mathcal{R}$ if there exists a homomorphism from M' into M for every sound state transition system M' with respect to $\mathcal{R}$. □

From the definition, a complete state transition system represents the largest one of sound ones except isomorphic ones, and it has all the states and transitions necessary for a functional requirement.

4.3 Soundness and Completeness of Synthesized Specification

The following theorem validates the transformation $\mathcal{T}$ by showing that the synthesized formal specification is sound and complete.

Theorem 1 The formal specification $\mathcal{T}(\mathcal{R}) = \langle Q, E, \longrightarrow, q_0 \rangle$ synthesized by the transformation $\mathcal{T}$ from a functional requirement $\mathcal{R} = \{(\mathcal{R}_k, \gamma_{0k})\}$ $(1 \leq k \leq n)$ of a concurrent system is sound and complete with respect to $\mathcal{R}$.

(Proof) The soundness is clear from the correspondence between its definition and the construction of $\mathcal{T}(\mathcal{R})$. We will prove the completeness. Let $M' = \langle Q', E', \longrightarrow', q_0' \rangle$ be a sound state transition system with respect to $\mathcal{R}$ in which the interpretation of the propositions generated from $\mathcal{A}$ is defined. Let $\varphi : Q' \longrightarrow Q$ be a mapping such that for a state $q \in Q$ of which proposition is the same as that of a state $q' \in Q'$, $\varphi(q') = q$. From the construction of Q, such a $q \in Q$ exists for any $q' \in Q'$. We prove the completeness by showing that this φ is a homomorphism from M' into $\mathcal{T}(\mathcal{R})$.

(1) Since both M' and $\mathcal{T}(\mathcal{R})$ are sound with respect to $\mathcal{R}$, $q_0' \models \bigwedge_k \gamma_{0k}$ and $q_0 \models \bigwedge_k \gamma_{0k}$. This means that the proposition of q_0' and that of q_0 are the same. From the definition of φ, $\varphi(q_0') = q_0$.

(2) Let $\langle p' \xrightarrow{a'} q' \rangle \in \longrightarrow'$. From the definition of φ, $\varphi(p') \models pre(\rho)$ and $\varphi(q') \models post(\rho)$ for every function $\rho \in \bigcup_k R_k$ satisfied by this transition. Furthermore, if $A \in \mathcal{A}$ is independent of $post(\rho)$, then $\varphi(p') \models A$ iff $\varphi(q') \models A$. Thus, from the construction of $\mathcal{T}(\mathcal{R})$, $\langle \varphi(p') \xrightarrow{a'} \varphi(q') \rangle \in \longrightarrow$.

(3) Since the proposition of $p \in Q'$ and that of $\varphi(p) \in Q$ are the same from the definition of φ, $p \models f$ iff $\varphi(p) \models f$ for any proposition f generated from $\mathcal{A}$. □

5. Application

We apply our specification and synthesis methods to a simple mobile system. As shown in **Fig. 3**, the system consists of three subsystems:

(1) *ms* — a mobile station
(2) *base1* — a base station interacting with the mobile station
(3) *base2* — another base station

When *ms* communicates with a base station, it first issues a signal *enter* to start a communication, then communicates with each other using a signal *talk*, and finally issues a signal *leave* to terminate the communication. *ms* can communicate with at most one base station at a time. Through a movement, it can switch to another base station. A *handover* procedure is executed when *ms* moves from the current base station to another base station.

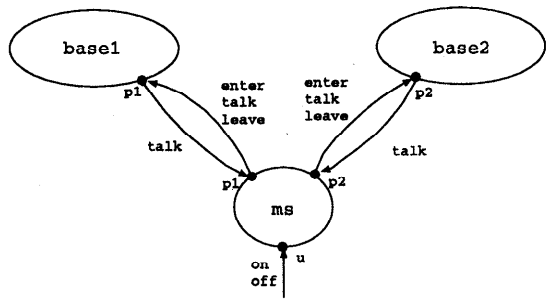


Fig. 3 An example of a mobile system.

A functional requirement $\mathcal{R}$ of this mobile system may be described as follows:

$$\mathcal{R} = \{(\mathcal{R}_1, \gamma_{01}), (\mathcal{R}_2, \gamma_{02}), (\mathcal{R}_3, \gamma_{03})\}$$

$$\mathcal{R}_1 = \langle R_1, \{u, p_i \mid 1 \leq i \leq 2\}, \{u : on, u : off, p_i : talk \mid 1 \leq i \leq 2\}, \{p_i : enter, p_i : talk, p_i : leave \mid 1 \leq i \leq 2\}, \{power, conn, loc_i, startHandover, handover, forceLeave \mid 1 \leq i \leq 2\}\rangle$$

$$\gamma_{01} = \neg power \wedge \neg conn \wedge loc_1 \wedge \neg loc_2 \wedge \neg startHandover \wedge \neg handover \wedge \neg forceLeave$$

$$\mathcal{R}_2 = \langle R_2, \{p_1\}, \{p_1 : enter, p_1 : talk, p_1 : leave\}, \{p_1 : talk\}, \{conn1\}\rangle$$

$$\gamma_{02} = \neg conn1$$

$$\mathcal{R}_3 = \langle R_3, \{p_2\}, \{p_2 : enter, p_2 : talk, p_2 : leave\}, \{p_2 : talk\}, \{conn2\}\rangle$$

$$\gamma_{03} = \neg conn2$$

$\mathcal{R}_1$, $\mathcal{R}_2$ and $\mathcal{R}_3$ are functional requirements of *ms*, *base1* and *base2*, respectively. R_1, R_2 and R_3 are described in **Table 2**☆. The atomic propositions and their meanings used in $\mathcal{R}$ are explained in **Table 3**.

In $\mathcal{R}$, the status of the power of the mobile station *ms* is affected by an input *on*/*off* at the external port *u*. *ms* is initially in the cell of the base station *base1*, and a movement of *ms* is modeled by an internal action ε.

Figure 4 shows the formal specification (state transition system) synthesized from $\mathcal{R}$

☆ The names of functions are omitted.

Table 2 The functions of *ms*, *base1* and *base2*.

R_1	$\neg power \overset{u:on?}{\Longrightarrow} power$
	$power \wedge \neg conn \wedge loc_i \overset{p_i:enter!}{\Longrightarrow} conn \quad (1 \leq i \leq 2)$
	$conn \wedge loc_i \wedge \neg(startHandover \vee handover \vee forceLeave) \overset{p_i:talk!}{\Longrightarrow} conn \wedge loc_i \ (1 \leq i \leq 2)$
	$conn \wedge loc_i \wedge \neg(startHandover \vee handover \vee forceLeave) \overset{p_i:talk?}{\Longrightarrow} conn \wedge loc_i \quad (1 \leq i \leq 2)$
	$conn \wedge loc_i \wedge \neg(startHandover \vee handover \vee forceLeave) \overset{p_i:leave!}{\Longrightarrow} \neg conn \quad (1 \leq i \leq 2)$
	$power \wedge \neg conn \overset{u:off?}{\Longrightarrow} \neg power$
	$conn \wedge \neg(startHandover \vee handover \vee forceLeave) \overset{u:off?}{\Longrightarrow} forceLeave$
	$forceLeave \wedge loc_i \overset{p_i:leave!}{\Longrightarrow} \neg power \wedge \neg conn \wedge \neg forceLeave \quad (1 \leq i \leq 2)$
	$conn \wedge \neg(startHandover \vee handover \vee forceLeave) \overset{\varepsilon}{\Longrightarrow} startHandover$
	$startHandover \wedge loc_i \overset{p_i:leave!}{\Longrightarrow} \neg startHandover \wedge handover \wedge \neg loc_i \wedge loc_j \quad (1 \leq i \leq 2,\ 1 \leq j \leq 2,\ i \neq j)$
	$handover \wedge loc_i \overset{p_i:enter!}{\Longrightarrow} \neg handover \quad (1 \leq i \leq 2)$
R_2	$\neg conn1 \overset{p_1:enter?}{\Longrightarrow} conn1$
	$conn1 \overset{p_1:talk!}{\Longrightarrow} conn1$
	$conn1 \overset{p_1:talk?}{\Longrightarrow} conn1$
	$conn1 \overset{p_1:leave?}{\Longrightarrow} \neg conn1$
R_3	$\neg conn2 \overset{p_2:enter?}{\Longrightarrow} conn2$
	$conn2 \overset{p_2:talk!}{\Longrightarrow} conn2$
	$conn2 \overset{p_2:talk?}{\Longrightarrow} conn2$
	$conn2 \overset{p_2:leave?}{\Longrightarrow} \neg conn2$

Table 3 The atomic propositions and their meanings in the mobile system.

atomic Prop.	*meaning*
power	the power of *ms* is on.
conn	*ms* is connecting with a base station.
loc_1	*ms* is in the cell of *base1*.
loc_2	*ms* is in the cell of *base2*.
startHandover	start the handover procedure.
handover	in handover.
forceLeave	force *ms* to terminate communication.
conn1	*base1* is connecting with *ms*.
conn2	*base2* is connecting with *ms*.

by the transformation $\mathcal{T}$. In the formal specification, only the states reachable from the initial state (state 0) are shown; the unreachable states are omitted. If we assume the initial state of a system, then unreachable states represent system states that never appear in the behavior of the system. Thus, the formal specification shown in Fig. 4 just represents the whole behavior of the system.

In the formal specification, the characteristic transitions are as follows:

(1) *state* 2 $\overset{\varepsilon}{\longrightarrow}$ *state* 4: a movement of *ms* from the cell of *base1* to the cell of *base2*.

(2) *state* 4 $\overset{leave}{\longrightarrow}$ *state* 5: start the handover procedure.

(3) *state* 5 $\overset{enter}{\longrightarrow}$ *state* 6: terminate the handover procedure.

(4) *state* 6 $\overset{\varepsilon}{\longrightarrow}$ *state* 10: a movement of *ms* from the cell of *base2* to the cell of *base1*.

(5) *state* 10 $\overset{leave}{\longrightarrow}$ *state* 11: start the handover procedure.

(6) *state* 11 $\overset{enter}{\longrightarrow}$ *state* 2: terminate the handover procedure.

It is confirmed that the behavior of interactions between the mobile and base stations, including a movement of the mobile station and its influence (handover), are clearly reflected in the synthesized formal specification.

6. Conclusion

In this paper, we have proposed a propositional logic-based description method for a requirement specification (functional requirement) of a concurrent system consisting of a number of subsystems. Each subsystem is given as a local functional description, and their association is represented by a provision of a communication mechanism between subsystems. This communication mechanism has an ability in a synchronous, multicast communication.

Given a requirement specification (functional requirement) of a concurrent system, we have

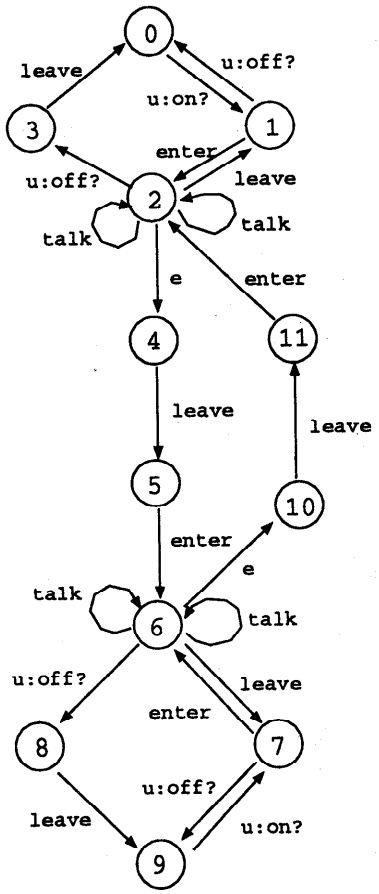


Fig. 4 The synthesized formal specification of the mobile system (note: e represents ε).

also presented a method to synthesize a formal specification (state transition system) that explicitly expresses the behavior of the system. It has been shown that the synthesized formal specification is sound and complete with respect to the requirement specification. This gives a validity of the synthesis method.

In order to show applicability of our methods, a description of a simple mobile system has been given, and a formal specification representing its behavior has been synthesized.

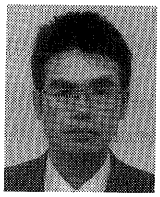
Our future study includes:

- applying our methods to the development of a real system;
- providing an ability in a hierarchical description of a functional requirement, to enhance the flexibility of description and to decrease the number of states in a formal specification;
- enabling a description of communication constraints, to reflect communication conditions in a concurrent system, if any;
- introducing the concept of mobility as in π-calculus [12]; and
- developing a support system for functional requirement description and formal specification synthesis.

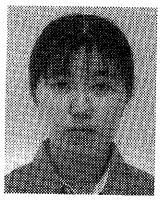
References

1) Turner, K.J.: *Using Formal Description Techniques*, John Wiley & Sons (1993).
2) Song, K., Togashi, A. and Shiratori, N.: A Requirement Description Method Based on Propositional Logic and Its Semantic Description by State Transition System, *Trans. IPS Japan*, Vol.37, No.4, pp. 511–519 (1996) (in Japanese).
3) Togashi, A., Usui, N., Song, K. and Shiratori, N.: Synthesis of Formal Specifications from User Requirements and its Flexibility, Technical Report of IEICE, IN94-200 (1995).
4) Song, K.H., Togashi. A. and Shiratori, N.: Verification and Refinement for System Requirements, *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, Vol.E78-A, No.11, pp.1468–1478 (1995).
5) Usui, N.: Development Support of Formal Specifications based on Functional Requirements, Master's Thesis, Tohoku University (1995) (in Japanese).
6) Hirakawa, Y. and Takenaka, T.: Telecommunication Service Description Using State Transition Rules, *Proc. 6th Int. Work. Software Specification and Design*, pp.140–147 (1991).
7) Cheng, Z., Takahashi, K., Shiratori, N. and Noguchi, S.: An Automatic Implementation Method of Protocol Specifications in LOTOS, *IEICE Trans. Information and Systems*, Vol.E75-D, No.4, pp.543–556 (1992).
8) Nagao, M. and Fuchi, K.: *Logic and Semantics*, Iwanami-Kouza, Information Science 7, Iwanami (1983) (in Japanese).
9) Schagrin, M.L., Rapaport, W.J. and Dipert, R.R.: *Logic: A Computer Approach*, McGraw-Hill (1985).
10) Milner, R.: *Communication and Concurrency*, Prentice-Hall (1989).
11) Hoare, C.A.R.: *Communicating Sequential Processes*, Prentice-Hall (1985).
12) Milner, R., Parrow, J. and Walker, D.: A Calculus of Mobile Processes, *Information and Computation*, Vol.100, pp.1–77 (1992).

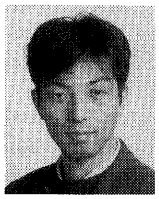
(Received April 30, 1998)
(Accepted November 9, 1998)



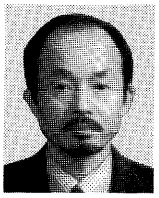
Kaoru Takahashi joined the Research Institute of Electrical Communication, Tohoku University in 1974. He received Ph.D. degree from Tohoku University in 1992. From 1993 to 1995, he was a senior visiting researcher in the Advanced Intelligent Communication Systems Laboratories. He is now an associate professor in the Department of Information and Communication Engineering, Sendai National College of Technology. Currently he is doing research on formal description techniques which cover the whole process of distributed systems design. He is a member of IPSJ and IEICE.



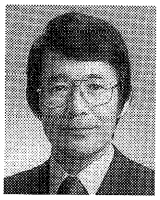
Kana Sugawara graduated from the Department of Information and Communication Engineering, Sendai National College of Technology in 1998. She is now with the Department of Communications and Systems, Faculty of Electro-Communications, University of Electro-Communications. She has researched on distributed system specification and verification techniques.



Toshihiko Ando is a lecturer in the Department of Information Engineering, Sendai National College of Technology. His research interests are formal specification and modeling of concurrent systems and application of them to software engineering. He is a member of IPSJ, the Society of Physics Japan and the Society for Science on Form Japan.



Yasushi Kato is currently a professor in the Department of Information Engineering, Sendai National College of Technology. His research interests include protocol design and verification, formal description techniques, temporal logic, dynamical behavior of cellular automata, and education method for digital technology. He received the Dr. of Eng. degree from Tohoku University in 1978. He has been at Twente University in the Netherlands as a residential researcher from 1995 to 1996 and has engaged in the research on formal description techniques. He is a member of IPSJ, IEICE and JSEE. He is also a senior visiting researcher of Miyagi, Fukushima and Iwate Prefectural Institute of Technology.



Norio Shiratori received the B.E. degree from Tokai University, Tokyo, in 1972, and the M.E. and Ph.D. degrees in Electrical and Communication Engineering from Tohoku University, Sendai, in 1974 and 1977, respectively. He has joined RIEC (Research Institute of Electrical Communication), Tohoku University, Sendai, since 1977, and he is now a professor of Computer Science at RIEC of Tohoku University. His current research interests include Flexible computing and its application to Post modern distributed system and symbiosis cyberspace of human and computers. He has been named a Fellow of the IEEE for his contributions to the field of computer communication networks.