Regular Paper

System Requirements and Formal Specifications of Hierarchical Reactive Systems☆

Fumiaki Kanezashi† and Atsushi Togashi††

A methodology for the description of system requirements and formal specifications of reactive systems and the synthesis of formal specifications is presented. Based on a hierarchical structure of system properties a hierarchical assertional language is used as a requirement language and hierarchical state transition systems are used as formal specifications. Sound and complete formal specifications are synthesized from system requirements automatically. Modularity and reusability are supported by the introduction of requirement and specification modules and a partial order relation over these modules. The methodology has a practical significance because desired specifications of reactive systems can be derived or synthesized from user requirements on system functions in a systematic and stepwise way.

1. Introduction

A reactive system is characterized by being event-driven and continuously reacting to external stimuli. For a complex reactive system, operational descriptions of the whole system might be too tedious to handle for rapid prototyping and analysis of the system's behavior. In such cases, it is more convenient to express system requirements in a functional and assertional manner and to derive system specifications and implementations with different levels of granularity in a stepwise refinement way.

In this paper, we propose a new methodology for the description of system requirements and formal specifications of reactive systems and the synthesis of formal specifications. The unique feature of our methodology is that based on a hierarchical structure of system properties a hierarchical assertional language is used as a requirement language and hierarchical state transition systems, which combine characteristics of traditional labeled transition systems and Kripke structures, are used as formal specifications to specify changes of both actions and states. Formal specifications are synthesized from system requirements automatically and can be taken as models of system requirements. Requirement and specification modules are introduced to specify behavior of system components. These modules form a hierarchical structure by a partial order relation over them and is reusable based on system context. Modularity and reusability make it easy to specify reactive systems succinctly and to construct prototype systems.

The system overview of our software development method of reactive systems is shown in **Fig. 1**. Requirement acquisition systems for obtaining system requirements and compilers to produce programs are not contained in our present research. The simulator[15] is a graphical tool for representing static structure and simulating dynamic behavior of reactive systems with diagnosis system[16]. The verifier[17] is to verify that formal specifications of reactive systems satisfy linear and branching time properties based on compositional verification methods without generation of global transition systems. Reflection system is proposed in the paper[14]. The present research is an extension of previous work[11)~13)] with hierarchical structures of system properties, requirements, and specifications.

In the literature on reactive systems, numerous formal specification methods have been proposed, including Statecharts[1], Modechart[3], VFSM[4], SDL[5], LOTOS[7], and Estelle[6], etc.. The conventional state machine oriented approaches such as SDL and Estelle and algebraic approach such as LOTOS are suitable for the purpose of description and investigation of total behavior of systems. Statecharts and Modechart are also state diagrams with depth and orthogonality concepts and broadcast communication, but there are no system requirement definitions in these visual formalisms

† Graduate School of Science and Engineering, Shizuoka University
†† Department of Computer Science, Shizuoka University
☆ A preliminary version of this paper was published in the Proceedings of the 7th IEEE International Conference on Parallel and Distributed Systems '2000.

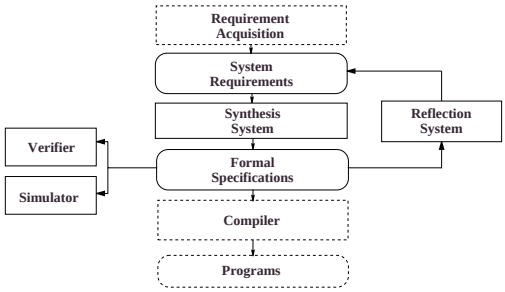

Fig. 1 System overview.

and without state interpretation and condition-dependent decomposition, concepts like modularity and reusability are not supported.

The outline of this paper is as follows: In Section 2, based on the description of a hierarchical structure of system properties, we give the definitions of system requirement module and system requirement. In Section 3, after proposing definitions of state transition module and hierarchical formal specification, we discuss the semantics of formal specification by introducing global state transition systems. In Section 4 the soundness and completeness of formal specifications with respect to system requirements is defined in two levels: module level and system level. Section 5 gives synthesis method of formal specifications as hierarchical state transition systems from assertional system requirements. Section 6 is the conclusion.

2. Hierarchical System Requirement

2.1 System Property Structure

Requirements of a system can be described as expressions based on propositional logic. Let $\mathcal{P}$ be a set of *atomic propositions*. A *literal* is an atomic proposition A or the negation of an atomic proposition $\neg A$ for $A \in \mathcal{P}$. Each atomic proposition describes a specific property of the intended system under the target of design. Some atomic propositions describe general properties of the system while some just specify detailed or partial aspects of the system and are dependent on the appearance of other properties. So the system property structure can be defined by a partial order relation over partitions of $\mathcal{P}$, and it can be seen as a hierarchical structure with propositions independent of others at the topmost level.

Definition 2.1 A *system property structure* is a tuple $\mathcal{S} = \langle \mathcal{P}, \mathcal{L}, \succ, \mathcal{L}_0 \rangle$, where

(1) $\mathcal{P}$ is a finite set of all atomic propositions;

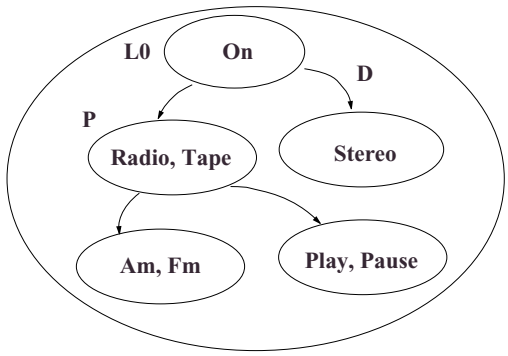

Fig. 2 System overview.

(2) $\mathcal{L}$ is a family of subsets of $\mathcal{P}$, which forms a partition of $\mathcal{P}$, i.e.
   (1) $\mathcal{P} = \bigcup_{L \in \mathcal{L}} L$,
   (2) $\forall L_i, L_j \in \mathcal{L}(i \neq j), L_i \cap L_j = \emptyset$;

(3) $\succ \subseteq \mathcal{L} \times \mathcal{L}$ is a partial order (irreflective, transitive) relation representing the dependency relation over sets of atomic propositions. If $L_1 \succ L_2$, then L_2 is said to be *dependent* on L_1;

(4) $\mathcal{L}_0 \subseteq \mathcal{L}$ is a family of sets of propositions at the topmost level such that for all $L_0 \in \mathcal{L}_0$, there are no $L \in \mathcal{L}$ such that $L \succ L_0$. □

Example 2.1 A very simple cassette tape player will be used as an example throughout the paper to illustrate hierarchical specification techniques proposed in the paper. The machine can be functional only when the power is *On*. The player functions as a *Radio* or as a cassette *Tape* player. A switch for the selection of *Stereo* or mono acts separately. If the player is working as a *Radio*, band *Am* or *Fm* can be selected. Cassette *Tape* player can *Play* or *Pause* by pressing respective control buttons. The *system property structure* of the player is shown in the following definitions and **Fig. 2**.

(1) $\mathcal{P} = \{On, Radio, Tape, Stereo, Am, Fm, Play, Pause\}$;

(2) $\mathcal{L} = \{\{On\}, \{Radio, Tape\}, \{Stereo\}, \{Am, Fm\}, \{Play, Pause\}\}$;

(3) $\succ = \{(\{On\}, \{Radio, Tape\}), (\{On\}, \{Stereo\}), (\{Radio, Tape\}, \{Am, Fm\}), (\{Radio, Tape\}, \{Play, Pause\})\}$;

(4) $\mathcal{L}_0 = \{\{On\}\}$.

2.2 System Requirement

In a reactive system, a system function can be specified as a transition on different condi-

tions according to a specific system action or input. When its pre-condition holds in the current state, a system function can be invoked by a specific input and executed, possibly producing some appropriate outputs. After the execution, the current state is changed into a new one, and another function can be applicable in the new state.

Definition 2.2 Let L be a set of propositions. A *function requirement* over L is a tuple $\rho_L = \langle id, a, f_{in}, o, f_{out} \rangle$, where

(1) id is the *name* of the function;
(2) a is an *input symbol* of the function;
(3) f_{in} is a *pre-condition* of the function to be satisfied before execution, which is represented as a consistent conjunction of literals of atomic propositions in L;
(4) o is an *output symbol* of the function;
(5) f_{out} is a *post-condition* of the function to be satisfied after execution, which is represented as a consistent conjunction of literals of atomic propositions in L. □

When a name is omitted, a function requirement $\rho = \langle a, f_{in}, o, f_{out} \rangle$ is often abbreviated as $\rho : f_{in} \overset{a/o}{\Rightarrow} f_{out}$. In this paper we describe system requirements as a set of system requirement modules with a hierarchical structure constructed by partial order relation over modules to avoid the state explosion problem.

Definition 2.3 Let $\mathcal{S} = \langle \mathcal{P}, \mathcal{L}, \succ, \mathcal{L}_0 \rangle$ be a system property structure and $L \in \mathcal{L}$. A *system requirement module* over L is a tuple $R_L = \langle F, \gamma_0, B, \Sigma, O, \mathcal{C} \rangle$, where

(1) F is a set of *function requirements* over L;
(2) γ_0 is an *initial condition* of the system requirement module represented as a consistent conjunction of literals in L;
(3) B is a *background condition* represented as a consistent proposition using atomic propositions in $\bigcup_{L_u \succ L} L_u$;
(4) Σ is a set of *input symbols* of the system requirement module. If all the system requirement modules have the same set of input symbols, Σ is the set of *system input symbols*;
(5) O is a set of *output symbols* of the system requirement module. If all the system requirement modules have the same set of output symbols, O is the set of *system output symbols*;
(6) $\mathcal{C}$ is a set of *consistent propositions* which should be satisfied by all the states of the system requirement module. □

A *background condition* is a propositional constraint represented by atomic propositions of upper level system requirement modules. A module can be activated only when its *background condition* is satisfied by some conditions of upper level modules. $\mathcal{C}$ is used to specify local constraints of a system requirement module.

Based on the definition of a system requirement module, a system requirement can be defined as a hierarchical structure of system requirement modules with a partial order relation over these modules and initial modules at the topmost level.

Definition 2.4 Let $\mathcal{S} = \langle \mathcal{P}, \mathcal{L}, \succ, \mathcal{L}_0 \rangle$ be a system property structure. A *system requirement* on $\mathcal{S}$ is a tuple $\mathbf{R} = \langle \mathcal{R}, \mathcal{R}_0, \mathcal{C} \rangle$, where

(1) $\mathcal{R} = \{ R_L \mid L \in \mathcal{L} \text{ in } \mathcal{S} \}$ is a family of *system requirement modules* over $L \in \mathcal{L}$ in $\mathcal{S}$;
(2) $\mathcal{R}_0$ is a family of initial system requirement modules over elements of $\mathcal{L}_0$ in $\mathcal{S}$;
(3) $\mathcal{C}$ is a set of *constraints* expressed by consistent propositions, describing the global properties of the system. □

The partial order relation $\succ$ on $\mathcal{L}$ can be extended to the system requirement: Let R_1 and R_2 be system requirement modules over L_1 and L_2 in $\mathcal{L}$ of $\mathcal{S}$ respectively. $R_1 \succ R_2$ iff $L_1 \succ L_2$ in $\mathcal{S}$.

Example 2.2 The *system requirement modules* of the Radio/Tape player described in the previous subsection are defined as follows.

(1) $R_{\{On\}}$: *Power Control*
  (1) $F = \{\neg On \overset{Power}{\Rightarrow} On, On \overset{Power}{\Rightarrow} \neg On\}$;
  (2) $\gamma_0 = \neg On$;
  (3) $B = \mathbf{true}$;
  (4) $\Sigma = \{Power\}$;
  (5) $O = \emptyset$;
  (6) $\mathcal{C} = \emptyset$.
(2) $R_{\{Radio,Tape\}}$: *Radio and Tape Selection*
  (1) $F = \{Radio \overset{RT}{\Rightarrow} Tape, Tape \overset{RT}{\Rightarrow} Radio\}$;
  (2) $\gamma_0 = Radio$;
  (3) $B = On$;
  (4) $\Sigma = \{RT\}$;
  (5) $O = \emptyset$;

(6) $\mathcal{C}$ =$\{Radio \vee Tape,\ Radio \supset \neg Tape,\ Tapa \supset \neg Radio\}$.

(3) $R_{\{Stereo\}}$: *Stereo Control*

(1) $F = \{\neg Stereo \overset{S}{\Rightarrow} Stereo, Stereo \overset{S}{\Rightarrow} \neg Stereo\}$;
(2) $\gamma_0 = Stereo$;
(3) $B = On$;
(4) $\Sigma = \{S\}$;
(5) $O = \emptyset$;
(6) $\mathcal{C}$ =$\emptyset$.

(4) $R_{\{Play,Pause\}}$: *Tape Control*

(1) $F = \{\neg Play \overset{PL}{\Rightarrow} Play,$
$Play \wedge \neg Pause \overset{PA}{\Rightarrow} Pause,$
$Play \wedge Pause \overset{PA}{\Rightarrow} \neg Pause,$
$Play \overset{Stop}{\Rightarrow} \neg Play \wedge \neg Pause\}$;
(2) $\gamma_0 = \neg Play \wedge \neg Pause$;
(3) $B = Tape$;
(4) $\Sigma = \{PL, PA, Stop\}$;
(5) $O = \emptyset$;
(6) $\mathcal{C}$ =$\{Pause \supset Play\}$.

(5) $R_{\{Am,Fm\}}$: *Radio Control*

(1) $F = \{Am \overset{AF}{\Rightarrow} Fm, Fm \overset{AF}{\Rightarrow} Am\}$;
(2) $\gamma_0 = Am$;
(3) $B = Radio$;
(4) $\Sigma = \{AF\}$;
(5) $O = \emptyset$;
(6) $\mathcal{C}$ =$\{Am \vee Fm,\ Am \supset \neg Fm,\ Fm \supset \neg Am\}$.

The *system requirement* $\mathbf{R} = \langle \mathcal{R}, \mathcal{R}_0, \mathcal{C} \rangle$ of the player is defined as follows.

(1) $\mathcal{R}$ = { $R_{\{On\}}$, $R_{\{Radio,Tape\}}$, $R_{\{Stereo\}}$, $R_{\{Play,Pause\}}$, $R_{\{Am,Fm\}}$ };
(2) $\mathcal{R}_0 = \{R_{\{On\}}\}$;
(3) The derived partial order relation: $R_{\{On\}} \succ R_{\{Radio,Tape\}}, R_{\{On\}} \succ R_{\{Stereo\}}$, $R_{\{Radio,Tape\}} \succ R_{\{Am,Fm\}}, R_{\{Radio,Tape\}} \succ R_{\{Play,Pause\}}$;
(4) $\mathcal{C}$ =$\{Stereo \supset On,\ Radio \vee Rage \supset On,\ Am \vee Fm \supset Radio,\ Play \vee Pause \supset Tape\}$.

Recall that local constraints describe local properties to be satisfied by local system specification modules. On the other hand, costraints in a system requirement express global constraints to be satisfied by all system specification modules.

3. System Specifications

In this paper hierarchical state transition systems are considered as formal specifications of reactive systems. At first, a state transition module is defined as the specification of a system requirement module, then a formal specification is defined as the set of state transition modules with the same partial order relation as modules of system requirement. Based on the formal specification, the global state transition system is derived for specifying dynamic features of the entire reactive system.

3.1 State Transition Module

A partial formal specification corresponding to each system requirement module is represented by a state transition module.

Definition 3.1 Let $\mathcal{S} = \langle \mathcal{P}, \mathcal{L}, \succ, \mathcal{L}_0 \rangle$ be a system property structure and $L \in \mathcal{L}$. A *state transition module* over L is a tuple $M_L = \langle Q, \Sigma, O, \rightarrow, q_0, B \rangle$, where

(1) Q is a set of *states* in which atomic propositions in L(in $\mathcal{P}$) are interpreted (partially interpreted);
(2) Σ is a set of *input symbols*;
(3) O is a set of *output symbols*;
(4) $\rightarrow \subset Q \times \Sigma \times O \times Q$ is a *transition relation*;
(5) $q_0 \in Q$ is an *initial* or a *default state*;
(6) B is the *background condition* as a consistent proposition of higher level, i.e., a consistent proposition using atomic propositions in $\bigcup_{L_u \succ L} L_u$. □

The transition relation defines the change of states as input symbols are read. For $(p, a, o, q) \in \rightarrow$, we normally write $p \overset{a/o}{\rightarrow} q$. For simplicity, we just consider transitions inside modules and do not discuss transitions among different modules. *Background condition* specifies conditions by which a state transition module is activated and at the same time *initial state* of the module is reached as one of the current states. In each module, only one state can be the current state. A state may activate multiple state transition modules by satisfaction of background conditions of these modules, and a module can be entered from states in the same or different modules.

A *partial interpretation* I is a partial function $I : \mathcal{P} \rightarrow \{\mathbf{true}, \mathbf{false}\}$, where **true** and **false** are the truth values of propositions. If the truth value of a proposition f under I is defined to be **true**, i.e. $I(f)$=**true**, then we say that I

satisfies f, denoted by $I \models f$. These can be defined inductively as follows:

(1) $I \models A$ ($I \not\models A$) if I is defined on A and $I(A) = \mathbf{true}$ ($I(A) = \mathbf{false}$), where $A \in \mathcal{P}$.
(2) $I \models \neg f$ ($I \not\models \neg f$) if $I \not\models f$ ($I \models f$).
(3) $I \models f \wedge g$ ($I \not\models f \wedge g$) if $I \models f$ and $I \models g$ ($I \not\models f$ or $I \not\models g$).
(4) $I \models f \vee g$ ($I \not\models f \vee g$) if $I \models f$ or $I \models g$ ($I \not\models f$ and $I \not\models g$).

Note that the truth value of a proposition under an interpretation is not always defined since only partial interpretations are concerned.

We assume that for an atomic proposition $A \in \mathcal{P}$ and for a state $q \in Q$ it is pre-defined whether or not A holds (is satisfied) in q if the truth value of A in q is defined. $q \models A$ indicates that the truth value of A in q is defined and A holds in q. Let us define the partial interpretation associated with a state q in a state transition module or a global transition system, denoted by $I(q)$, in such a way that

$$I(q)(A) = \begin{cases} \mathbf{true} & \text{if } q \models A \\ \mathbf{false} & \text{if } q \not\models A \ (q \models \neg A) \\ \text{undefined} & \text{otherwise} \end{cases}$$

for all atomic propositions $A \in \mathcal{P}$. In a similar way, we can define the interpretation $I(\gamma)$ for a consistent conjunction of literals γ. Let $Sat(q) = \{A \mid A \in \mathcal{P}, q \models A\} \cup \{\neg A \mid A \in \mathcal{P}, q \models \neg A\}$.

Proposition 3.1 $q \models f$ iff f is implied from $Sat(q)$, i.e. every interpretation satisfying $Sat(q)$ also satisfies proposition f.

Proof: The proof is by structural induction on propositions f. □

By the completeness of propositional logic, we have that $q \models f$ iff $Sat(q) \vdash f$, f is provable from $Sat(q)$. Two states p and q in M are *logically equivalent* iff $I(p) = I(q)$. A state transition module M is *logically reducible* if there exist distinct logically equivalent states in M. Otherwise, the module is *logically irreducible*. To the rest of this paper, unless stated otherwise, a state transition module means a logically irreducible system. Thus, $p = q$ iff $I(p) = I(q)$ ($Sat(p) = Sat(q)$). By this assumption, note that a state q in a (an irreducible) state transition module M can be equivalently represented as a consistent conjunction of literals of set Q_q, where $q \models A$ ($q \models \neg A$) iff $A \in Q_q$ ($\neg A \in Q_q$).

State transition modules can be derived by synthesis method from the corresponding system requirement modules. A formal specification represented by hierarchical state transition systems can be defined as follows.

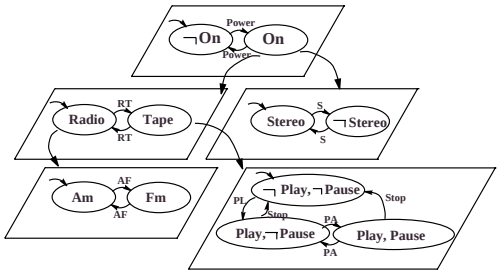

Fig. 3 Hierarchical system specification of the Radio/Tape player.

Definition 3.2 Let $\mathcal{S} = \langle \mathcal{P}, \mathcal{L}, \succ, \mathcal{L}_0 \rangle$ be a system property structure. A *system specification* on $\mathcal{S}$ is a tuple $\mathbf{M} = \langle \mathcal{M}, \mathcal{M}_0 \rangle$, where

(1) $\mathcal{M} = \{M_L | L \in \mathcal{L} \text{ in } \mathcal{S}\}$ is a family of state transition modules;
(2) $\mathcal{M}_0 = \{M_{L_0} | L_0 \in \mathcal{L}_0 \text{ in } \mathcal{S}\}$ ($\subseteq \mathcal{M}$) is a family of initial state transition modules at the topmost level. □

The partial order relation $\succ$ on $\mathcal{L}$ can be applied to state transition modules in $\mathbf{M}$ in the similar way: Let M_1 and M_2 be state transition modules over L_1 and L_2 in $\mathcal{L}$ of $\mathcal{S}$ respectively, $M_1 \succ M_2$ iff $L_1 \succ L_2$ in $\mathcal{S}$.

Example 3.1 Based on the definitions of system requirement of the Radio/Tape player, the system specification represented as hierarchical state transition systems can be synthesized from the system requirement. See section 5 for detailed discussions. The derived hierarchical system specification is illustrated in **Fig. 3**.

As can been seen in the example, there are no inconsistency in the conditions themselves, i.e., pre-conditions, post-conditions, initial conditions, background condition, and constraints are consistent as they are. But, in the synthesis process there may occur contradiction. For example, if we add an extra condition $\{Am \wedge Fm\}$ as a global constraint, then in the resulting global transition system in Fig.4, the state with the interpretation $\{On, Radio, Am, Stereo\}$ and its related transitions are not constructed, this check can be done in the synthesis process. Besides, the synthesis process also checks potential inconsistency via generating the global transition sys-

tem, where global consistency is check

3.2 Global State Transition System

According to the definition of partial order relation and background dependency, there are mainly two kinds of relationships between state transition modules: *hierarchical* and *independent*. Two modules are hierarchical if the modules are related by the application of transitive law of the partial order relation, otherwise they are independent modules. For independent modules, their relationship can be further defined as *parallel* and *sequential*. Two independent modules are parallel if both their background conditions in their least common upper hierarchical module are consistent (i.e., can be satisfied by a state in their least common upper hierarchical module), otherwise the modules are sequential. These kinds of relations can also be extended to a set of state transition modules.

A module is entered when the module's background condition is satisfied by a state of an immediate upper hierarchical module, so a set of parallel modules with background condition satisfied by the same state is activated simultaneously. At this time, all the initial states of these modules are the current states. Correspondingly, activated parallel modules are left at the same time because of a transition in their common upper hierarchical modules.

Let $\mathbf{M} = \langle \mathcal{M}, \mathcal{M}_0 \rangle$ be a given system specification on a system property structure $\mathcal{S}$, and fix to the rest of this section. To simplify the notations for a state transition module M, its set of states and its transition relation are denoted by Q_M and $\rightarrow_M$ respectively. We apply the same notations to other ingredients. The global state and transition are presented by the following constructive definitions.

Definition 3.3 A state p in a state transition module M in $\mathbf{M}$ is a *basic state* if there are no $M^{'}$ in $\mathbf{M}$ such that $M \succ M^{'}$ and $p \models B_{M'}$. □

Example 3.2 *Basic states* are states with no dependent state transition modules. The basic states of the player are $\neg On$, Am, Fm, $Stereo$, $\neg Stereo$, $\{\neg Play, \neg Pause\}$, $\{Play, \neg Pause\}$, $\{Play, Pause\}$.

Definition 3.4 The set of *sub-states* of a state q in a state transition module (or a state transition module M) denoted as $substates(q)$ (or $substates(M)$) is defined inductively:

(1) If q is a basic state, $substates(q)=\{q\}$;

(2) If M is a state transition module, $substates(M) = \bigcup_{q \in M} substates(q)$;

(3) If q is a state in a state transition module M with immediate dependent state transition modules $M^{'}$ such that $M \succ M^{'}$ and $q \models B_{M'}$, $substates(q) = \bigcup_{q \models B_{M'}} substates(M^{'}) \cup \{q\}$. □

Definition 3.5 The set of *global states* of a state q in a state transition module (or a state transition module M) denoted as $gstates(q)$ (or $gstates(M)$) is defined inductively:

(1) If q is a basic state, $gstates(q) = \{\{q\}\}$;

(2) If M is a state transition module, $gstates(M) = \bigcup_{q \in Q_M} gstates(q)$;

(3) If q is a state in a state transition module M with immediate dependent state transition modules $M^{'}$ such that $M \succ M^{'}$ and $q \models B_{M'}$, $gstates(q) = \{\{q\} \cup \{\bigcup_{1 \le i \le n} Q_i) | (Q_1, ..., Q_n) \in \Pi_{q \models B_{M'}} gstates(M^{'})\}\}$, where n denotes the number of $M^{'}$ such that $q \models B_{M'}$ and Π is the *set-theoretic Cartesian product* operation. □

Definition 3.6 The set of *default states* of a state q in a state transition module (or a state transition module M) denoted as $default(q)$ (or $default(M)$) is defined inductively:

(1) If q is a basic state, $default(q) = \{q\}$;

(2) If M is a state transition module with q_0 as the initial state, $default(M) = \{q_0\} \cup default(q_0)$;

(3) If q is a state in a state transition module M with immediate dependent state transition modules $M^{'}$ such that $M \succ M^{'}$ and $q \models B_{M'}$, $default(q) = \bigcup_{q \models B_{M'}} default(M) \cup \{q\}$. □

Sub-states of a state or module are all hierarchical dependent states on the state or module. *Global states* of a state or module are all possible consistent configurations containing this state or a state in this module. *Default state* of a state or module is a element of its *global states* and is the set of initial states of all dependent state transition modules.

Example 3.3 In the example Radio/Tape player, the *sub-states* of state *Radio* is $\{Radio, Am, Fm\}$, the *global states* of state *Radio* is $\{\{Radio, Am\}, \{Radio, Fm\}\}$, the *default state* of state *On* is $\{On, Radio, Am, Stereo\}$.

Definition 3.7 The *global transition system* of a system specification $\mathbf{M} = \langle \mathcal{M}, \mathcal{M}_0 \rangle$ on a system property structure $\mathcal{S}$ is the tuple $\mathbf{Mg} = \langle Q_g, \Sigma_g, O_g, \rightarrow_g, q_{g0} \rangle$, where

(1) $Q_g = \{(\bigcup_{1 \le i \le n} Q_i) \mid (Q_1, ..., Q_n) \in \Pi_{M_0 \in \mathcal{M}_0} \mathit{gstates}(M_0)\}$ is the set of *system global states*, where n denotes the number of M_0 in $\mathcal{M}_0$.

(2) Σ_g is the set of *system input symbols*. $\Sigma_g = \bigcup_{M \in \mathcal{M}} \Sigma_M$.

(3) O_g is the set of *system output symbols*. $O_g = \bigcup_{M \in \mathcal{M}} O_M$.

(4) $\rightarrow_g \subset Q_g \times \Sigma_g \times O_g \times Q_g$ is the *global state transition relation*. $q_g \overset{a/o}{\rightarrow} q_g^{'}$ iff there is a state transition module M of $\mathbf{M}$ such that $q_g^{'} = (q_g \text{ - } \mathit{substates}(q)) \cup \mathit{default}(q^{'})$ for some transition $q \overset{a/o}{\rightarrow} q^{'}$ in M, where $q \in q_g$.

(5) $q_{g0} \in Q_g$ is a *global initial state*. $q_{g0} = \cup_{M_0 \in \mathcal{M}_0} \mathit{default}(M_0)$. □

The *system global states* are componentwise union of the product of *global states* of parallel initial state transition modules. *System input(output) symbols* are *input(output) symbols* of all state transition modules. A *global transition relation* can be defined if there exists a possible transition in a state transition with source state, input and output satisfied and after this transition all the sub-states of the source state of this transition are left and all the default states of the target state of this transition are reached. *Global initial state* is the union of *initial states* of all initial state transition modules.

Example 3.4 The global transition system of the Radio/Tape player can be derived from system specification, as it is shown in **Fig. 4**.

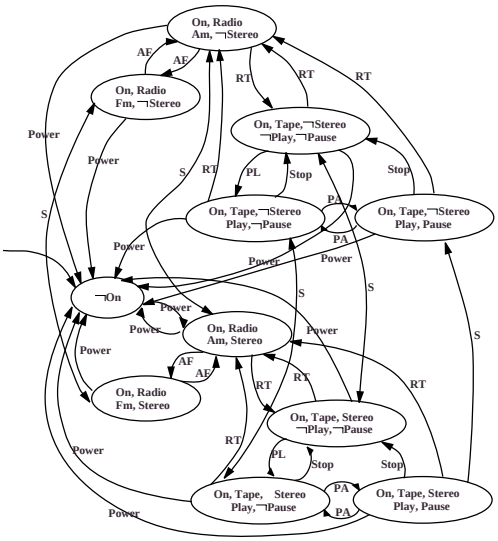

Fig. 4 Global transition system of the Radio/Tape player.

Example 3.4 The global transition system of the Radio/Tape player can be derived from system specification, as it is shown in **Fig. 4**.

4. Soundness and Completeness

In the previous two sections, the user requirements and their formal specifications have been given as system requirements and system specifications, respectively. This section is devoted to describe the relationship between user requirements and system specifications. Thus, the definitions of soundness and completeness of system specifications with respect to system requirements are given. Soundness ensures that a system specification is correct with respect to a system requirement. On the other hand, completeness indicates that the given system specification represents entire properties given as the system requirements. Because of the hierarchical structure of system properties, soundness and completeness definitions of system specifications are based on the definitions on state transition modules.

4.1 Soundness

Definition 4.1 Let $R_L = \langle F, \gamma_0, B, \Sigma, O, \mathcal{C} \rangle$ be a system requirement module over L and $M_L = \langle Q, \Sigma, O, \rightarrow, q_0, B \rangle$ be a state transition module over L. A state transition $t = \langle p \overset{a_1/o_1}{\rightarrow} q \rangle$ in M_L *satisfies* (is *correct w.r.t.*) a function requirement $\rho : f_{in} \overset{a_2/o_2}{\Rightarrow} f_{out}$ in R_L, denoted as $t \models_L \rho$, if:

(1) If $a_1 = a_2, o_1 = o_2, p \models f_{in}$, then $q \models f_{out}$;

(2) The partial interpretations $I(p)$ and $I(q)$ are identical if atomic propositions not occurring in f_{out} are only concerned. □

The condition (1) means the pre-condition and the post-condition of the function requirement must hold in the current state and the next state, respectively. The condition (2) states that for an atomic proposition A independent of f_{out}, $p \models A$ iff $q \models A$. This means that the truth values of independent atomic propositions *w.r.t.* the post-condition remain unchanged through the state transition.

Definition 4.2 A state transition module $M_L = \langle Q_{M_L}, \Sigma, O, \rightarrow, q_0, B_{M_L} \rangle$ is *sound w.r.t.* a system requirement module $R_L = \langle F, \gamma_0, B_{R_L}, \Sigma, O, \mathcal{C} \rangle$, denoted as $M_L \models R_L$,

if:

(1) $I(q_0) = I(\gamma_0)$;
(2) For any transition t in M_L, there is a function requirement ρ in R_L s.t. $t \models_L \rho$;
(3) $I(B_{M_L}) = I(B_{R_L})$;
(4) $\forall f \in \mathcal{C}$ in R_L, $\forall q \in Q_{M_L}$, we have $q \models f$. □

A state transition module is sound with respect to a system requirement module if its initial state or condition and their background conditions have the same partial interpretations, all transitions in the state transition module can be satisfied and all states in the state transition module satisfy constraints specified in the system requirement module.

The soundness of system specification is based on soundness of every state transition module and satisfaction of initial state transition modules and global constraints, as is given in the following definition.

Definition 4.3 Let $\mathcal{S} = \langle \mathcal{P}, \mathcal{L}, \succ, \mathcal{L}_0 \rangle$ be a system property structure. A system specification $\mathbf{M} = \langle \mathcal{M}, \mathcal{M}_0 \rangle$ on $\mathcal{S}$ is *sound w.r.t.* a system requirement $\mathbf{R} = \langle \mathcal{R}, \mathcal{R}_0, \mathcal{C} \rangle$ on $\mathcal{S}$, if

(1) $\forall\ M_0$ in $\mathcal{M}_0$, $\exists\ R_0$ in $\mathcal{R}_0$ s.t. $M_0 \models R_0$;
(2) For all state transition modules M_L in $\mathcal{M}$, $\exists R_L$ in $\mathcal{R}$ s.t. $M_L \models \mathcal{R}_L$;
(3) If $\mathbf{M_g}$ is the global state transition system obtained from $\mathbf{M}$, $\forall c$ in $\mathcal{C}$ of $\mathbf{R}$, $\forall$ global state q_g in $\mathbf{M_g}$, $q_g \models c$. □

4.2 Completeness

Before the definition of completeness, module level and system level simulation relations, called homomorphisms, are first introduced. The homomorphisms specify homomorphic relationship between state transition modules and between system specifications, respectively. A homomorphism is isomorphic if the mapping is a bijection. The concept of standard system is introduced to denote sound and complete system specifications.

Definition 4.4 Let $M = \langle Q, \Sigma, O, \rightarrow, q_0, B \rangle$ and $M' = \langle Q', \Sigma, O, \rightarrow', q_0', B \rangle$ be state transition modules with common sets of input and output symbols and also with the same background condition. A *homomorphism* from M into M' is a mapping $\psi : Q \rightarrow Q'$ such that

(1) $\psi(q_0) = q_0'$.
(2) If $p \stackrel{a/o}{\rightarrow} q$ in M, then $\psi(p) \stackrel{a/o}{\rightarrow}{}' \psi(q)$ in M'.
(3) $p \models f$ implies $\psi(p) \models f$ for all states p in M and for all propositions f. □

The third condition (3) in the above definition can be equivalently relaxed:

(3') $p \models l$ implies $\psi(p) \models l$, for all states p in M and for all literals l.

Definition 4.5 Let $\mathbf{M} = \langle \mathcal{M}, \mathcal{M}_0 \rangle$ and **M'** $= \langle \mathcal{M}', \mathcal{M}_0' \rangle$ be system specifications on a common system property structure $\mathcal{S}$. A *homomorphism* from $\mathbf{M}$ into **M'** is a mapping $\xi : \mathcal{M} \rightarrow \mathcal{M}'$ such that

(1) $\xi(M_0) \in \mathcal{M}_0'$, where $M_0 \in \mathcal{M}_0$.
(2) If $M_1 \succ M_2$ in $\mathbf{M}$, then $\xi(M_1) \succ \xi(M_2)$ in **M'**.
(3) For every M in $\mathbf{M}$, there is a homomorphism ψ from M in $\mathbf{M}$ to $\xi(M)$ in **M'** i.e., $\psi : Q_M \rightarrow Q_{\xi(M)}$. □

If a homomorphism $\xi : \mathcal{M} \rightarrow \mathcal{M}'$ is a bijection, a one-to-one and onto mapping, and each homomorphism $\psi : Q_M \rightarrow Q_{\xi(M)}$ is also a bijection for every M in $\mathbf{M}$, then ξ is called an *isomorphism*. If there is an isomorphism from $\mathbf{M}$ to **M'**, then $\mathbf{M}$ and **M'** are *isomorphic*.

Definition 4.6 Let $\mathcal{S} = \langle \mathcal{P}, \mathcal{L}, \succ, \mathcal{L}_0 \rangle$ be a system property structure. Let $\mathbf{M}$ be a sound system specification on $\mathcal{S}$ with respect to a system requirement $\mathbf{R}$ on $\mathcal{S}$. $\mathbf{M}$ is called *complete* with respect to $\mathbf{R}$ if there is a homomorphism ξ from **M'** to $\mathbf{M}$ for every sound system specification **M'** with respect to $\mathbf{R}$, and there is a homomorphism ψ from $M^{'}$ in **M'** to $\xi(M^{'})$ in $\mathbf{M}$ for every $M^{'}$ in **M'** with the following condition satisfied: For any transition $p \stackrel{a/o}{\rightarrow} q$ in $M^{'}$ and for any system requirement ρ in R of $\mathbf{R}$, $p \stackrel{a/o}{\rightarrow} q \models \rho$ in $M^{'}$ implies $\psi(p) \stackrel{a/o}{\rightarrow} \psi(q) \models \rho$ in $\xi(M^{'})$. □

Definition 4.7 A sound and complete system specification with respect to a system requirement $\mathbf{R}$ on a system property structure $\mathcal{S}$ is called a *standard system* (*model*) of $\mathbf{R}$ on $\mathcal{S}$. □

Recall that the notions soundness and completeness capture the relationship between system requirements and system specifications. Soundness ensures that a system specification is correct with respect to a system requirement. Completeness indicates that the given system specification represents entire behaviors given as the system requirements. So that a standard system can be taken as a faithful realization of a system requirement

5. Synthesis of Formal Specification

The target of synthesis method is to derive a sound and complete system specification as a hierarchical state transition system $\mathbf{M}$ from a given system requirement $\mathbf{R}$. After we state a transformation $\mathcal{T}_m$ from a system requirement module R into a state transition module M, the transformation $\mathcal{T}$ from a system requirement $\mathbf{R}$ to a formal specification $\mathbf{M}$ is defined.

Let us define a state transition module $\mathcal{T}_m(R) = \langle Q, \Sigma, O, \rightarrow, q_0, B\rangle$ from a system requirement module $R{=}\langle F_R, \gamma_{R0}, B_R, \Sigma_R, O_R, \mathcal{C}\rangle$ on L, where

(1) $\Sigma = \Sigma_R$; $O = O_R$; $B = B_R$;
(2) $q_0 = \gamma_{R0}$;
(3) $Q' = \{\gamma \mid \gamma$ is a consistent conjunction of literals in L and $I(\gamma) \models c$ for each $c \in \mathcal{C}\}$ and $Q = \{\gamma \in Q' \mid \gamma$ can be reached from initial condition γ_{R0} by the resulting transition relation $\rightarrow\}$;
(4) For γ and $\gamma' \in Q'$, $\gamma \stackrel{a/o}{\rightarrow} \gamma'$ iff there exists a function requirement $\rho : f_{in} \stackrel{a/o}{\Rightarrow} f_{out} \in F_R$ such that
   (a) $I(\gamma) \models f_{in}$;
   (b) $I(\gamma') \models f_{out}$;
   (c) If A is an atomic proposition not occurring in f_{out}, then $I(\gamma) \models A$ iff $I(\gamma') \models A$.

Recall that the partial interpretation associated with a state γ in $\mathcal{T}_m(R)$ is defined as $I(\gamma)$. It is trivial from the construction that $\mathcal{T}_m(R)$ is irreducible.

Definition 5.1 Let $\mathcal{S} = \langle \mathcal{P}, \mathcal{L}, \succ, \mathcal{L}_0\rangle$ be a system property structure. The system specification $\mathcal{T}(\mathbf{R}) = \langle \mathcal{M}, \mathcal{M}_0\rangle$ on $\mathcal{S}$ is defined from a system requirement $\mathbf{R} = \langle \mathcal{R}, \mathcal{R}_0, \mathcal{C}\rangle$ on $\mathcal{S}$, where

(1) $\mathcal{M} = \{\mathcal{T}_m(R) \mid R \in \mathcal{R}\}$;
(2) $\mathcal{M}_0 = \{\mathcal{T}_m(R_0) \mid R_0 \in \mathcal{R}_0\}$;
(3) $\mathbf{Mg}$ is the global state transition system obtained from $\mathcal{T}(\mathbf{R})$ s.t. $\forall c$ in $\mathcal{C}$ of $\mathbf{R}$, $\forall\ q_g$ in $\mathbf{Mg}$ s.t. $q_g \models c$.

Theorem 5.1 Given a system property structure $\mathcal{S}$, the system specification $\mathcal{T}(\mathbf{R})$ derived from a system requirement $\mathbf{R}$ on $\mathcal{S}$ by $\mathcal{T}$ is a standard system of $\mathbf{R}$ on $\mathcal{S}$.

Proof:

Soundness: This property is clear from the construction of the system specification $\mathcal{T}(\mathbf{R})$. More difficult property is completeness.

Completeness: Let $\mathbf{M} = \langle \mathcal{M}, \mathcal{M}_0\rangle$ be a sound system specification on $\mathcal{S}$ with respect to $\mathbf{R} = \langle \mathcal{R}, \mathcal{R}_0, \mathcal{C}\rangle$ on $\mathcal{S}$. Let us define two kinds of mappings:

(1) a mapping $\xi : \mathcal{M}_{\mathbf{M}} \rightarrow \mathcal{M}_{\mathcal{T}(\mathbf{R})}$. Let M_L be a state transition module over L in $\mathcal{M}$ which satisfies a system requirement module R_L in $\mathcal{R}$. Let us define $\xi(M_L) = \mathcal{T}_m(R_L)$.
(2) a family of mappings $\{\psi_L : Q_{M_L} \rightarrow Q_{\mathcal{T}_m(R_L)} |$ for $q \in Q_{M_L}$, $M_L \in \mathcal{M}$ and for $\gamma \in Q_{\mathcal{T}_m(R_L)}$,$R_L \in \mathcal{R}$, $\psi_L(q) = \gamma$ such that $I(q) = I(\gamma)\ \}$.

According to the construction, the mappings ξ from $\mathcal{M}$ to $\mathcal{M}_{\mathcal{T}(\mathbf{R})}$ and ψ_L from Q_{M_L} to $Q_{\mathcal{T}_m(R_L)}$ with $M_L \in \mathcal{M}$ of $\mathbf{M}$ and $R_L \in \mathcal{R}$ of $\mathbf{R}$ are well defined.

First, we will show that $\psi_L : Q_{M_L} \rightarrow Q_{\mathcal{T}_m(R_L)}$ is a homomorphism from a sound state transition module $M_L{=}M{=}\langle Q_M, \Sigma, O, \rightarrow_M, q_{M0}, B\rangle$ w.r.t. $R_L = T = \langle F, \gamma_0, B_{R_L}, \Sigma, O, \mathcal{C}\rangle$ in $\mathbf{M}$ into $\mathcal{T}_m(R_L) = \langle Q_T, \Sigma, O, \rightarrow_T, q_{T0}, B\rangle$ in $\mathcal{T}(\mathbf{R})$.

(1) It can be easily checked that $\psi(q_{M0}) = q_{T0}$ since $I(q_{M0}) = I(\gamma_0)$ by the soundness of M_L and $q_{T0} = \gamma_0$ by $\mathcal{T}_m$, so $I(q_{M0}) = I(q_{T0})$.
(2) Let $p \stackrel{a/o}{\rightarrow} q$ be any transition in M_L. Suppose $\rho : f_{in} \stackrel{a/o}{\Rightarrow} f_{out}$ is a function requirement in R_L satisfied by this transition. So we have $p \models f_{in}$ and $q \models f_{out}$. By the definition of ψ_L, $I(p){=}I(\psi_L(p))$ and $I(q){=}I(\psi_L(q))$. Thus $\psi_L(p) \models f_{in}$ and $\psi_L(q) \models f_{out}$, and $\psi_L(p) \models A$ iff $\psi_L(q) \models A$ for every atomic proposition A not occurring in f_{out} because $p \models A$ iff $q \models A$ for every atomic proposition A not occurring in f_{out}. Therefore, we have a transition $\psi_L(p) \stackrel{a/o}{\rightarrow}_T \psi_L(q)$ in $\mathcal{T}_m(R_L)$ and this transition also satisfies the same function requirement ρ.
(3) By the definition of ψ_L, $p \models f$ implies $\psi_L(p) \models f$ for all propositions f. Hence, ψ is a homomorphism from M_L into $\mathcal{T}_m(R_L)$.

Then, we will prove that ξ is a homomorphism from a sound system specification $\mathbf{M} = \langle \mathcal{M}, \mathcal{M}_0\rangle$ into $\mathcal{T}(\mathbf{R}) = \langle \mathcal{M}, \mathcal{M}_0\rangle$. This is obvious from the definition of ξ. □

The next theorem ensures that a standard system is unique. Thus, there exists a unique standard system for a system requirement by Theorem 5.1.

Theorem 5.2 A standard system of $\mathbf{R}$ on

a system property structure $\mathcal{S}$ is unique up to isomorphism.

Proof: Suppose $\mathbf{M} = \langle \mathcal{M}, \mathcal{M}_0 \rangle$ is a standard system of $\mathbf{R}$ on $\mathcal{S}$. It is enough to show that $\mathbf{M}$ and $\mathcal{T}(\mathbf{R})$ are isomorphic. Since $\mathcal{T}(\mathbf{R})$ is a standard system by Theorem 5.1.

Let $\xi : \mathcal{M} \rightarrow \mathcal{M}_{T(\mathbf{R})}$ be a homomorphism with a family of homomorphism $\psi_L : Q_{M_L} \rightarrow Q_{\mathcal{T}_m(R_L)}$ for $L \in \mathcal{L}$ in $\mathcal{S}$ whose constructions are given in the proof of Theorem 5.1.

To prove the theorem, we have to show that ξ and ψ_L are bijections (one to one and onto). It is clear from the definition that $\xi : \mathcal{M} \rightarrow \mathcal{M}_{T(\mathbf{R})}$ is a bijection. Then, it is left to show that $\psi_L : Q_{M_L} \rightarrow Q_{\mathcal{T}_m(R_L)}$ are bijections.

Injectiveness (one to one) of ψ_L: Suppose $\psi_L(p) = \psi_L(q)$ for some states p and q in M_L. This means that $I(p) = I(q)$. Thus, we have $p = q$ since we assume that every state transition module is irreducible.

Surjectiveness (onto) of ψ_L: Since both $\mathcal{T}_m(R_L)$ and M_L are standard w.r.t. R_L, there must exist a homomorphism $\psi : Q_{\mathcal{T}_m(R_L)} \rightarrow Q_{M_L}$. Let γ be any state in $Q_{\mathcal{T}_m(R_L)}$. By construction, γ is reachable from the initial state γ_0 and there is a transition sequence:

$$\gamma_0 \overset{a_1/o_1}{\rightarrow} \gamma_1 \overset{a_2/o_2}{\rightarrow} \ldots \overset{a_n/o_n}{\rightarrow} \gamma_n (= \gamma)$$

Let $p_i = \psi(\gamma_i)$ for $i, 1 \leq i \leq n$. Since ψ is a homomorphism we have the following transitions in M_L.

$$p_0 \overset{a_1/o_1}{\rightarrow} p_1 \overset{a_2/o_2}{\rightarrow} \ldots \overset{a_n/o_n}{\rightarrow} p_n$$

We can show that $I(p_i) = I(\gamma_i)$, i.e., $\psi_L(p_i) = \gamma_i$ for all i, by mathematical induction on i. Thus, for p_n in M we have $\psi_L(p_n) = \gamma$.

In the same way, we can also show that for any transition $\gamma \overset{a/o}{\rightarrow} \delta$ in $Q_{\mathcal{T}_m(R_L)}$, there is a transition $p \overset{a/o}{\rightarrow} q$ in M_L such that $\psi_L(\gamma) = p$ and $\psi_L(\delta) = q$.

Hence ψ_L is surjective. □

Note that the notion of logical irreducibility is crucial to prove the theorem. If we drop this condition, Theorem 5.2 does not hold in general.

6. Conclusion

In this paper, we proposed a methodology for the description of system requirements and formal specifications of hierarchical reactive systems. Both system requirement modules and formal specification modules are defined based on the hierarchical structure of system properties. For the study of dynamic behavior of formal specifications, global transition system is introduced. An automatic mechanism called synthesis system is used to derive formal specifications from system requirements, and the soundness and completeness of synthesized formal specifications is also ensured by this process.

Future research work includes specifications of systems with predicate conditions or time constraints and further compositional verification methods for verifying that system specification satisfies both linear-time and branching-time temporal properties. At the present time, we are implementing a support system based on the methodology.

References

1) Harel, D.: Statecharts: A Visual Formalism for Complex Systems, *Science of Computer Programming*, Vol.8, pp.231–274 (1987).
2) Harel, D., Pnueli, A., Schmidt, J. and Sherman, R.: On the Formal semantics of Statecharts, *Proc. First IEEE Symp. Logic in Computer Science*, pp.54–64 (1986).
3) Jahanian, F. and Mok, K.A.: Modechart: A Specification Language for Real-Time Systems, *IEEE Trans. Softw. Eng.*, Vol.20, No.12 (1994).
4) Wagner, F.: VFSM Executable Specification, *Proc. IEEE Int'l Conf. Computer System and Software Engineering*, The Hague, pp.226–231 (1992).
5) CCITT.: SDL: Specification and Description Language, CCITT Z.100 (1988).
6) ISO.: Estelle: A Formal Description Technique based on the Extended State Transition Model, ISO 9074 (1989).
7) Bolognesi, T. and Brinksma, E.: Introduction to the ISO Specification Language LOTOS, in the Formal Description Technique LOTOS, pp.23–73, Elsevier Sci. Pub. (1989).
8) Clarke, M.E., Emerson, A.E. and Sistla, P.A.: Automatic verification of finite state concurrent system using temporal logic, *ACM Trans. on Programming Languages and Systems*, Vol.8, No.2, pp.244–263 (1986).
9) Alur, R. and Yannakakis, M.: Model Checking of Hierarchical State Machines, *Sixth ACM Symposium on Foundations of Software Engineering* (1998).
10) Emerson, E.A.: *Temporal and Modal Logic, Handbook of Theoretical Computer Science*, pp.995–1072, Elsevier Science Publishers B.V. (1990).
11) Song, K., Togashi, A. and Shiratori, N.: Verification and Refinement of System Re-

quirements, *IEICE Trans. on Fundamentals of Elec., Comm. and Comput. Sci.*, E78-A No.11, pp.1468–1478 (1995).
12) Togashi, A., Kanezashi, F. and Lu, X.: A Methodology for the Description of System Requirements and the Derivation of Formal Specifications, *FORTE/PSTV'97*, pp.383–398 (1997).
13) Kanezashi, F., Lu, X. and Togashi, A.: Derivation Method of Formal Specifications from System Requirements, *Journal of Information Processing Society*, Vol.40, No.1, pp.310–321 (1999).
14) Kanezashi, F., Lu, X. and Togashi, A.: Reflection to System Requirements from Formal Specifications, *Foundation of Software Engineering'98*, pp.74–79, Software Science Society of Japan (1998).
15) Ishii, M., Yamada, M., Kanezashi, F. and Togashi, A.: System Simulator for Effective Software Development Environment, *Proc. No.57 National Conference of Information Processing Society* (1998).
16) Lu, X., Kanezashi, F. and Togashi, A.: Diagnosis of System Requirements and Specifications, *Foundation of Software Engineering'97*, pp.39–42, Software Science Society of Japan (1997).
17) Lu, X.: Specification and Verification of Hierarchical Reactive Systems, Research Report in Computer Science, Shizuoka University (1999).

(Received June 14, 2001)
(Accepted December 18, 2001)

Fumiaki Kanezashi was born in 1974. He received the B.E. and the M.E. degrees in Computer Science from Shizuoka University, Japan in 1997 and 1999, respectively. Now he is a student of Doctor's course in Science and Engineering at Shizuoka University, Japan. His research interests include software development environment, software component techniques, and process calculi. He is a member of Information Processing Society of Japan.

Atsushi Togashi was born in 1956. He received the B.E. degree from Yamagata University in 1979, the M.E. and the Dr.Eng. degrees from Tohoku University in Japan, in 1981 and 1984, respectively. After receiving his doctor degree, Dr. Togashi joined the RIEC (Research Institute of Electrical Communication) of Tohoku University, as a Research Associate and as an Associate Professor. Currently, he is a Full Professor of the Department of Computer Science in Shizuoka University in Japan. His research interests include concurrent computation, process calculi, program synthesis based on software component techniques, and new computational paradigms. He is a member of the Information Processing Society of Japan, Japan Society for Software Science and Technology, and ACM.