Model based diagnosis in complex industrial systems: a methodology

Leonardo Barbini[1], Carmen Bratosin[2], and Emile van Gerwen[3]

[1,2,3] *ESI (TNO), High Tech Campus 25, Eindhoven, 5656 AE, The Netherlands*
leonardo.barbini@tno.nl
carmen.bratosin@tno.nl
emile.vangerwen@tno.nl

ABSTRACT

To fulfil market demand, the complexity of high-tech industrial systems is increasing every year. As a consequence, diagnosing failures leads to long down times, often because the issue at hand has to be escalated all the way to the development department to access the required knowledge. In this paper we propose a model based diagnostic methodology to support the field service engineer in finding the root cause of an unscheduled downtime in a timely way by bringing actionable design knowledge at the service engineer's fingertips. We start by modeling the knowledge on the machine hardware decomposition, deployment and functional behavior by using a domain specific language. The language allows for an object oriented description of the system, with the functional behavior captured as probabilistic relationships between hardware components. This system description is then automatically transformed into a Bayesian belief network (BN): the diagnostic model. When a diagnosis is required, we infer the health state of the system's components by instantiating the BN with discretized sensor and control data collected prior to the occurrence of a downtime. Finally, we compute the root cause hypotheses as the minimum sets of faulty components that are consistent with the evidence. We explain in a visual way the essential part of the diagnostic reasoning by pruning the causal graph underlying the BN, augmented with the data items supporting the hypothesis. We illustrate the methodology on a thermal conditioning subsystem. Our approach is domain independent and applicable to a wider range of cyber physical systems (CPS).

1. INTRODUCTION

Serviceability of today's high-tech system is a difficult task even for experienced personnel.The growing complexity of the high tech system leads to an increasing probability of occurrence of a fault within one of its parts. A failing part triggers a subsequent cascade of misbehaving parts, i.e. parts which do not have faults but which do not behave as expected because of their connection with the faulted one. Finding the root cause becomes a cumbersome task.

At the same time, there is a transition towards systems which by design adapt their behavior according to the health conditions of their components, either by control loops or by instrumented safety measurements. Such adaptive system are designed to mitigate the effect of tear and wear of a component on the overall system behavior. In this case, the system level performance is no longer reflecting the health of individual components.

Diagnosis effort is also influenced by an increasing distance between the service and the development (design and engineering) departments. On the one hand the service personnel has to find the smallest replaceable part which after substitution will ensure a correct behavior of the system. On the other hand, the development personnel will seek the failure mode leading to the malfunctioning of such part, in order to redesign it and avoid similar failures in the next generation of the system. For diagnosing complex systems, the service department has to rely on actionable information from the development department, which in practice does not meet the service needs.

The challenges described above are unavoidable: the first two are dictated by the market's demand of growing productivity and efficiency of the industrial system while the latter is intrinsic to an industrial organization. To tackle the increasing difficulty in serviceability of high tech systems, in this paper we offer a model based diagnostic methodology which by means of proper transformations brings the knowledge of the expected system behavior, from the development to the service department.

The paper is organized as follows: Section 2 introduces the relevant literature, Section 3 describes the steps involved in the methodology, Section 4 shows its implementation on an industrial use case, remarks and directions for future research are drawn in Section 5.

2. RELATED RESEARCH

The field of diagnostics is wide and has many different facets. Starting from the diverse diagnostic goals such as a reactive or a proactive one, which can be reached with different methods, knowledge based, data based or hybrid (Isermann, 2006).

Diagnostics has different targets: component, system or the product of an industrial process. (Yang, Duan, Shah, & Chen, 2014). Where a component is a basic element, such as a spare part an electronic board or a software function, a system is a set of connected components that perform a specific task. While the product is the output of a system, its quality is defined by the actions of the system but also by other systems, in the case of a manufacturing line. For these reasons, the literature in the field is abundant and the outcomes are continuously summarized in thorough reviews such as (Solé, Muntés-Mulero, Rana, & Estrada, 2017) and (Escobet, Bregon, Pulido, & Puig, 2019).

Despite the varied landscape of the diagnostic field, from a general point of view, it is possible to observe only two main approaches. In the first approach, one tries to identify the failure modes of faulty components and the propagation of each mode through the different parts of the system, by providing the set of expected symptoms.The second approach is to model only the normal, i.e. expected, behavior of the system's components and the interconnection between the components: Model Based Diagnostics (MBD).

Traditionally, the industry has adopted failure modes based approach and implements it by using standards methods such as failure mode and effects analysis, fault tree analysis or their combination (Stamatelatos, Vesely, Fragola, Minarick, & Railsback, 2002) and (Peeters, Basten, & Tinga, 2018). The advantage of such an approach is that it provides a direct mapping from symptoms to faults. However, it has several limitations (Pillay & Wang, 2003). For example the space of all possible faults of a complex industrial systems is very large, with a considerable number of faults not foreseen at design time, but only discovered during the operational phase. Additionally, many faults have cross-functional collateral damages that would require multidisciplinary teams to define them. Finally, this approach increases the workload on the development organization.

MBD models have the advantage that can be extracted from design models and functional requirements, discovered from data or directly made by experts. Given such models the diagnostic problem then becomes a consistency problem between observed behavior and modeled behavior (de Kleer & Williams, 1987) and (Reiter, 1987). Several algorithms have been developed for the MBD approach (Wotawa, 2001) and (Feldman, Provan, & Van Gemund, 2010).

Most of the papers (Wotawa, 2001; Feldman et al., 2010), translate the MBD into a first order logic representation that allows fast inference. In this paper, we use Bayesian belief networks (BN) as reasoning engine for MBD due to their ability to deal with partial observability of a system, which is very often encountered in large systems (Flesch, 2008). This allows inference of the health state of components even with a limited set of sensors deployed in the system. Moreover, BNs have reached a maturity level for industrial application, thanks to software (*Norsys*, 2019) which provides BN modeling environment together with the necessary reasoning algorithms.

The disadvantage of MBD lies in the fact that the negation of normal behavior contains also physically unlikely behavior, thus generating false diagnoses. However this is mitigated by excluding them during the modelling phase. This approach provides excellent results on localization of the defective components, and in capturing the effect of the propagation of a failure in the system.

3. METHODOLOGY

In this paper we adopt MBD due to its fault localization power that guides a field service engineer towards the right corrective measure. In this section firstly we introduce some definitions from the MBD approach, then we show how to build a diagnostic model for a small system. Furthermore, given such a model, we illustrate how to perform diagnostic inference and finally we show how to help the field service engineer by explaining the results. Note that throughout the paper we assume to start our diagnostic analysis from a machine hard-down, i.e. after fault detection was already performed.

3.1. Model based diagnosis

MBD (Reiter, 1987) assumes the system $S = (SD, CMP)$ where SD is the system description, given as a finite set of logical formulae, and CMP is a finite set of components. A component c can be $Normal$ or $Abnormal$, i.e. behaving as expected or not. The $Normal$ behavior is defined by functions on the inputs and outputs of a component. The $Abnormal$ behavior is defined as negation of the $Normal$ behavior. We use the shorthand notation $c = Abnormal$ to express if a component is in an $Abnormal$ state. Components are connected via functions, linking the outputs of a component to the inputs of another component. Both the functions describing the behavior of the components and their interconnections belong to the SD.

Given a set of observations OBS (assignment of inputs and outputs of components to a specific value or state) and D the assignment of CMP to $Normal$ or $Abnormal$. D is called a diagnosis iff the observations are consistent both with the system description SD and the diagnosis D:

$$SD \cup D \cup OBS \not\models \bot \tag{1}$$

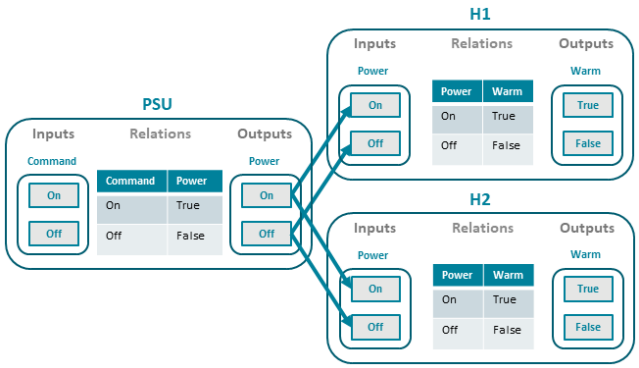

Figure 1. Model of expected behavior for a small system.

Where $\not\models$ stands for the negations of the logical entailment relation and $\perp$ represents a contraction.

Given D a diagnosis and $AB \subset CMP$ the set of components such that $D = \{c = Abnormal \mid c \in AB\} \cup \{c = Normal \mid c \in CMP \setminus AB\}$. D is a minimum diagnosis iff $\nexists\, AB' \subset AB$, such that $D' = \{c = Abnormal \mid c \in AB'\} \cup \{c = Normal \mid c \in CMP \setminus AB'\}$ is a diagnosis.

3.2. Modeling

As an illustrative example of our methodology let us consider a small system S consisting of two heaters ($H1$, $H2$) powered by a power supply unit (PSU). For S we have $CMP = \{H1, H2, PSU\}$. Then, to define SD, we start by modeling the expected behavior of the system's components. This is shown in Figure 1.

Each component has $Inputs$, $Outputs$ and $Relations$. $Inputs$ and $Outputs$ consist of variables, e.g. $Command$ for the PSU, where each variable can take only discretized states, e.g. On or Off. Relations are between states of $Input$ variables and states of $Output$ variables, e.g. if the $Command$ is On in a PSU we expect $Power$ to be On. $Relations$ are captured as first order logic.

Then, we define $Connections$ between components, as shown with blue arrows. $Connections$ are between states of $Output$ variables of one component and states of $Input$ variables of another component. For this system, SD is then constituted by both $Relations$ and $Connections$.

Finally, we define observables the variables for which we can measure the state, e.g. by sensor data. Usually, the number of observables is a small subset of the total number of variables.

3.2.1. Bayesian belief networks for MBD

In this paper we implement MBD by translating SD into a Bayesian belief network (BN) (Pearl, 1986). A BN is defined as a directed acyclic graph, and a joint probability distribution of a set of random variables X. There exists a 1–1 correspon-

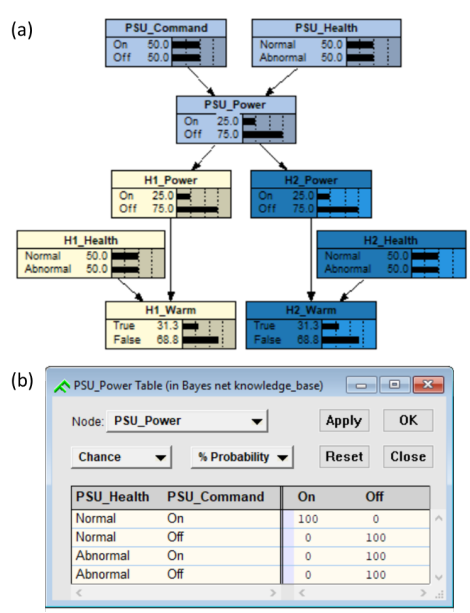

Figure 2. BN for the small system (a). Conditional probability table for the node PSU_Power (b).

dence between the nodes of the graph and the random variables in X. The arcs of the graph express causal dependence relationships between the variables. For each of the nodes, a local conditional probability table (CPT) is defined. In the following we use the Netica software from (*Norsys*, 2019) for visualization of BNs and its API for the Bayesian inference.

We translate the system SD into a BN with five steps, similarly to (Flesch, 2008):

1. Map each variable in SD to a node in the BN, i.e. a random variable with name $\ll component \gg\_\ll variable \gg$
2. Insert edges between $Inputs$ and $Outputs$ nodes to capture $Relations$
3. Insert edges between $Outputs$ and $Inputs$ nodes to capture $Connections$
4. Add a $Health$ node (denoted $\ll component \gg\_Health$) with states $Normal$ and $Abnormal$ for each component
5. Insert an edge between the $Health$ node and the $Outputs$ nodes for each component

The role of the $Health$ nodes in the BN is to monitor the $Input$-$Output$ relations. After applying these steps, the resulting BN for the system of Figure 1 is shown in Figure 2 (a).

For the prior probabilities of the BN we set a uniform distribution for all the root nodes, e.g. $Inputs$ and $Health$ nodes. For the conditional probability table of the $Outputs$ nodes, we have to fill in all the combinations between states of the $Inputs$ nodes and the two states of the $Health$ node. We use the first order logic from the $Relations$, for the combinations

containing the $Normal$ state. While we use their negation for the combinations containing the $Abnormal$ state.

Finally, we modify all the physically impossible $Abnormal$ behaviors. For example, for a PSU the physically impossible behavior is $PSU\_Command = Off$, $PSU\_Health = Abnormal$ and $PSU\_Power = On$, as negation of $Relation$ $PSU\_Command = Off$, $PSU\_Health = Normal$ and $PSU\_Power = Off$. We modify the impossible behavior into $PSU\_Power = Off$. An example of a CPT is shown in Figure 2 (b).

3.2.2. Implementation via domain specific language

Industrial systems contain thousands of components and knowledge on their behaviour is distributed between multiple departments. For this reason, the diagnostic model has to be implemented in a compositional and scalable manner. Additionally, systems are built incrementally or as a family of systems that share a common platform. So one wants to be able to re-use parts of the model that are still relevant.

To answer the above requirements on compositionality, scalability and re-usability, we propose to construct the diagnostics models by using a Domain Specific Language (DSL) which allows for an object oriented approach. Throughout this paper we use a DSL previously introduced by the authors in (Velikova, Bratosin, Ypma, Lemmen, & van Wijk, 2019). This DSL allows for a quick generation of large models.

Specifically, in the DSL we define classes for each type of component. Then we create and interconnect instances of these classes according to the system description. The DSL has an import functionality that we use to import the created instances and to add hierarchy to the diagnostic model. The DSL tooling then automatically generates the BN and a graphml (*GraphML*, 2019) representation of the DAG. In this way the DSL allows us to quickly generate large BNs in an object oriented way (Borth & Barbini, 2019).

In Figure 3 (a) and (b) we show the DSL instances to represent the small system of Figure 1. Specifically (a) shows the instance describing the PSU and (b) the small system. For details on the semantics of the language we refer the reader to (Velikova et al., 2019). The DSL instance of Figure 3 (b) automatically generates the BN of Figure 2 and the graph of Figure 3 (c).

3.3. Diagnostic inference

Given a generated BN for a system S the MBD consistency diagnosis of Equation 1 can be translated into classical Bayesian reasoning (Geffner & Pearl, 1992) and (Flesch, 2008). That is, given the observations OBS, on the states of the observable nodes of the BN, and D the diagnosis as the set of $Normal$ / $Abnormal$ values for the $Health$ nodes:

$$SD \cup D \cup OBS \not\models \bot \iff P(OBS|D) \neq 0 \qquad (2)$$

where $P(OBS|D)$ is the probability of OBS given D for BN.

Practically, when we have to solve a diagnostic problem, we determine the set of abnormal components AB as all components with a $Health$ node with probability higher than 0.5 of being in state $Abnormal$, when adding OBS as evidence in the BN, i.e.: $AB = \{c \mid c \in CMP$ and $P(\ll c \gg\_Health = Abnormal|OBS) > 0.5\}$

Algorithm 1 Minimum Diagnoses

```
function MINIMUMDIAGNOSES(BN,AB,OBS)
    MD ← {}
    for number_failures ∈ [1 : length(AB)] do
        AB' ← combinations(AB,number_failures)
        for comps ∈ AB' do
            D' ← {c = Abnormal | c ∈ comps}
            D' ← D' ∪ {c = Normal | c ∈ AB \ comps}
            if P(OBS|D') ≠ 0 then
                if {m ⊂ comps | m ∈ MD} = ∅ then
                    MD ← MD ∪ {D'}
                end if
            end if
        end for
    end for
    return MD
end function
```

Note that AB defines diagnosis $D = \{c = Abnormal \mid c \in AB\} \cup \{c = Normal \mid c \in CMP \setminus AB\}$ that is a superset of all possible minimum diagnosis, as defined in Section 3.1. To find the minimum diagnosis we use Algorithm 1.

The inputs to Algorithm 1 are BN, OBS and the set AB. Note that we iterate the subsets from length 1 to the length of AB and we check that new found diagnoses are minimum with respect to previous ones. Algorithm 1 returns MD as the set of sets of minimum diagnoses. The number of computations of Algorithm 1 can be greatly reduced by iterating only up to a given number of allowed simultaneous diagnosis instead of the length of AB, i.e. by using the principle of minimal cardinality.

For the simple system considered here, let us suppose that the observations are: $OBS = \{PSU\_Command = On, H1\_Warm = False, H2\_Warm = False\}$. Then using Bayesian inference $AB = \{PSU, H1, H2\}$ and using the above algorithm $MD = \{\{PSU = Abnormal, H1 = Normal, H2 = Normal\}, \{PSU = Normal, H1 = Abnormal, H2 = Abnormal\}\}$, i.e., either the PSU is broken, or the two heaters H1 and H2 are simultaneously broken.

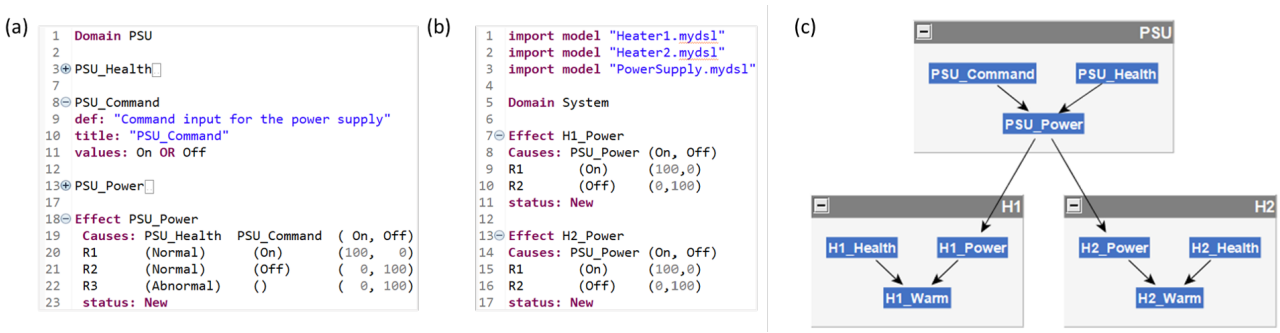

Figure 3. DSL instances for the PSU (a) and the system (b). Causal graph for the system (c).

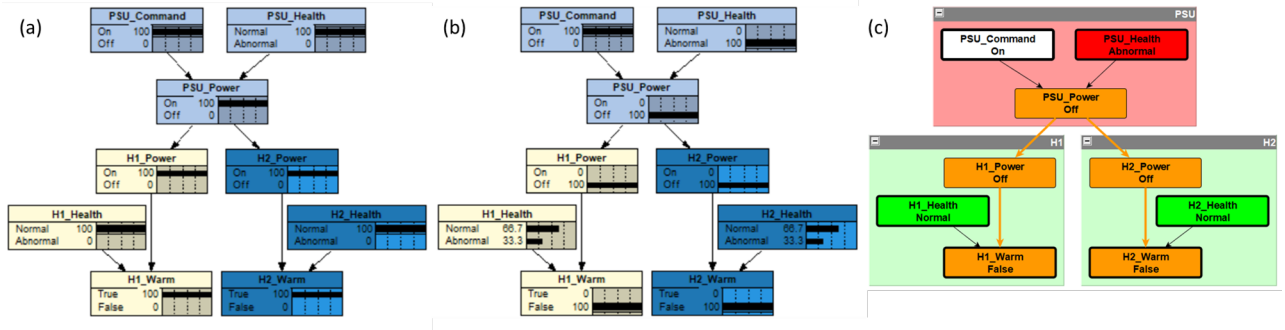

Figure 4. BN with expected behavior (a). BN with observed behavior (b). Explanation graph (c)

3.4. Diagnostics under dynamic behaviour

The diagnostic inference approach described above returns the diagnoses for the system given OBS at only one timestamp, i.e. is a static approach. As faults typically cause a cascading effect of misbehavior, the diagnostic process requires going back in time to identify the root cause.

For example, let us think of a system with a built-in safety function. With the function triggered whenever one of its inputs exceeds a safety value. From a diagnostic point of view, one then needs to diagnose which failing components triggered the safety function, not the safety function itself.
To tackle this intrinsic dynamic property of the diagnostic analysis, we apply Algorithm 1 at different time stamps. Specifically, given the time sequence of observations

$$[OBS_{t_{inc}}, OBS_{t_{inc}-1}, OBS_{t_{inc}-2} \cdots]$$

we start from the machine hard-down time, i.e. incident time t_{inc}, and we repeatedly apply the algorithm while proceeding backwards in time. Further, we compute the minimum diagnoses only when we observe a state change in the system: $OBS_k \neq OBS_{k-1}$. This gives us a time sequence of minimum diagnoses

$$[MD_{t_{inc}}, MD_{t_{inc}-i}, MD_{t_{inc}-j}, \cdots]$$

where i, j are the timestamps of the states changes preceding t_{inc}. Given this sequence, each MD holds from the state it is computed up to to the next state change, i.e. $MD_{t_{inc}-i}$ holds up to $MD_{t_{inc}}$. We stop this iterative approach based on manual input from the service engineer, when the computed MD satisfies the diagnostic needs.

3.5. Explanation

Besides delivering the diagnoses, in an industrial application being able to explain the diagnostic reasoning that led to such an outcome is a great added value. To do so we propose a visual explanation of the diagnoses. We achieve this in two steps: pruning to remove irrelevant facts and coloring to convey essential information.

To prune the BN we repeatedly use the notion of Markov Blanket (MB) (Pearl, 1988) of a node n in a BN. That is, all the nodes that make n conditionally independent from the rest of the BN. $MB(n)$ is represented by the children of n, its parents and its children's parents. Algorithm 2 presents the pruning algorithm. The inputs are BN, AB as the set of abnormal components and O as nodes with evidence in OBS. The algorithm stops when all boundary nodes of the pruned graph are either a root node or a node in O. As a second step we use different colors for nodes and edges in the graph which have a different meaning for the diagnostic reasoning. Specifically, for all nodes in AB with red and all nodes in $CMP \setminus AB$ with green. Further, we emphasize the nodes in the observed behavior (OB) that differ from the expected

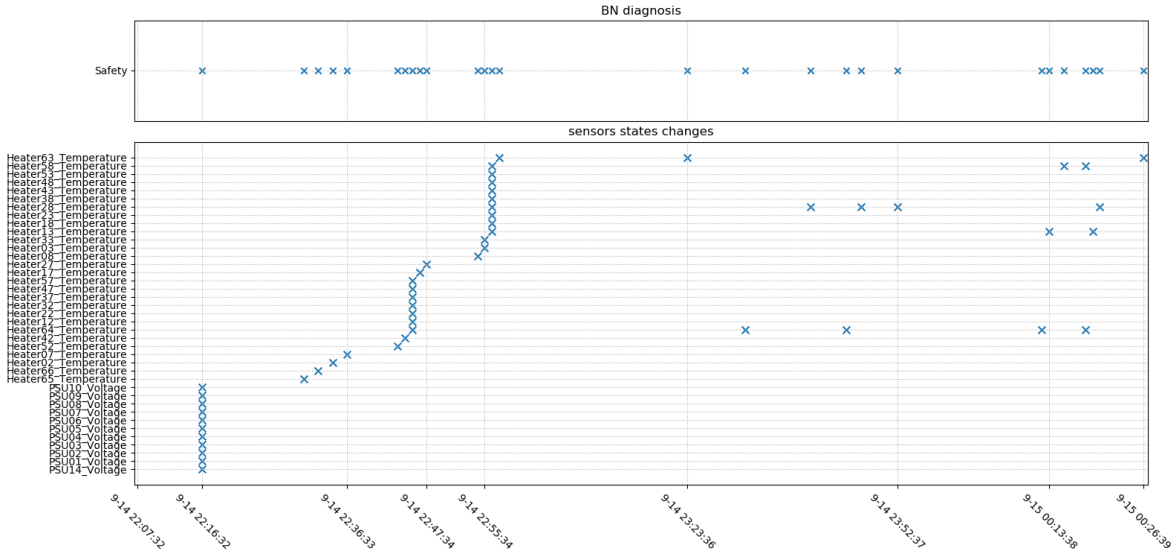

Figure 5. Dynamic reasoning with BN for the industrial use case.

behavior (EB) of the system.

EB is defined as the states of BN nodes when all $Health$ nodes states are $Normal$ and all root nodes are set to expected values by design. For the example of Section 3.2 to obtain EB we add as evidence $PSU\_Command = On$, $PSU\_Health = Normal$, $H1\_Health = Normal$, and $H2\_Health = Normal$. Figure 4 (a) shows the resulting EB. The OB for a diagnosis D, is obtained by inserting OBS and all components AB in D with $Health$ state *Abnormal* as evidence. Figure 4 (b) presents OB when we insert $OBS = \{PSU\_Command = On, H1\_Warm = False,$ *H2_Warm* $= False\}$ and PSU_Health = *Abnormal*. Finally, the nodes with different states in OB and EB are colored in orange, together with the shared edges. Nodes with same state are colored in white. Finally, for all nodes with evidence in OBS and all Health nodes we draw a thick border. For our example we present the explanation graph in Figure 4 (c).

Algorithm 2 Pruned Graph

function PRUNEDGRAPHNODES(BN,AB,O)
  $Q \leftarrow \emptyset$
  $PG \leftarrow \emptyset$
  $Q \leftarrow Q \cup AB$
  while $Q \neq \emptyset$ **do**
    $n = removeElement(Q)$
    if $n \notin O \cup rootNodes(BN) \cup PG$ **then**
      $PG \leftarrow PG \cup \{n\}$
      $Q \leftarrow Q \cup MB(n)$
    end if
  end while
  return PG
end function

4. INDUSTRIAL APPLICATION

We applied our approach for the heating subsystem of a cyber physical system (CPS). The heating subsystem contains the following components: 16 PSUs, 70 heaters and 24 thermocouples. The PSUs, heaters and thermocouples form 16 groups. Each group is controlled with a closed loop that regulates the heaters temperature.

Two safety systems are present in the CPS, one at the level of the heating subsystem and one at the CPS level. The former instantaneously stops the heating on detection of an error on any of the heating subsystem's components. The latter, stops the CPS after two hours of continuously missing heating support.

We model this system using our methodology and we abstract from the control and safety loops by defining $Control$ and $Safety$ as On/Off switches. To perform diagnostics we instantiate the BN with discretized sensor data coming from a CPS which resulted in system downtime. For the discretization we use expert-defined thresholds.

The results of our methodology applied on a specific downtime are summarized in Figure 5. The top plot shows possible diagnosis computed by the BN, the bottom plot shows state changes across the discretized sensor data. In Figure 5 the moment of the incident, i.e. the moment when the field service engineer is alerted, is on the right hand side. We observe that according to our model the system is shut down because $Safety$ was triggered.

To further diagnose what triggered $Safety$, we repeat the analysis going backwards in time, as explained in Section 3.4. In Figure 5 we notice that the diagnosis remains the same until two hours and ten minutes before the incident time,

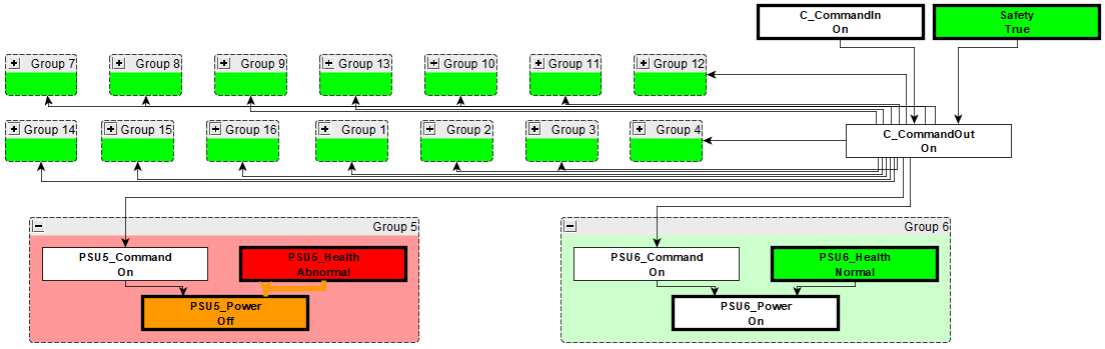

Figure 6. Explanation graph for PSU5 Abnormal behavior.

i. e. machine hard-down. Before that moment the system was in a $Normal$ state. So, we conclude that in the window of time between the $Normal$ diagnosis and the diagnosis $Safety = Unsafe$, one of the components failed. To finalize our diagnosis, we use data from additional sources, e.g. event logs and board dumps, at this specific window of time and we identify the component that failed.

Without our model in place, the field service engineer would manually check at the incident time the health of 110 components. Furthermore, he would not have the knowledge that the failure that actually triggered the safety measure occurred more than two hours before and that all the following behavior of the CPS was as expected by the safety system. Therefore, he would end up in a long service action. For the incident in Figure 5 a downtime of almost six hours was recorded. Our analysis took less than a minute where sensor data was available for a time frame of 24 hours.

As described in the previous section, we also provide an explanation graph of the diagnosis. Specifically, we produce an explanation graph for each minimal diagnosis at each time for which we perform diagnostics. Figure 6 shows an example of such an explanation graph for a diagnosis for the CPS. Please note that this is a different downtime than the one shown in Figure 5. By using our pruning algorithm, the DAG of the CPS was reduced of a factor of 6 , from more than 600 nodes to approximately 100 nodes and colouring quickly leads to the faulted component.

5. CONCLUSIONS

In this paper we showed how to exploit model based diagnosis to support field service engineers in the localisation of faults in complex industrial systems. To build our diagnostic models we used a domain specific language combined with an object oriented modeling approach. This allowed us to create large models quickly and with little effort. In addition, we used Bayesian networks to perform diagnostic inference. In this way we coped with the partial observability of the system.

In our work we extended state of the art by introducing an algorithm to compute minimal diagnosis given the diagnostic outcome of the Bayesian network. Also, we showed how to explain a diagnosis in a visual way by pruning and coloring the graph underlying the Bayesian network. Finally, we applied our methodology on an industrial use case and we described how to cope with the dynamic aspect of the diagnostic problem by using the model at different moments in time.

Our diagnostic models are currently instantiated using operational sensor data, i.e. data that is often used for control purposes. However this data is often not enough to solve diagnostics problems and additional data sources must be taken into account. For this reason, in the future we want to augment our models by connecting to machine event logs or electronic boards dumps that are automatically collected whenever there are state changes at the hardware or software level. As future work, we are also looking into further ease the modeling efforts by automatically extracting the models from design artefacts as in (Tretmans, 2007). Lastly, we are currently extending our methodology to support design engineers in using model based diagnosis to define an optimal sensors placement, i.e. design for diagnostics.

ACKNOWLEDGEMENT

The research is carried out as part of the ADiA program under the responsibility of ESI (TNO) with ASML as the carrying industrial partner. The ADiA research is supported by the Netherlands Ministry of Economic Affairs.

REFERENCES

Borth, M., & Barbini, L. (2019, September). Probabilistic health and mission readiness assessment at system-level. In *Proceedings of the annual conference of the phm society* (Vol. 11).

Escobet, T., Bregon, A., Pulido, B., & Puig, V. (Eds.). (2019). *Fault diagnosis in dynamic systems*. Springer International Publishing.

Feldman, A., Provan, G., & Van Gemund, A. (2010). Approximate model-based diagnosis using greedy stochastic search. *Journal of Artificial Intelligence Research*, *38*, 371-413.

Flesch, I. (2008). *On the use of independence relations in bayesian networks*. Radboud University, Nijmegen.

Geffner, H., & Pearl, J. (1992). Distributed diagnosis of systems with multiple faults. *Readings in model-based diagnosis*, 453-459.

Graphml. (2019). `graphml.graphdrawing.org`.

Isermann, R. (2006). *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer-Verlag Berlin Heidelberg.

de Kleer, J., & Williams, B. C. (1987). Diagnosing multiple faults. *Artificial Intelligence*, *32*(1), 97 - 130.

Norsys. (2019). `www.norsys.com/netica.html`.

Pearl, J. (1986). Fusion, propagation, and structuring in belief networks. *Artificial Intelligence*, *3*(29), 241-288.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Peeters, J., Basten, R., & Tinga, T. (2018, 4 1). Improving failure analysis efficiency by combining fta and fmea in a recursive manner. *Reliability engineering and system safety*, *172*, 36–44. doi: 10.1016/j.ress.2017.11.024

Pillay, A., & Wang, J. (2003). Modified failure mode and effects analysis using approximate reasoning. *Reliability Engineering and System Safety*, *79*(1), 69-85.

Reiter, R. (1987). A theory of diagnosis from first principles. *Artif. Intell.*, *32*(1), 57-95.

Solé, M., Muntés-Mulero, V., Rana, A. I., & Estrada, G. (2017). *Survey on models and techniques for root-cause analysis.*

Stamatelatos, M., Vesely, D. J., W., Fragola, J., Minarick, J., & Railsback, J. (2002). *Fault tree handbook with aerospace applications.* NASA.

Tretmans, J. (Ed.). (2007). *Tangram: model-based integration and testing of complex high-tech systems*. Eindhoven, NL: Embedded Systems Institute.

Velikova, M., Bratosin, C., Ypma, A., Lemmen, V., & van Wijk, R. J. (2019). Assisted diagnostics methodology for complex high-tech applications. In *2019 4th international conference on system reliability and safety (icsrs)* (p. 457-463).

Wotawa, F. (2001). A variant of reiter's hitting-set algorithm. *Information Processing Letters*, *79*(1), 45 - 51.

Yang, F., Duan, P., Shah, S. L., & Chen, T. (2014). *Capturing connectivity and causality in complex industrial processes*. Springer International Publishing.