Proceeding of the 2004 American Control Conference
Boston, Massachusetts June 30 - July 2, 2004

A Methodology for Detecting Routing Events in Discrete Flow Networks

Humberto E. Garcia and Tae-Sic Yoo {garcia,tyoo}@anlw.anl.gov
Systems Modeling, Analysis, and Control Group, Argonne National Laboratory

***Abstract*— A theoretical framework for formulating and implementing model-based monitoring of discrete flow networks is discussed. Possible flows of items are described as the sequence of discrete-event (DE) traces. Each trace defines the DE sequence(s) that are triggered when an entity follows a given flow-path and visits tracking locations distributed within the monitored system. Given the set of possible discrete flows, a possible-behavior model -an interacting set of automata- is constructed, where each automaton models the discrete flow of items at each tracking location. Event labels or symbols contain all the information required to unambiguously distinguish each discrete flow. Within the possible behavior, there is a special sub-behavior whose occurrence is required to be detected. The special behavior may be specified by the occurrence of routing events, such as faults. These intermittent or non-persistent events may occur repeatedly. An observation mask is then defined, characterizing the actual observation configuration available for collecting item tracking data. The analysis task is then to determine whether this observation configuration is capable of detecting the identified special behavior. The assessment is accomplished by evaluating several observability notions, such as detectability and diagnosability. If the corresponding property is satisfied, associated formal observers are constructed to perform the monitoring task at hand. The synthesis of an optimal observation mask may also be conducted to suggest an appropriate observation configuration guaranteeing the detection of the special events and to construct associated monitoring agents. The proposed framework, modeling methodology, and supporting techniques for discrete flow networks monitoring are presented and illustrated with an example.**

I. INTRODUCTION

The ability to monitor and track the flow of items or entities within a system in an effective, nonintrusive manner has significant implications in many critical applications including item/material tracking (including the tracking of nuclear material and radioactive sources), item movement violation detection, operations accountability, network security, networked manned and unmanned systems, mission planning, mission execution monitoring, operations safety, operations security, and nuclear safeguards. Assuring item traceability can in fact be crucial in the establishment of many industries (e.g., [2]). However, monitoring systems are seldom designed and instrumented to assure their inherent system properties regarding item traceability. Rather, observational requirements are often fitted to the monitored systems a posteriori. It is desirable to develop a behavioral analysis formalism, with associated formal methods and tools, for designing monitoring agents capable of detecting identified special behaviors. By designing for discrete flow observability as an intrinsic system property, detection of special behaviors is improved in detectability capability, information management, and time response.

Behavioral analysis of discrete event systems (DES) is an active area of research. One relevant field is failure analysis, in which special events are identified as faults. Recently, significant attention has been given to fault analysis; see for example [1,3,5-9,11]. The definition of diagnosability based on failure-event specifications was first introduced in [7]. The notion of diagnosability introduced in [7, 8] characterizes single time detection capability of monitoring agents. Some variations to the initial definition in [7, 8] have been proposed recently. Failure states are introduced in [11] and the notion of diagnosability is accordingly redefined. However, these methodologies are not adequate in the context of discrete flow networks, where routing events may occur repeatedly and need to be reported repeatedly as well. In order to capture the repeatable nature of special events, several efforts have been reported recently. The issue of diagnosing repeatedly occurring faults was first studied in [6]. The notion of $[1, \infty]$-diagnosability is introduced in [6], along with a polynomial algorithm for checking it. However, the time complexity of the algorithm provided in [6] for checking this notion is $O(|X|^6 \times |\Sigma|^2)$ and $O(|X|^4 \times |\Sigma|^2)$ for nondeterministic and deterministic behaviors, respectively, on the number of system states X and the cardinality of the system event set , which severely restricts its applicability. To improve this complexity, an algorithm for checking $[1, \infty]$-diagnosability is introduced in [9] with the reduced complexity of $O(|X|^5 \times |\Sigma|^2)$ and $O(\min(|X|^5, |X|^3 \times |\Sigma|^2))$ respectively for those behaviors. For the problem of designing observation configurations, efforts reported in [3-5, 10] are closely related. The problem of selection of optimal set of sensors is studied in [3-5] that are sufficient yet minimal. The NP-completeness of this sensor selection problem is shown in [10]. This paper builds upon the above efforts to introduce a methodology for detecting routing events in discrete flow networks. Besides the overall description here provided, further technical details, including formal definitions and algorithms, may be found in [3] and [9].

II. PROBLEM STATEMENT AND EXAMPLE

A. Problem Statement

The objective is to monitor item/entity motions and detect special behaviors within the possible behavior of the monitored system by analyzing observable event traces. The possible system behavior is divided into two mutually

0-7803-8335-4/04/$17.00 ©2004 AACC

exclusive regions, namely, the special behavior of interest (which is needed to be detected) and the ordinary behavior (which does not need detection). When the special behavior is specified by events, the task of behavioral analysis is to monitor the system behavior and report the occurrence of any special event (i.e., detection), identify its type (i.e., diagnosis) and count the number of occurrences. The intermittent or non-persistent occurrence of special events is possible. In order to improve information management and reduce information cost, the design goal is to construct a monitoring observer with a detection capability that relies on not only observed tracking measurements but also on recorded knowledge built from continuous system observation.

B. Working Example

To illustrate the developed methodology, the example shown in Fig. 1 is used throughout this paper.

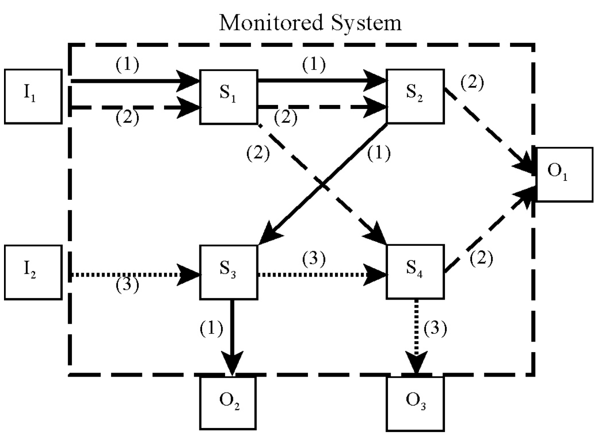

Fig. 1. Monitored system showing normal flows

It consists of a monitored system with two input ports, I_1 and I_2, four internal stations, S_i, $i = 1, 2, 3,$ and 4, and three output ports, O_1, O_2, and O_3. This system may represent a processing facility, a communication network, an air-traffic region, or a battlespace. Item flow-paths are specified by the sequence of input ports, internal tracking stations, and output ports that should be visited by in-transit items. The special behavior is in-turn specified with item movements that violate normal routing requirements. Three authorized routes, (1), (2), and (3) are identified in Fig. 1. An item following route (i) is said to be of type (i). A given item flow specifies the sequence of input ports, internal tracking stations, and output ports that items/entities should visit under its domain. Under route (1), an item should enter the monitored system through the input port I_1, move sequentially to locations S_1, S_2, and S_3, and finally exit through the output port O_2. Under route (2), an item should enter through the input port I_1, move to location S_1, move to either location S_2 or S_4, and finally exit through the output port O_1. Under route (3), an item should enter the monitored system through the input port I_2, move sequentially to locations S_3 and S_4, and finally exit through the output port O_3. Multiple in-transit items may be present within the system, with no restriction on their type. Multiple flows of items can thus exist concurrently at any time. To simplify the description of the methodology, it is assumed, without loss of generality, that each tracking station has a buffer capacity of one item.

III. PROPOSED METHODOLOGY

The proposed approach is to first construct formal descriptions of identified entity flows possible in the system, observational requirements, and observational constraints. Given these descriptions, observational configurations and associated algorithms for data integration and analysis can be systematically found that optimize given observational criteria. To formalize item flows, a model G must be constructed defining how system states change due to event occurrences. To formalize observational requirements, two design elements must be specified, namely, the set of special events S requiring detection and the intrinsic observability property P (i.e., detectability or diagnosability) regarding S. To formalize observational constraints, a cost functional C should be included indicating the costs associated with observation device types and locations. The cost structure of C may be formulated as a partially ordered set or any other topological ordering that better defines observational preferences and constraints. Given G, S, P, and C, the design task is to compute an observational configuration or observation mask M that guarantees P of S with respect to G, while optimizing C. This mask M defines an underlying observational configuration that specifies the locations and types of observational devices required to assure the observability of special routing events. After a suitable observation mask M has been computed, the implementation task is to construct an observer O that will guarantee the P of S by observing G via the observation mask M. In practice, the observer O is essentially an algorithm for integrating and analyzing retrieved tracking information to report whether a special event type has occurred.

The proposed methodology can be summarized as follows. For verification, the methodology assesses whether a given observation configuration assures the observability of special behaviors within possible system behaviors. Fig. 2.(a) illustrates this verification capability. For design, the methodology identifies, for each event, which attributes need to be observed and suggests an optimal observation configuration meeting the observational requirements. Fig. 2.(b) illustrates this design capability.

IV. MODELING OF PARTIAL OBSERVATION

A. Preliminary

A monitoring agent (or observer) O detects special behaviors by analyzing observable discrete qualitative changes called discrete events (DE). Discrete events are symbols generated by observation devices (e.g., sensors) when items, traveling through the monitored system, visit tracking locations. The monitored system can thus be modeled as a DE system (DES), exhibiting discrete state spaces and event-driven dynamics. Suppose that the system G to be monitored has N tracking locations. This model G accounts for both the ordinary (non-special) and special behavior of the monitored system. The modeling approach is to construct DES models G^i for each of these tracking locations and then derive the global model G as the composition of these G^i; i.e., $G =\|_i G^i$ where $\|$ is the synchronous composition operator. Each of these locations is in-turn modeled by a finite state machine (FSM) G^i of four tuple, $G^i = \{X^i, \Sigma^i, \delta^i, x_0^i\}$, where X^i is a finite set of states, Σ^i is a finite set of event labels, $\delta^i : X^i \times \Sigma^i \times X^i$ is a transition relation on the state set, and $x_0^i \in X^i$ is the initial state of the system. The symbol ϵ denotes the silent event or the empty trace. An important element in formulating G is event labeling. In an item tracking application, each event uniquely corresponds to a particular item movement between two tracking locations. To this end, three attributes are here associated with each event symbol, namely, source location, target (current) location, and type (or flow-path). Accordingly, events are labeled as (i, j, k), where the event attributes i, j, and k identify the current location, previous location, and type, respectively, associated with an item.

B. Partial Observation

Under full observation, the cost of information may not represent an important deciding factor and all movement attributes (i.e., item type, previous and current locations) are assumed to be observable at each tracking location. Implementing full observation is often prohibitive due to many factors including information cost, intrusiveness, accessibility, covertness, and safety. Reducing information requirements for monitoring may also lead to monitoring applications with improved information management, reduced information cost, enhanced security, and tamper-resistance characteristics. Under partial observation, some item attributes may not be observable at some tracking locations. Unobservable events may cause changes in the monitored system, but are not completely communicated by any observation device. To model observational limitations, an observation mask function $M : \Sigma \rightarrow \Delta \cup \{\epsilon\}$ is introduced, where Δ is the set of observed symbols, which may be disjoint with Σ, with $M(\epsilon) = \epsilon$.

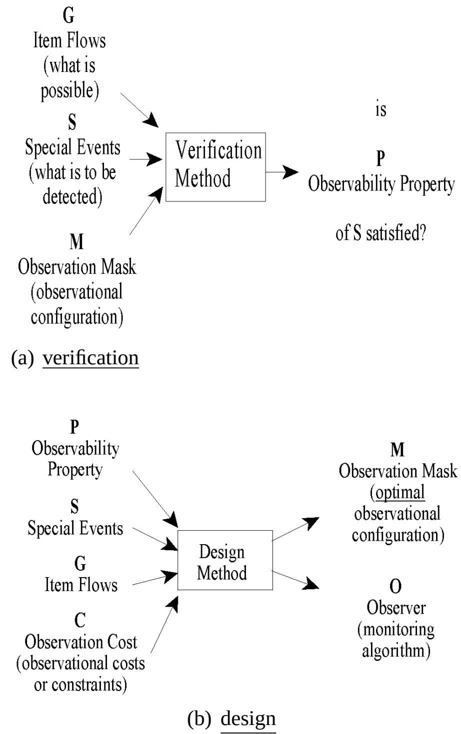

Fig. 2. Proposed methodology for item-flow monitoring

V. OBSERVABILITY NOTIONS

A. Definitions

Let $\Sigma_s \subseteq \Sigma_L$ denote the set of special events S which should be detected. This set Σ_s is partitioned into disjoint sets corresponding to different types of special events. This partition is denoted by Π_s and defined as $\Pi_s = \{\Sigma_{si} : \Sigma_s = \Sigma_{s1} \cup \ldots \cup \Sigma_{sm}\}$. Special events can occur repeatedly, so they need to be detected repeatedly. It is assumed that events in S are not fully-observable because otherwise they could be detected/diagnosed trivially. Due to partial observation, special events in S may be observable but with an observation symbol that is not unique under M. The notions of detectability and diagnosability are then central in the monitoring problem here addressed. Under detectability, the interest is in signaling the occurrence of special events (such as faults or route violations), but without explicitly indicating which routing event exactly has occurred. On the other hand, diagnosability is a refined case of detectability, where the interest often is in exact event identification. Thus, diagnosis is equivalent to detection when there is only one type of special events; i.e., $\Pi_s = \{\Sigma_{s1}\}$. Because detectability can be expressed as a relaxed case of diagnosability, we only provide the definition of diagnosability (as formulated in [9]) in this paper. To this end, the necessary notation is presented next. For all $\Sigma_{si} \in \Pi_s$ and a trace $s \in L$, let N_s^i denote the number of events in s that belongs to the special event type Σ_{si} (or i for simplicity). The post-language L/s is the set of possible suffices of a trace s; i.e., $L/s := \{t \in (\Sigma)^* : st \in L\}$. Notice that these observability notions are inherent properties of the monitored system G for given M and S.

Definition 1: (Uniformly bounded delay) $[1, \infty]$-Diagnosability [6, 9]
A prefix-closed live language L is said to be uniformly $[1, \infty]$-diagnosable with respect to a mask function M and a special-event partition Π_s on Σ_s if the following holds:

$$(\exists n_d \in \mathbb{N})(\forall i \in \Pi_s)(\forall s \in L)(\forall t \in L/s)[|t| \geq n_d \Rightarrow D_\infty]$$

where $\mathbb{N}$ is the set of non-negative integers and the condition

D_∞ is given by:

$$D_\infty : (\forall w \in M^{-1}M(st) \cap L)[N_w^i \geq N_s^i].$$

Definition 1 is suitable for the situation requiring the repeated detection of special events. Being repeatable, special events do not need to be permanent (nor remain in effect permanently) after their initial occurrence. A polynomial algorithm is provided in [6] for checking $[1, \infty]$-diagnosability. In order to improve it, [9] provides an algorithm that is one $|X|$ order less in time complexity. This algorithm was implemented in software and used in this paper to compute the sensor configuration illustrated.

B. Computation of Member-by-Member Optimal Observation Masks

The problem of selection of an optimal mask function is studied in [5]. A mask function is called optimal if any coarser selection of the equivalent class of events results in some loss of the events observation information so that the task at hand cannot be accomplished. This study also shows that the optimal mask function selection problem is NP-hard, in general. Assuming a mask-monotonicity property, it then introduces two algorithms for computing an optimal mask function. However, these algorithms assume that a sensor set supporting the mask function can be always found, which may not be true in practice. Given the above considerations, an algorithm is introduced in [3] that is related to the approaches presented in [4, 5]. This algorithm instead searches the sensor set space rather than the mask function space. The computed sensor set induces a mask function naturally. Thus, it does not suffer from the issue of realization of the mask function.

C. Procedure for Constructing a Observer

After computing an acceptable M that guarantees the desired observability property (i.e., detectability or diagnosability) using the optimization algorithm of [3], an associated observer O is constructed. The observer algorithm will integrate and analyze tracking information (or measurements) and report the occurrences of special events. To implement it, either an offline or an online design approach may be used for its construction. Under an offline design approach, the deterministic automaton representation of the observer is *a priori* constructed, which takes exponential time and space. To overcome this computational complexity, an online approach may be used instead [6]. Further improving [6] regarding computational complexity, an algorithm is provided in [9] that reduces not only the space required for realizing the observer state by $|X|^2$ but also the time complexity by $|X|$ if $\log(|X|) \approx |\Sigma|$.

VI. MODELING OF ITEM FLOWS

It is desirable to develop a modeling methodology that enables automatic construction of G. This objective can be accomplished if the methodology uses standard operators, such as the shuffle and the synchronous composition operators, to construct G from simple, readily-obtainable FSM models. To this end, the proposed methodology suggests constructing a possible-behavior model that embodies all possible routes that an item may follow, including both the special and the ordinary (non-special) behaviors. The ordinary component is constructed by considering only the item movements that are not required to be detected. The special component results from adding the set of special events of interest for detection. Thus, two types of states can be identified, namely, ordinary and special states. From the initial (reset) state, an ordinary state is reached by event traces containing no special events. A special state is reached by traces containing a special event. To illustrate the construction of a possible-behavior model for a monitored system, consider the system of Fig. 1, with the three (normal) item routes described. Any event belonging to one of these routes is denoted ordinary to signify that its occurrence is within the ordinary behavior of the monitored system. For simplicity, consider that each internal tracking location S_i has a unit capacity. The corresponding models of each tracking location i are shown in Fig. 3 and constructed as follows. At each tracking location i, define an "empty" state $0O$ that indicates that there is no item in it (O stands for "ordinary"). After identifying the set of ordinary item flows A_i that can bring an item to location i, define $|A_i|$ full states, with $|A_i|$ denoting the cardinality of A_i. Each full state kO indicates that there is an item at location i that arrived following the $k \in A_i$ ordinary route. Special events of interest also need to be included in these models. To illustrate, consider the five additional movements labeled with an S (for special) in Fig. 4, classified (by the observer) as special moves that should be detected, with (kS) denoting a special movement of type k.

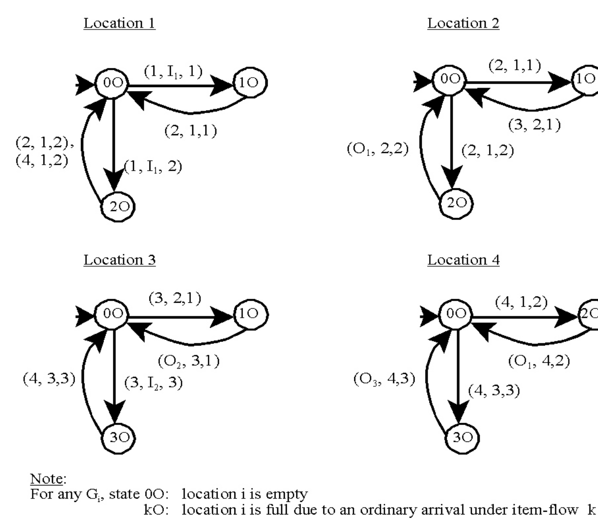

Fig. 3. Finite state models of each (inside) tracking location

These events are added to each relevant model G^i, as shown in Fig. 5. A state kS denotes that there is an item of type k at location i after completing a special movement. Notice that at the moment of their occurrence, special

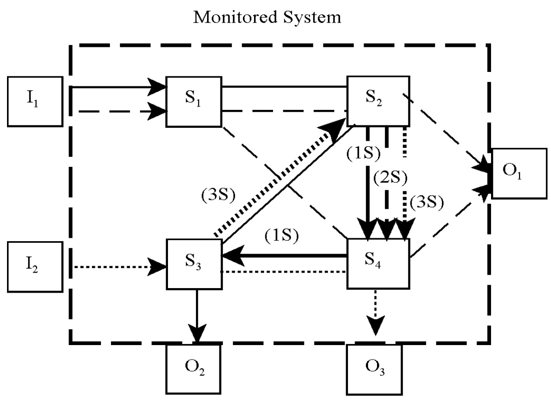

Fig. 4. Monitored system showing identified special movement of events S

events may be indistinguishable from ordinary events at the observer, depending on the M selected for the system. For example, assume that a "type" observation device is installed at the tracking location 4 in Fig. 4. In this case, the special event $(4, 2, 2)$ and the ordinary event $(4, 1, 2)$ are both seen with the same observation symbol $(4, -, 2)$ at the observer O so indistinguishable (at the observation instant), where $-$ is an unique symbol indicating that no information is available regarding the corresponding attribute. However, if the system is diagnosable w.r.t. $\{\{(4, 2, 2)\}\}$ (and M), O will eventually report its occurrence. To complete the construction of G, one additional modeling step is performed. As only logical specifications are considered here, the modeling absence of time leads to severe practical implications for the detection of logical specifications. To consider broader applications, the notion of n-event-causality may need to be enforced. This notion limits the number of system events that may occur between the occurrences of two consecutive events belonging to a given route. To model this n-event-causality requirement, additional states are added to each G^i illustrated in Fig. 5. For example, Fig. 6 shown the modified model for location 1 when enforcing a one-event-causality requirement. The symbol Σ_i^c denotes the set of events defined for the system, excluding those events defined for a given location i. An n-event-causality requirement is modeled similarly by adding n more states at each branch.

VII. ILLUSTRATIVE EXAMPLES

Consider the monitored system described in Figs. 1 and 4. The design objective is to identify an observation configuration M that provides sufficient item/entity tracking data to an observer O for detecting the occurrence of special events. For comparison purposes, Fig. 7 illustrates an observation configuration (from an ad hoc design) that would allow an observer to immediately detect any special event after its occurrence (i.e., with zero detection delay). Three types of observation devices are shown for retrieving item movement data. A circle, square, and triangular device provides the current location (i.e., attribute i), previous

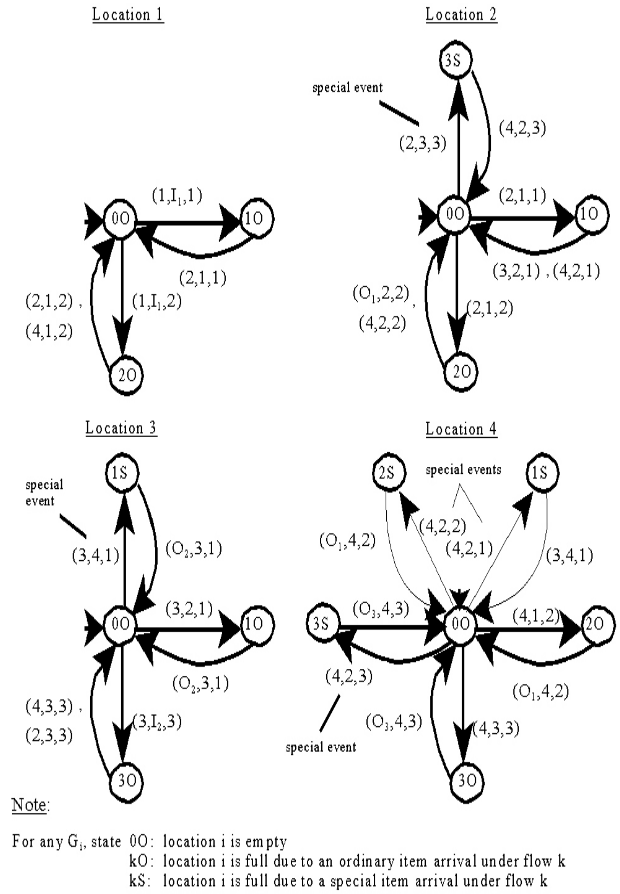

Fig. 5. Finite state models of each tracking location with both ordinary and special behaviors

location (i.e., attribute j), and type (i.e., attribute k) of an item, respectively.

To identify optimal observation configurations, the possible-behavior model G is constructed. The monitoring goal P regarding the set of special events S and an information cost C criterion are also specified. Assume that event diagnosis is of interest. The set S may thus be partitioned as $\Pi_s = \{\Sigma_{si} : i = 1, \ldots, 3\}$ with $\Sigma_{s1} = \{(4, 2, 1), (3, 4, 1)\}$, $\Sigma_{s2} = \{(4, 2, 2)\}$, and $\Sigma_{s3} = \{(2, 3, 3), (4, 2, 3)\}$. The procedure described in Sections III and V is then invoked to compute an M that optimizes C. For example, Fig. 8 illustrates an observation configuration where the design goal C is to reduce information requirements and preferably avoid using square observation devices. For this case, the monitoring goal P is the detectability of S and the model G considered satisfies a 2-events-casuality. Additional simulations were conducted with different designed M and O, considering different P, C, and "n" event casuality. The observer was always able to signal the occurrence of special events, with no miss-detection and no false alarms.

VIII. CONCLUSION

A mathematical framework and associated techniques for systematically designing and implementing model-based

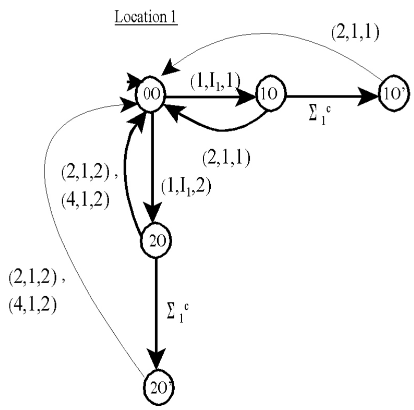

Fig. 6. DE model for location 1 including possible behavior and 1-event-causality requirement

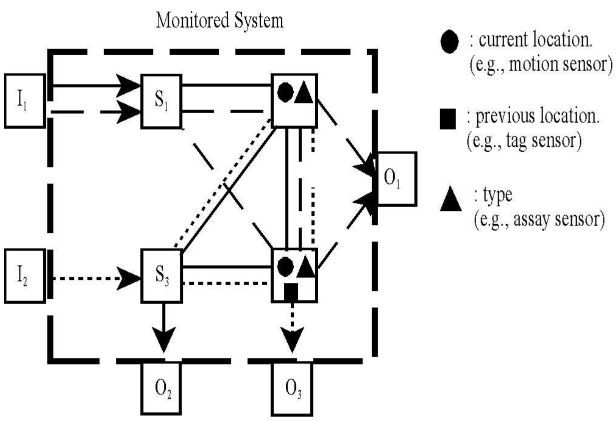

Fig. 7. Observation configuration for detecting special events with zero detection delay

monitoring of discrete flow networks was presented. The methodology can be used to: i) assess the intrinsic observability property (e.g., detectability and diagnosability) of a monitored system, given the selected observation configuration and the special behavior of interest; ii) implement model-based monitoring of discrete flow networks using a set of observation devices that optimize specified observational criteria; and iii) construct observers that automatically integrate and analyze item tracking data, formally guaranteeing observability of special item movements. The proposed methodology can thus be used to answer the question of how to optimally instrument a given monitored system based on its anticipated item/entity flows and the specified signatures or behaviors of concern.

Monitoring observers use not only current but also previously observed item tracking information for decision making. Knowledge and observation requirements are balanced for decision-making. This design and implementation methodology opens the possibility for information management optimization to reduce costs, decrease intrusiveness, and enhance automation, for example. Furthermore, it provides rich analysis capability (enabling optimization,

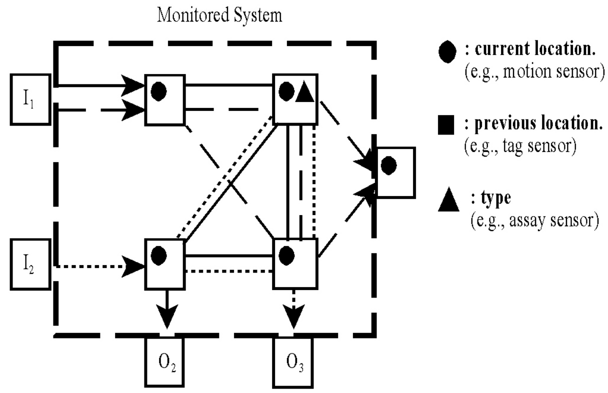

Fig. 8. Optimal observation configuration for detecting special events, assuming 2-event-causality

sensitivity, and what-if analysis), guarantees mathematical consistency and intended monitoring performance, yields a systematic method to deal with system complexity, and enables portability of system monitoring. Monitoring of item flows can thus be efficiently implemented in many applications, including large manufacturing facilities, complex computer networks, and dynamic battlespace operations centers.

ACKNOWLEDGEMENT

This work was supported by the U.S. Department of Energy contract W-31-109-Eng-38. The authors want to acknowledge Prof. R. Kumar for his contributions and interest on this work.

REFERENCES

[1] O. Contant, S. Lafortune, and D. Teneketzis, "Failure diagnosis of discrete event systems: the Case of intermittent failures," In Proc. 41^{st} *IEEE Conf. on Decision Control*, pp. 4006-4011, 2002.

[2] H.E. Garcia and T. Yoo, "Methodology to optimize and integrate safeguards sensor configurations and measurements in large nuclear processing facilities," In Proc. *INMM Annual Meeting*, 2003.

[3] H.E. Garcia and T. Yoo, "Modelbased detection of routing events in discrete flow networks," submitted for publication, 2003.

[4] A. Haji-Valizadeh and K.A. Loparo, "Minimizing the cardinality of an even set for supervisors of discrete event dynamical systems," *IEEE Trans. on Autom. Control*, 41(11):1579-1593, 1996.

[5] S. Jiang, R. Kumar, and H.E. Garcia, "Optimal sensor selection for discrete event systems with partial observation," *IEEE Trans. Autom. Control*, 48(3):369-381, 2003.

[6] S. Jiang, R. Kumar, and H.E. Garcia, "Diagnosis of repeated/intermittent failures in discrete event systems," *IEEE Trans. Robotics and Automation*, 19(2):310-323, 2003.

[7] M. Sampath, R. Sengupta, K. Sinnamohideen, S. Lafortune, and D. Teneketzis, "Diagnosability of discrete event systems," *IEEE Trans. Autom. Control*, 40(9):1555-1575, 1995.

[8] M. Sampath, S. Lafortune, and D. Teneketzis, "Active diagnosis of discrete event systems," *IEEE Trans. Autom. Control*, 43(7):908-929, 1998.

[9] T. Yoo and H.E. Garcia, "Event diagnosis of discrete event systems with uniformly and nonuniformly bounded diagnosis delays," In *Proc. 2004 American Control Conf.*, 2004.

[10] T. Yoo and S. Lafortune, "NP-completeness of sensor selection problems arising in partially-observed discrete-event systems," *IEEE Trans. on Autom. Control*, 47(9): 1495-1499, 2002.

[11] S.H. Zad, "Fault diagnosis in discrete event and hybrid systems," PhD thesis, University of Toronto, 1999.