Integrated Control and Diagnostics in Discrete Event Dynamic Systems with Hierarchical Time-Extended Petri Nets

S. Ramaswamy, K. P. Valavanis

Intelligent Robotic Systems Laboratory, The Center for Advanced Computer Studies, The University of Southwestern Louisiana, Lafayette, LA 70504

Abstract

Hierarchical Time-Extended Petri Nets (H-EPNs) are proposed as a modeling, analysis and simulation tool to study and coordinate real-time system operations including Failure Detection and Identification (FDI). The structure of any hierarchically decomposable system is considered to be a three interactive level hierarchical structure of organization, coordination and execution of tasks [7, 9]. The proposed hybrid model may be used at the organization level for off-line evaluation of different planning scenarios as well as at the coordination level for on-line system control (monitoring the real-time system operations).

1. Introduction

Petri nets (PNs) are a graphical and mathematical modeling tool. The PN representation of a system consists of places and transitions (represented as a circle and rectangle, respectively, in a PN representation), with tokens flowing along the arcs interconnecting them. These tokens are used to simulate the dynamic and concurrent activities within the system. As a mathematical tool PNs are used to describe the behavior of the system they represent as state equations and algebraic equations. An autonomous manufacturing system may be characterized as a discrete event dynamic system [5] where the state changes are primarily derived by the occurrence of internal and external events rather than by a concept of synchronized global clock. Petri net (PN) theory has been applied to various application areas, especially in the domain of discrete event systems.

The complex nature of DEDS dictates the need for the use of structured tools to model, analyze, test, verify, validate and evaluate the system operations. From the PN system modeling point of view, FDI is a model-based approach. Moreover, PNs inherently capture the various asynchronous, sequential and parallel interactions between the various system resources and operations with great ease. In [4], PNs are shown to be useful for the detection of abnormal process behavior, or for the measurement of faults with very low time constants (faults with a low time constant will change the measurement signals to a minimum extent over a time period and therefore, would be undetectable) during the real-time monitoring of power plant systems. In [1] fuzzy PNs have been applied for error recovery in robotic work cells. Farah in [3] emphasizes on PN based error recovery in Intelligent Systems.

The paper is organized as follows: Section two discusses the system structure. Section three introduces the H-EPN basics. Section four presents the four grouping structural properties of the H-EPN model. Section five presents a abstract view of the overall H-EPN model of any system. Section six concludes the paper.

2. System Structure

The system structure is assumed to be a three level hierarchical structure of organization, coordination, and execution of tasks [7, 9], as shown in Figure 1a. Although the definition of the modified coordination level is very similar to the one described in [7], the functions of the components that make up this modified structure differ extensively. The original and modified coordination level structures are shown in Figure 1b and 1c, respectively. The communication paths between the three levels in the original and modified structures is shown in Figure 1d.

The modified topology of the coordination level consists of two distinct vertical layers: *(i)* ***The dispatcher / analyzer***: This layer provides communication between the organization level and the H-EPN based controller, and initiates and oversees the system operations. It contains error analysis and resolution[1] algorithms for previously classified and unclassified errors, and, *(ii)* ***The H-EPN based controller***: The system is represented in terms of a H-EPN model. This (pre conceived) system model is used to oversee the expected system operations (expected normal operations with expected failure situations; see Figure 3). The H-EPN controller is composed of two (horizontal) sub-levels of control: *a*) the Higher Level Control and Coordination (HL), and, *b*) the Lower Level Control and Coordination (LL).

This modified topology allows for related historic data to be used at the coordination level (observing Figure 3, this corresponds to the input for the on-line error analysis algorithms during the system initiation) and this data serves as the basis for the coordination level error resolution algorithms (for unclassified errors). One important implication of the above is that the organization level no longer remains a totally off-line entity, as explained in [7, 9], when error analysis is involved. When historic data is required during the decision making process of the coordination level, access of this data in the shared memory by the coordination level algorithms is allowed.

The HL-LL communication paths indicated in Figure 1 represent the following in an H-EPN design: *(i)* Places

[1] Resolution algorithms refer to simple and fast coordination level algorithms that are used to identify and classify a previously unclassified error. Sometimes, errors thus classified, may be accommodated on-line.

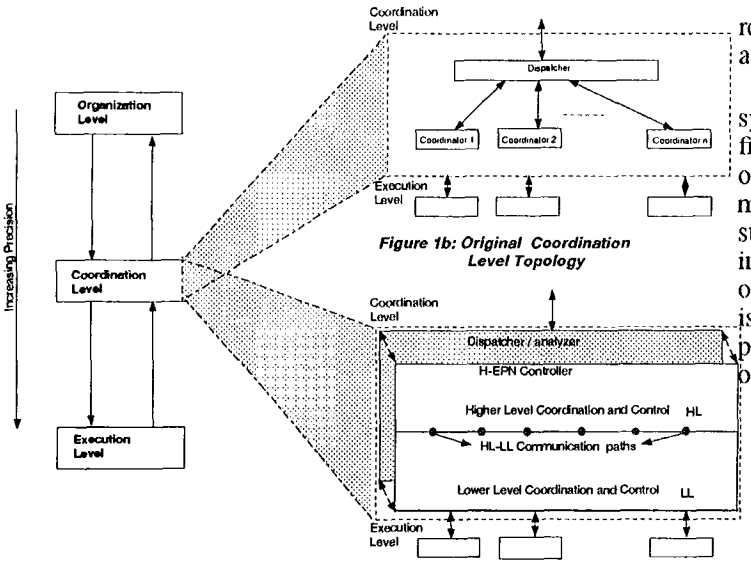
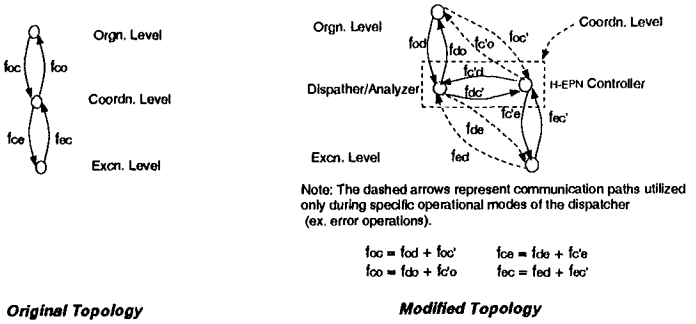


Figure 1b: Original Coordination Level Topology

Figure 1a: System Structure

Figure 1c: Modified Coordination Level Topology



Original Topology

Modified Topology

Figure 1d: Communication paths

Figure 1. ***Block Diagram of the Coordination Level***

that are marked by LL to indicate the need for coordinated error recovery, and transitions that are input to these places, and, *(ii)* Transitions that restore normal operations after the process of error recovery. In this paper, we will concentrate in the structural description of the H-EPNs.

Potential system failures/errors are identified through sensory inputs and are classified, in general, as ***recoverable***, ***non-recoverable*** and ***unclassified errors***. Recoverable errors are further classified as either ***locally recoverable errors*** (LREs) or ***mutually recoverable errors*** (MREs). Non-recoverable errors are ***fatal errors*** (FEs). Unclassified errors (that is, potential errors with no prior knowledge related to them; therefore, with no provisions in the H-EPN system model) are randomly generated errors that occur during real-time operations and may be subsequently reclassified as LREs, MREs, or FEs. Error identification always occurs at the LL layer. Error classification of previously known errors (and thus incorporated within the H-EPN design) occurs during the H-EPN modeling stage. Unclassified errors are analyzed and subsequently classified by the dispatcher/analyzer. LREs are accommodated at the LL layer by resources associated with a single coordinator, while MREs require multiple resources from different coordinators, and are accommodated at the HL layer.

According to the H-EPN based modeling approach, the system operational states may be classified into one of the five states shown in Figure 2a. The advantage of this type of classification is that, depending upon the level of system model detail, the states may be appropriately redefined and studied. For example, a subnet place may be decomposed into a detailed H-EPN net at a lower level, and this process of refinement may continue until a desired level of detail is achieved. A fatal error analysis could be triggered if prolonged periods of a suspended state of operation are observed by the dispatcher/analyzer.

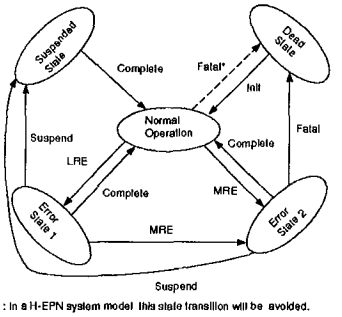


Figure 2(a): State Diagram Representation

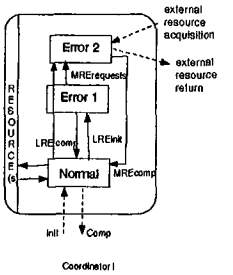


Figure 2(b): Coordinator Operations Classification

Figure 2. ***State Representation***

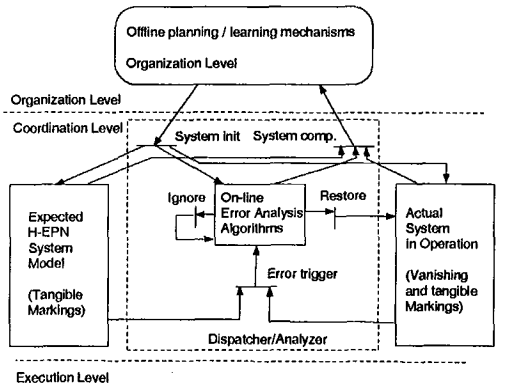


Figure 3. ***A General Error Model***

As the system reaches a steady state of operation, almost all errors are classified as either recoverable or non-recoverable errors. A generalized error model is represented

in Figure 3. The dispatcher/analyzer performs the role of a certifier/verifier of the actual system operations using the expected H-EPN based model as the baseline for such certifications. While the expected H-EPN model is represented by only tangible markings, the actual system may contain both vanishing and tangible markings. Tangible markings are markings corresponding to the system states associated with timed transitions and vanishing markings are markings corresponding to system states associated with immediate transitions. Timed and immediate transitions are discussed in section three. When a discrepancy is observed between the operational states of the actual system and expected system (H-EPN) model, error analysis is triggered. If a "known" soft failure is subsequently observed (that is, a soft failure that has already been incorporated into the H-EPN model), normal system operation is restored on-line (since the process of error recovery has already been specified in the H-EPN model). Thus, over a period of time, all allowable system markings are known. These markings may correspond to actual system errors, or to vanishing markings in the H-EPN system model. In case of other "unknown" system soft failures (observed, but not incorporated into the H-EPN model), on-line restoration (error resolution) is performed if the error recovery process associated with the particular error marking is previously known. In case of hard failures in the actual system, the system halts; then the state of the actual system is reflected in terms of the state (marking) of the expected H-EPN model. This marking is observed by the dispatcher/analyzer and communicated to the organization level. This process simplifies off-line error analysis; that is, it provides a starting point for off-line error analysis algorithms at the organization level. Subsequently, the error recovery process (if it can be incorporated), is included as part of the input information to the dispatcher/analyzer by the organization level. Periodic updates of the H-EPN model may be done to reflect the most recently available information on error identification and recovery. This will ease the bottleneck at the dispatcher/analyzer while overseeing failure-free system operations.

3. The H-EPN Basics

The definition of the H-EPN structure, is a generalization of EPNs described in [8]. The proposed extensions/modifications simplify the modeling and analysis of any hierarchically decomposable system. The generalizations/modifications are discussed in the sequel: *(i)* Five different types of places are defined, *(ii)* Two different types of transitions are defined, *(iii)* Two different zero-weighted arcs are defined. These are the activator and inhibitor arcs. More on the arc extensions may be found in [6, 8] and, *(iv)* Two different types of tokens make up the graphical system model, the solid and dotted tokens.

The different types of places are: *(i)* ***Status place (s)***: This place is similar to a place in the original PN definition, *(ii)* ***Action place (a)***: This place denotes an operation (action) 'a' being performed by the system, *(iii)* ***Decision place (d)***: This place denotes a conflict. A token at this place may uniquely fire any one of the transitions that it enables. This definition of a decision place is different from the definition in [6, 8], where a decision place essentially represents a binary switch, indicating a yes/no situation, *(iv)* ***Subnet place (su)***: This is similar to the subnet place defined in [6, 8]. A H-EPN subnet is a representation of the operations of a subsystem that minimally interacts with other such subsystems. A subnet at the lowest level will perform an elementary operation of the system, that is, the subnet degenerates into an action place. The restriction that the subnet place be a single-input single-output (SISO) place is relaxed, thereby allowing for multiple points of entry to and/or exit from the subnet. This results in greater flexibility in the definition and usage of subnet structures, *(v)* ***Source-Sink place (ss)***: This place represents the origin and end of tokens in the net (this place essentially substitutes the need for separate source and sink places as in [6, 8]). This implies that a complete H-EPN system model could exist between the output and input arcs to this place (see Figure 4c).

The graphical representation of the places in a H-EPN design is shown in Figure 4a. The *ss* place (Figure 4b) is essentially a specialized subnet place of two immediate transitions and four status places. The dotted arcs represent the connections that are used to study system properties during simulation. This structure for the *ss* place implicitly adheres to the property that every source place has a corresponding sink place in the net, thereby allowing for token conservation (that is, all tokens that enter the system, exit the system). This factor is important for ensuring the property of system boundedness.

As an individual entity, a subnet place is not live; its liveness is dictated by the dynamic flow of tokens in the net. Such a property is called "quasi-liveness". Action or status places are always the entry places to and exit places from a subnet. A *ss* place is useful for analyzing the properties of subnet places, since they can be used to study individual subnet properties as shown in Figure 4c. As mentioned earlier, the restriction that the subnet place be a SISO place is relaxed, thereby allowing for multiple points of entry to and/or exit from the subnet. In such a case, every point of entry is associated with a corresponding point of exit. This implies that similar operations performed at different system areas may be grouped together as a single subnet place. The status of the initiation of such an operation (represented by a subnet place) is maintained by a conjugate place. Thus, although a subnet place may lead to the firing of different transitions after the completion of associated operations, the use of the conjugate place enables the firing of only one transition.

The action, decision and subnet places have associated discrete times. The time associated with a subnet place is a variable entity (but it is restricted to a set of values) since a subnet place may have different paths between its entry and exit points, each of which may have different sets of associated action or decision places. Thus the action, decision and subnet places are "timed places".

The transition extensions correspond to the "timing" and "event-driven behavior" of the transitions (see Figure 5a). The firing of transitions is assumed to be triggered by the occurrence of events. A transition will fire if and only if tokens exist in all of its input places and an event (or a set of events) associated with the transition occurs. Events are

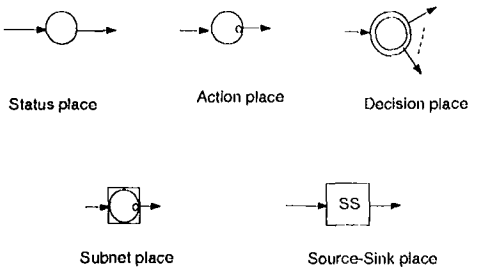


Figure 4(a)

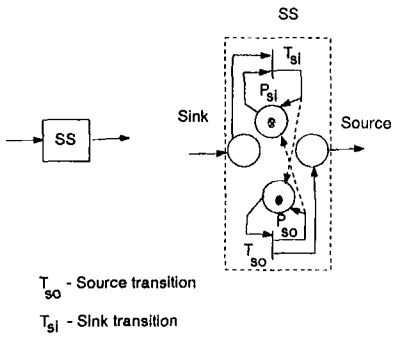


Figure 4(b)

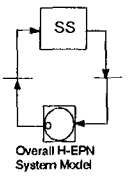


Figure 4(c)

Figure 4. ***Places in an H-EPN***

assumed to be caused by sensory inputs, the beginning or completion of an operation, etc. Two classes of transitions are defined: *(i)* ***Internally Driven Transitions*** (IDT), that are caused by internal events, and, *(ii)* ***Externally Driven Transitions*** (EDT), that are caused by external events.

Events that can fire a transition may be classified as: *(i)* Internal Events: Internal events are more of a predictable nature and can be quantified by means of stochastic or discrete firing times; they may be due to the beginning or completion of an operation, or, sensory input data, *(ii)* External Events[2]: External events occur sporadically; they may occur at random and cannot be quantified; they may be due to the occurrence of a random external event required to fire a transition. This event could be generated by the dispatcher/analyzer and thus not quantified in the H-EPN model, or, *(ii)* An operator interruption.

All transitions have three distinct discrete time periods (the associated values may be derived by corresponding constant or variable functions) associated with them. These are:

i) ***Enable time*** (E): The enable time of a transition t_i, is the time period that starts when all input places to the transition are marked and the transition can possibly fire, and ends when the associated events occur. IDTs have a universal event, *u*, associated with the enable time, while EDTs have in addition to *u*, a randomly occurring event, *r*, associated with the enable time[3]. When the events occur, the corresponding transition is considered "enabled". *u* can be considered as a universal event that occurs periodically, and at every such occurrence, all IDTs that have satisfied the token constraints in their input places are enabled. The default value of E is '0'. The concept of reserved tokens [2] is non existent, that is, tokens are not reserved for the firing of a particular transition once it is enabled, and all enabled transitions have an equal opportunity of firing. Thus, even if an event *x*, corresponding to the enable time of some transition t_i occurs, it does not imply that t_i is the only transition that can possibly fire.

ii) ***Holding time*** (H): The holding time of a transition is the maximum time period associated with any input place to the transition. This indicates the time associated with the operation performed by an input place. Depending upon the input places p_i to the transition t_i ($p_i \in P$ and $P = \bullet(t_i)$), there exist two types of default values for the holding time of transition t_i. These are:

a. H = 0; this is the case when there does not exist a subnet (su), action (a) or decision (d) place, as input places to the transition:

$$\neg\exists\, p_i,\ p_i \in P_k : k \in \{a, d, su\}$$
$$i.e.,\ P_k \cap \bullet(t_i) = \emptyset$$

b. H = τ; this is the case when there exists at least one input place to the transition, which is either a subnet, action or decision place, τ being the maximum time associated with some input place to the transition (see Figure 5b):

$$\exists\, p_i,\ p_i \in P_k : k \in \{a, d, su\}$$
$$i.e.,\ P_k \cap \bullet(t_i) \neq \emptyset$$

It is obvious that the value of H is a run time variable depending on the input set, P, of timed places. The justification for this type of definition for H is that, if error recovery is built into a subnet place, say S_p, then the time τ_p associated with this place is a dynamic variable. As stated earlier, this is because there can exist multiple traversal paths between the subnet entry and exit points, traversing through different sets of timed places.

iii) ***Firing time*** (F): This time denotes the waiting time period for the occurrence of a firing event[4], *f*, before a transition fires, thereby removing tokens from its input places and adding tokens to its output places. The default value of F is '1'. Event *f* occurs periodically, and

[2] Events that are external to the system, but quantifiable and included in the H-EPN model, are not considered to be external events.

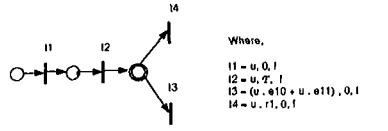
[3] Not all EDTs are driven by a single random event. Thus, $r \in R$, where *R* is the set of all possible random events. Some random events can be generated by the dispatcher/analyzer.

[4] *f* is also a periodic event like *u*. We distinguish it as *f* for ease of understanding.

can occur at different times for a particular transition depending on the value of H, either at time instant 1 or τ + 1. When there is more than one input place (each associated with different operations) to a transition, then, f can be a composite[5] event, which occurs only if any corresponding events associated with the completion of all such operations have occurred.

Thus, every transition (see Figure 5c) in the H-EPN model can be represented by a 3–tuple, (x, y, z), where x and z have associated events[6] (and thus can be represented as event compositions) and y represents the holding time (0 or τ).

Immediate transitions (IT) are not defined as a separate class, and are assumed to be a special case of IDTs but with the default enable, holding (H = 0) and firing times (driven by only events u and f) associated with them. In the H-EPN model, IDTs are associated with all types of places while EDTs are associated only with special status or decision places that initiate a transition to a state of the system involving some external random variable.



Events	Description	Associated Trans. Type
u	universal event	IDT/EDT
f	firing event	IDT/EDT
e3	composite event	IDT/EDT
e10	Payload variation	IDT
e11	Object dropped	IDT
r1	Interrupt	EDT

Figure 5(a) - Example 1

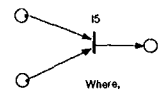


Figure 5(b) Example 2

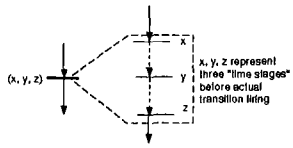


Figure 5(c) Transition timing structure

Note: 1. e10 and e11 are sensor driven events.

2. Composition of events is always represented by boolean operators. Thus, + and ., represent boolean operations AND and OR respectively.

Figure 5. ***Time, Events, and Transitions***

In the H-EPN model of the system, an enabled transition implies that events associated with the transition are expected to occur within the specified time limits. This time limit is the actual firing time of the transition. If the transition, t_i, does not fire within this firing time limit, it must be due to one of the following reasons:

i) The transition t_i, is no longer enabled: This implies that an event e_j, associated with transition t_j, that belongs to the same conflict set Λ_s, has occurred before the event e_i associated with t_i, and,

ii) The transition t_i, is still enabled but the event associated with the transition did not occur. This indicates an error condition that prevented the occurrence of the event and a short on-line error diagnosis is triggered.

Two classes of tokens are visually distinguished: ***solid***, and ***dotted*** tokens. Solid tokens are tokens that are an essential part of the actual system description. Dotted tokens (transient tokens) are tokens that are generated during system operation. Dotted tokens ensure the quasi-liveness of the H-EPN model. These are not part of either the model or the actual system; they come into existence during system operation (either through the *ss* places or through intermediate net operations), thereby marking initially unmarked places. They are useful for simplifying the net analysis (subnet reachability analysis, simulation, etc. since they dictate the property of liveness).

4. The H-EPN Structure

4.1. Hierarchical Structure of Places and Transitions

This section discusses the grouping of places and transitions in an H-EPN model of the system coordination level. Places and transitions are grouped into four different categories depending upon the state of the system they represent, namely the normal state of operation and the error states. Figure 6a gives a detailed representation of this classification. Then the essential properties of the grouping are stated.

Let P_K and T_K represent all the places and transitions in an H-EPN model of the overall system operations. Then, the H-EPN controller at the coordination level is a representation:

$$H - EPN = (P_K, T_K, I, O)$$

where I and O denote the input and output functions as in the case of an ordinary PN. P_K and T_K represent the set of places and transitions that make up the net. From a system description point of view, this set of places P_K may be redefined to be a union of four subsets of places P_{ei}, where i = {0, 1, 2, 3}. These four subsets of places depend on the various system modeling factors illustrated in Figure 6b: *(i)* P_{e0} are places associated with the normal operations of the system (excluding LRE operations), *(ii)* P_{e1} are places associated with the operations of the system when an LRE occurs, *(iii)* P_{e2} are places that are associated with the operations of the system when an MRE occurs, and, *(iv)* P_{e3} are places associated with fatal errors.

Similarly, the transitions in the H-EPN model may be defined to belong to one of four subsets of transitions T_{ei}, where i = {0, 1, 2, 3}. Again, these four subsets of transitions depend on the factors illustrated in Figure 6b: *(i)* T_{e0} are transitions associated with the normal operations of the system, *(ii)* T_{e1} are transitions associated with the operations of the system when an LRE occurs or the transitions that lead to this state from a normal state of operation, *(iii)* T_{e2} are transitions that are associated with the operations of the system when an MRE occurs; transitions that lead to this state from a normal state of operation or from an error state 1 belong to this group, and, *(iv)* T_{e3} are transitions associated with fatal errors (meaning that these may lead to a state of deadlock); transitions that lead to this state from a normal state of operation or the two error states belong to this group.

---

5 The boolean operators (ex. AND, OR) are used to represent event compositions.

6 Events may be of a composite nature, that is, x and z may be a composition of a set of atomic events, from which at least one element of (x, z) is (u, f) in case of IDTs and $(u.r, f)$ in case of EDTs, respectively (see Figure 5).

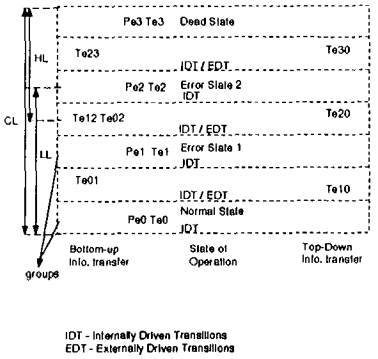


Figure 6(a)

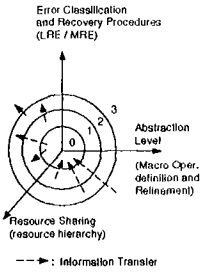


Figure 6(b)

Figure 6. ***Classification of Places and Transitions***

Furthermore, transition groupings T_{e1}, T_{e2}, and T_{e3} contain three disjoint subsets of transitions: *(i)* Transitions that have their input and output places at the same group (generally referred to as T_{ei} itself), *(ii)* Transitions that have at least one of their input places at a lower group and one of their output places at the highest group considered (either 1, 2, or, 3; see Figure 6b), and, *(iii)* Transitions that have at least one of their input places at a higher group (either 1, 2, or 3) and one of their output places at the lowest group (0; see Figure 6b).

We distinguish the last two types of transitions (which are of the form T_{eij}, and are a subset of transitions T_{ej}), to represent changes in the system operational states. Transitions of the form T_{e21}, T_{e32}, T_{e31} are not considered in the design, because it is assumed that the error recovery operations return the system to a normal state of operation. Transitions of the form T_{e03} and T_{e13} are possible but are prohibited to allow failure information to be propagated in a hierarchical manner

Some transitions of the form T_{eij} are EDTs. Therefore, the firing of these transitions is possible only when a random event *r* occurs. Figure 6a represents the grouping of places and transitions within the H-EPN model of the coordination level.

Transitions T_{e01}, T_{e02}, T_{e12}, etc., are a group of transitions (called blocked transitions) that might lead to an LRE or an MRE, but may also lead to different error recovery procedures (this might correspond to different entry points in some subnet place at a higher level, or to entirely different subnets). They can either be IDTs or EDTs depending upon the system details.

This classification does not have a separate grouping of places and transitions that represent a suspended state of operation. By definition, a suspended state of operation for an individual coordinator, includes places and transitions that are associated with the coordinator whose associated resource(s) is used by another coordinator during an MRE error recovery. Thus, the set of places and transitions that represent a suspended state may possibly span groups 0, 1, and 2 in the classification.

4.2. H-EPN Structural Properties

Consider that:

i) P_K and T_K denote sets of places and transitions,
ii) p_k and t_k denote an individual place and transition,
iii) Places with a suffix *d* denote a decision place, *a* denote an action place, *s* denote a status place, *ss* denote a source-sink place and *su* denote a subnet place,
iv) Transitions with a suffix of *idt* and *edt* denote internally and externally driven transitions respectively,
v) $\rho(P)$ denotes the cardinality of the set P,
vi) $\bullet(t_l)$ denotes the set of places that are input to the transition t_l, $t_l \in T_K$, and,
vii) $(t_l)\bullet$ denotes the set of places that are output to the transition t_l, $t_l \in T_K$.
viii) $p^p{}_{sj}$ denotes the pth status place at group j.

The hierarchical structural properties of the place and transition (top-down) groupings (refer to Figure 6a) are:

Property 1: The overall H-EPN model consists of all the different types of places and transitions:

$$P_K = \bigcup_{i=\{d,a,s,ss,su\}} P_i = \bigcup_{i=\{0,1,2,3\}} P_{ei}$$

$$T_K = \bigcup_{i=\{idt,edt\}} T_i = \bigcup_{i=\{0,1,2,3\}} T_{ei}$$

where,

$$\rho(P_K) = \rho(P_d) + \rho(P_a) + \rho(P_s) + \rho(P_{ss}) + \rho(P_{su})$$

$$= \rho(P_{e0}) + \rho(P_{e1}) + \rho(P_{e2}) + \rho(P_{e3})$$

$$\rho(T_K) = \rho(T_{idt}) + \rho(T_{edt})$$

$$= \rho(T_{e0}) + \rho(T_{e1}) + \rho(T_{e2}) + \rho(T_{e3})$$

Property 2: Actions are well defined at the lowest level of system detail, that is, all subnet places are refined into their constituent status, decision and action places:

$$\forall j \in \{1,2\}$$

$$P_{ej} = \bigcup_{k\in\{d,a,su,s\}} P_{ekj}$$

$$\forall j \in \{0\}$$

$$P_{ej} = \bigcup_{k\in\{d,a,s\}} P_{ekj}$$

Property 3: A subnet place at a given group consists of places and transitions defined at the same or lower

groups:

$$\forall\, i \in \{0,1,2\},\ j \in \{1,2\}\ and\ i \le j,$$
$$p^{p}_{suj} = \bigcup_{i} P'_{eki} \cup T'_{ei},\ where\ k \in \{d,a,su,s\}$$
$$where,$$
$$P'_{eki} = P_{eki} - \{p^{p}_{suj}\}$$

Property 4: Places fall uniquely within the groupings and there does not exist the same set of places that can occur at two different groups:

$$\forall i,j \in \{0,1,2,3\},$$
$$P_{e_i} \cap P_{e_j} = \begin{Bmatrix} \phi & i \neq j \\ P_{e_i} & i = j \end{Bmatrix}$$

Property 5: Transitions that fall within two groupings are those that have their input and output places in between these two groups, subject to the validity of properties 5a or 5b:

$$\forall i,j \in \{0,1,2,3\},$$
$$T'_{ei} \cap T'_{ej} = \begin{Bmatrix} T'_{eij} & i \neq j \\ T'_{ei} & i = j \end{Bmatrix}$$

Property 5a. There exists at least one input place p_r at group i and at least one output place p_s at group j to such transitions (of the form T_{eij}, $i < j$, as shown in Figure 6) in a bottom-up information transfer within the system model. p_r is either an action or decision place at group i and p_s is either status, action, decision or source-sink places at group j. Places that belong to groups m, $((m < i) \vee (m > j))$, cannot be either input or output places to transitions of the form T_{eij}. All other places that belong to the groups m', $i \le m' \le j$, can be either input or output places to such transitions. That is:

$$\forall\, t_l \in T'_{eij},\ i,\ j,\ m \in \{0,1,2,3\},$$
$$i < j,\ k \in \{d,a,su,s,ss\},$$
$$((m < i) \vee (m > j))$$
$$\exists\, p_r,\ p_r \in \bullet(t_l) :\ (p_r) \in (P_{eai} \vee P_{edi})$$
$$\exists\, p_s,\ p_s \in (t_l)\bullet :$$
$$(p_s) \in \begin{Bmatrix} (P_{esj} \vee P_{eaj} \vee P_{edj})\ if\ j = \{0,1,2\} \\ (P_{essj}) \qquad if\ j = 3 \end{Bmatrix}$$
$$\neg\exists\, p_t,\ p_t \in (t_l)\bullet \vee p_t \in \bullet(t_l) :\ (p_t) \in P_{ekm}$$

Property 5b. There exists at least one input place p_r at group i and at least one output place p_s at group j to such transitions (of the form T_{eij}, $i > j$, $j = 0$, as shown in Figure 6) in a top—down information transfer within the system model. p_r is either source-sink, action, decision, or status places at group i, and p_s is either a status or action place at group j. Places that belong to groups m, $m > i$, cannot be either input or output places to transitions of the form T_{eij}. All other places that belong to the groups m', $i > m' > j$, can be either input or output places to such transitions. That is:

$$\forall\, t_l \in T'_{eij},\ i,m \in \{1,2,3\},$$
$$j = 0,\ k \in \{d,a,su,s,ss\},\ m > i,$$
$$\exists\, p_r,\ p_r \in \bullet(t_l) :$$
$$(p_r) \in \begin{Bmatrix} (P_{esi} \vee P_{eai} \vee P_{edi})\ if\ i = \{1,2\} \\ (P_{essi}) \qquad if\ i = 3 \end{Bmatrix}$$
$$\exists\, p_s,\ p_s \in (t_l)\bullet :\ (p_s) \in (P_{esj} \vee P_{eaj})$$
$$\neg\exists\, p_t,\ p_t \in \bullet(t_l) \vee p_t \in (t_l)\bullet :\ p_t \in P_{ekm}$$

The above properties reflect the H-EPN controller structure as seen by the dispatcher/analyzer. When the dispatcher/analyzer functions as a certifier/verifier of the overall system operations, all actions performed at the LL sublevel are not "visible" to the dispatcher. However, when error analysis is triggered, the dispatcher selectively checks subnets (and their corresponding lower level nets) until the pertinent error marking is reached. Once this marking is identified, error analysis begins. This essentially implies that the context sensitive nature of subnet initiations and the reduced number of subnet reachability markings simplify the process of error integration.

5. A Generalized H-EPN Structure

Figure 7 represents an abstract view of the H-EPN model of the coordination level with failure accommodation, for any hierarchical system. This is the way in which any H-EPN model is organized. Figures 7 and 8 can be used as the basis for the derivation of the H-EPN model of any system. These structures can be used to identify operations that are jointly performed by the various coordinators and their interactions. These two figures illustrate details that have been spread out throughout all the previous figures.

Every coordinator has associated operations, and all such operations have associated places and transitions that fall into the three groupings as shown in Figure 6 (normal, error state 1 and error state 2). When the coordinator begins operations it operates in its normal state. When LRE is detected (LRE_{init}), then the coordinator performs operations associated with the error state 1, and, when the LRE is accommodated (LRE_{comp}) it returns back to its normal state. If the LRE is not accommodated at error state 1, or if an MRE occurs, then the coordinator performs recovery operations associated with error state 2. Such operations need the support/use of resources associated with other coordinators

and the coordinator is "suspended" until these resources are available. The transfer of resources between two coordinators (from coordinator j to coordinator i) is represented by the dotted structure "X", called the resource transfer structure, as shown in Figure 7.

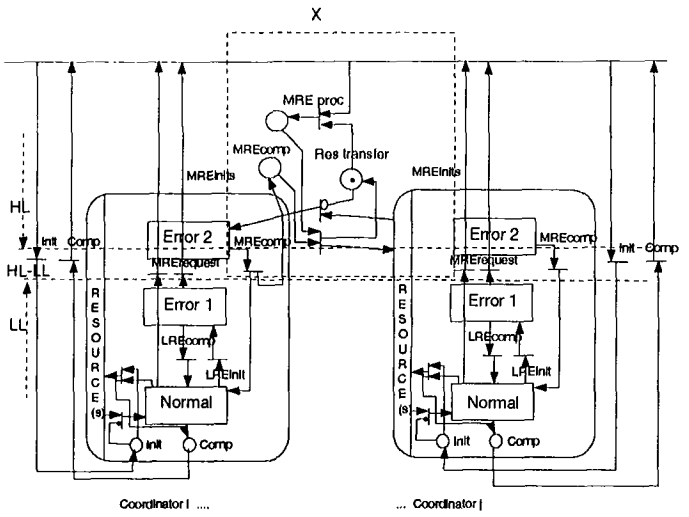


Figure 7. ***A General H-EPN Model Structure***

HL MESH	Coord.1	Coord.2	...	...	Coord.n
Coord.1	-	X	...	...	-
Coord.2	X	-	...	...	X
...		-	-		-
...	X	-	...	-	-
Coord.n	—	X	...	...	-

Figure 8. ***A Coordinator's Mesh***

The table in Figure 8 indicates the interactions between the various coordinators and the directional flow of resources when an MRE occurs. This matrix is application specific. This matrix structure C_x, is called the coordinators' mesh. An entry, X, in the matrix C_{ij} represents a system resource (normally operating under coordinator C_i) used during a MRE error handling state in coordinator C_j. *The existence of an entry for C_{ij} does not mean the existence of a corresponding entry for C_{ji}.* An "empty" entry in the matrix for both C_{ij} and C_{ji} represents mutually independent coordinators C_i and C_j. An entry in C_{ij} of the form $(X_p + ... + X_q)$, denotes that resources $X_i \in C_j$ (some or all of them) are used in the MRE recovery process in C_i.

6. Conclusions

H-EPNs have been proposed for the modeling and analysis of DEDS. A new two layer, two sub-level topological structure has been proposed and explained for describing the functionality of the coordination level of a hierarchical system. This structure can be used for developing a model-based approach to systems development that includes sensor-based on-line error identification and recovery and a supervisory controller for the monitoring of real-time system operaions. A functional grouping of the places and transitions that make up the H-EPN design of any hierarchical system has been defined and discussed from a failure diagnostic point of view.

References

[1] Cao, T., and Sanderson, A. C., "Sensor-Based Error Recovery for Robotic Task Sequences", *Proceedings of the IEEE International Conference on Robotics and Automation*, 1992.

[2] David, R., and Alla, H., *"Petri Nets and Grafcet: Tools for Modeling Discrete Event Systems"*, Prentice–Hall, New York, 1992.

[3] Farah, J. J., *"The Planning Coordinator: A Design Architecture for Autonomous Error Recovery and Online Planning of Intelligent Tasks"*, Tech. rep. 134, RPI, Troy, NY.

[4] Prock, J., "A New Technique for Fault Detection Using Petri Nets", *Automatica, Vol. 27*, No. 7, 1991.

[5] Ramadge, P. J., and Wonham, W. M., "The Control of Discrete Event Systems", *Proceedings of the IEEE, Vol. 77*, No. 1, Jan. 1989.

[6] Ramaswamy, S., and Valavanis, K. P., "Modeling, Analysis and Simulation of Failures in a Materials Handling System with Extended Petri Nets", *IEEE Transactions on Systems, Man and Cybernetics, Vol. 24*, No. 9, Sept 1994.

[7] Saridis, G. N., "Intelligent Robotic Control", *IEEE Transactions on Automatic and Control, Vol. AC-28*, No. 5, 1983.

[8] Valavanis, K. P., "On the Hierarchical Modeling Analysis and Simulation of Flexible Manufacturing Systems with Extended Petri Nets", *IEEE Transactions on Systems, Man and Cybernetics, Vol. 20*, No. 1, January/February 1990, 94–110.

[9] Valavanis, K. P., and Saridis, G. N., *"Intelligent Robotic Systems: Theory, Design and Applications"*, Kluwer Academic Publishers, 1992.