# Using Enterprise Architecture Models and Bayesian Belief Networks for Failure Impact Analysis

Oliver Holschke[1], Per Närman[2], Waldo Rocha Flores[2], Evelina Eriksson[2], and Marten Schönherr[3]

[1] Technische Universität Berlin, Fachgebiet Systemanalyse und EDV, FR 6-7, Franklinstr. 28-29, 10587 Berlin, Germany
Oliver.Holschke@sysedv.tu-berlin.de
[2] Dpt. of Industrial Information and Control Systems, Royal Institute of Technology (KTH), Stockholm, Sweden
{PerN,WaldoR,EvelinaE}@ics.kth.se
[3] Deutsche Telekom Laboratories, Ernst-Reuter-Platz 7, 10587 Berlin, Germany
Marten.Schoenherr@telekom.de

**Abstract.** The increasing complexity of enterprise information systems makes it very difficult to prevent local failures from causing ripple effects with serious repercussions to other systems. This paper proposes the use of Enterprise Architecture models coupled with Bayesian Belief Networks to facilitate Failure Impact Analysis. By extending the Enterprise Architecture models with the Bayesian Belief Networks we are able to show not only the architectural components and their interconnections but also the causal influence the availabilities of the architectural elements have on each other. Furthermore, by using the Diagnosis algorithm implemented in the Bayesian Belief Network tool GeNIe, we are able to use the network as a Decision Support System and rank architectural components with their respect to criticality for the functioning of a business process. An example featuring a car rental agency demonstrates the approach.

**Keywords:** Enterprise Architecture Management, Decision Support Systems, SOA, Bayesian Belief Nets, Diagnosis, Failure Impact Analysis.

## 1 Introduction

Today's businesses are increasingly dependent upon IT, not only to support their business processes but also to automate their business processes. With the advent of integration technologies the information systems have become more and more interconnected. This means that the decision makers in charge of managing the enterprise information systems have lesser possibilities of knowing how any particular decision concerning changes to the information systems affect other information systems or for that matter the business itself. The process of conducting and planning preventive IT maintenance and allocating maintenance and operation resources where they do the most good is one area where the sheer complexity of the information system poses a problem.

Enterprise Architecture (EA) is a proposed solution to reduce complexity and allow for better decision-making. Using EA models illustrates the architectural components of the enterprise system and their interconnections in a way that is comprehensible to ordinary people. There is a plethora of EA frameworks currently in use including The Open Group Architecture Framework (TOGAF) [1], the Department of Defense architecture framework (DoDAF) [2], and others [3]. Although many of these frameworks propose models that capture interconnections between systems, they fail to depict causal relations between availability of various architectural elements.

To be able to depict and model causal relations between systems, one might use Bayesian Belief Networks [4], which feature a graphical notation to capture causal relations in a more qualitative fashion. In addition to this, there is also a statistical apparatus behind the graphical notation with which one can quantitatively estimate which decision yields the most benefit. We propose a Decision Support System (DSS) for failure impact analysis for Enterprise Architectures based precisely on the Bayesian Belief Networks (BBN) introduced above. Our proposed management process and underlying DSS address several concerns of enterprise architects identified in the *Enterprise Architecture Management Pattern Catalogue* [5] which has been developed at the Technische Universität München, Germany. Our method can be regarded as an implementation of the *Infrastructure Failure Impact Analysis (M-34)* pattern which addresses concerns about infrastructure failures, including concerns about the probability of these failures being causes for process, service or other element defects. The method of creating the DSS consists of EA models based on the ArchiMate meta-model [6, 7] and their translation to a BBN, then using an algorithm to simulate which architectural element is the most critical. The following section proceeds to describe BBNs in general and diagnostic analysis. Section 3 describes how to apply BBNs and diagnostic analysis and how to create the Decision Support System for failure impact analysis. The creation of the DSS is demonstrated in section 4, applied to an example car rental agency. The diagnostic use of the DSS is illustrated in section 5. Section 6 concludes the paper.

## 2 Bayesian Belief Networks and Diagnostic Analysis

### 2.1 Bayesian Belief Networks

A Bayesian Belief Net (BBN) is a graphical model that combines elements of graph and probabilistic theory. A BBN describes a set of causal relations within a set of variables, and a set of conditional independencies including joint probabilities as depicted in Fig. 1. A directed, acyclic graph (DAG) represents the causal dependencies between the variables (or nodes). Each node represents a variable with corresponding conditional probability distribution, displayed in a Conditional Probability Table (CPT). The strength of BBN manifests in the possibility of reasoning about results given certain observations according to Bayesian rules. BBN can answer requests of the form "what, if …" with respect to specific variables. Applied in this way BBN are powerful probabilistic inference machines [8]. Further explanations on the semantics of BBN can be found in e.g., [4, 9].

While the structure of a BBN may in principle be unknown, we propose to exploit the availability of an EA model in which nodes and relations are "known". Doing this relieves us of learning an expressive BBN structure, e.g., by search-and-score procedures [8]. The exact mapping of an EA model to a BBN structure will be explained in section 3 and 4. The parameters of a BBN can either be collected from historical data and/or expert assessments, or learnt via estimation methods such as Maximum-Likelihood or Bayesian Estimation [8, 9]. Fully parameterized BBN can be used for different inferential tasks, i.e., classification (mapping a data element into a previously defined category), prediction (the forecast of a value of a dependent variable given a specific observation), and diagnosis (concluding a possible cause of a changed variable given a specific observation). With respect to the concerns of enterprise architects the use type *diagnosis* is of particular relevance. If the architect – in case of EA changes – had means of identifying causes of disruptive effects, this would benefit his architectural decision-making process in terms of efficiency and effectiveness. For this *diagnostic analysis* can be conducted on a BBN. In the following we briefly describe how diagnosis is applied.

## 2.2 Diagnostic Analysis

The following description of diagnosis in BBN is based upon [10], a master thesis that specifically describes the implementation of the relevant diagnostic functions in the Bayesian Belief Network modeling tool GeNIe [11] developed at the University of Pittsburgh.

Diagnosis involves two types of tasks: 1) determining the (combination of) causes of the observations, and 2) increasing the credibility of the diagnosis through the collection of additional, initially unobserved, data. Since information seldom comes for free, the second task by necessity involves the formulation of a strategy to gather information as cleverly as possible, i.e., to gain the most diagnostic value at the least cost. We now proceed to make this more precise.

Let a diagnostic probability network (DPN) be defined as a Bayesian Belief Network where at least one random variable $H$ is a hypothesis variable (e.g., an infrastructure element such as a server) and at least one other random variable $T$ is a test variable (those variables of the model that we potentially can collect information about, i.e., a manager's opinion about the availability of an architectural element he is responsible for).

Let $\boldsymbol{H}$ denote the set of all hypothesis variables, and $\boldsymbol{T}$ the set of all test variables. Furthermore, each test $T \in \boldsymbol{T}$ has a cost function $Cost(T)$: $\boldsymbol{T} \rightarrow \boldsymbol{R}$, because usually an effort has to be made to collect data about an object. If a test is free, the associated cost is set to zero. Also, each hypothesis $H$ has an associated value function, $V(P(H))$: $[0,1] \rightarrow \boldsymbol{R}$.

Given a DPN, we have the expected value $EV$ of performing a test $T \in \boldsymbol{T}$:

$$EV(T) = \sum_{t \in T} V(P(H \mid t)) \cdot P(t) \tag{1}$$

To make an informed decision, we also need to account for the expected outcome of not performing the test $T$. We therefore introduce the expected benefit $EB$:

$$EB(T) = EV(T) - V(P(H)) = \sum_{t \in T} V(P(H \mid t)) \cdot P(t) - V(P(H)) \tag{2}$$

Still, however, no connection has been made to the cost of the test. This is remedied by the test strength *TS,*

$$TS(H,T) = \frac{EB(T)}{V(P(H))} - K \cdot Cost(T)\,, \tag{3}$$

where we have introduced the coefficient *K*, reflecting the relative importance of the expected benefit versus the cost of the test.

The definition of the value function still remains. To optimize the test selection with respect to multiple hypotheses, a function based on the marginal probability between hypotheses (rather than the joint probability) called Marginal Strength 1 (*MS*1) is introduced [10],

$$MS1(P(F)) = \left( \frac{\sum_{t \in T} (f - 0.5)^2}{0.5^2} - n_F \right) \cdot \frac{1}{n_F}\ , \tag{4}$$

where *F* is the set of all selected target states $f_i$ of the hypotheses that the user wishes to pursue and $n_f$ is the total number of target states. This test selection function is convex with a minimum at 1 – $n_f$ and maxima at 0 and 1. The value function that we are looking for now becomes the sum of the marginal strength for all target states:

$$V(P(F)) = \sum_{f \in F} MS1(P(f))\ . \tag{5}$$

## 3 Using a BBN for Decision Support in Failure Impact Analysis

### 3.1 Management Process for Failure Analysis Using a Decision Support System

Before we describe our Decision Support System (DSS), we briefly provide the organizational context of failure impact analysis in Enterprise Architecture. For this we explain a management process that generally needs to be walked through in the case of disruptions in business processes, services or other architectural elements. The management process consists of the following 5 main activities: *1. Observe failure event:* Before any measures can be planned or taken to resolve a failure, the failure event has to be observed e.g., by the responsible enterprise architect. The failure can be an actual observation or be part of a simulation in order to prepare countermeasures for future events. Tools that support the inspection and visualization of failure events are of great assistance to the observing person. The information required to detect failures may be supplied by Business Activity Monitoring (BAM) systems. *2. Set observation in DSS:* The observed failure is provided to the DSS (next section) – either manually through exporting and importing data or automatically in case of an integrated system. *3. Conduct diagnosis*: The DSS conducts a diagnosis based on the provided observation and delivers a ranked list of architectural elements. The ranking is based on probabilistic information in the DSS and displays what probable causes the observed failure may have. *4. Availability of additional observations:* The person in charge of the failure analysis checks whether additional observations are available (that could also be to check how costly additional observations would be and if that would add more valuable information to the diagnosis). If positive, then this process loops back to step 2 and sets the additional observations in the DSS. If negative, the process can proceed to step 5. *5. Initiate repairing activities according to diagnosis:* Based on the ranked architectural elements resulting from the DSS, repairing activities or projects can be planned and initiated by the responsible architect and/or project manager. The probabilistic ranking shall make the sequential ordering of activities in these projects more efficient and complete.

## 3.2 Creating the Decision Support System for Failure Analysis Based on BBN

In the following we describe our method of how to create a BBN on the basis of an EA model in order to use it for decision support in EA. Starting point for the construction is a specified EA model. The overall method consists of four main steps. Steps 1 and 2 address the creation of the BBN's structure based on the EA model, i.e., what are the variables and what relations exist between them. Steps 3 and 4 address the parameters of the BBN: during step 3 discrete values for the variables are defined for improved usability; in step 4, the built BBN structure is complemented with conditional probability distributions for all variables. The method and its steps are shown in Fig. 1. The detailed actions within all steps will be explained in the next subsections.

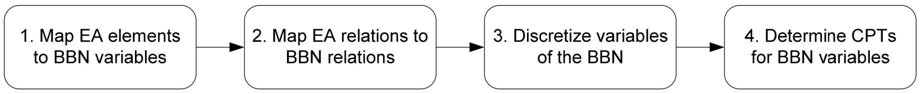

1. Map EA elements to BBN variables
2. Map EA relations to BBN relations
3. Discretize variables of the BBN
4. Determine CPTs for BBN variables

**Fig. 1.** Method for creating a Bayesian Belief Net on the basis of an Enterprise Architecture model in order to use it as a Decision Support System for failure impact analysis

### Mapping the EA Structure to the BBN Structure (Steps 1 & 2)

For obtaining the BBN structure we exploit the fact that in EA models as well as in BBN the central concepts elements and relations are used. Regarding the EA model as a graph allows us to map the EA model to a BBN. We define the following general mapping rule that constitutes step one and two of the method:

1. *Map EA elements to BBN variables:* Each EA element of the EA model maps to a variable (node) in the BBN.
2. *Map EA relations to BBN relations:* Each EA relation between two EA elements in the EA model maps to a causal relationship between two variables (nodes) in the BBN.

BBN as graphs are directed and acyclic. When mapping the EA model to the BBN, directedness and acyclicity must be preserved. Relations that violate this rule either have to be removed or be modified. For preserving acyclicity see also [12]. The result of the mapping is a DAG consisting of variables and relations representing EA elements and EA relations. Having defined the structure of the BBN, the parameters of all variables now have to be determined.

**Discretizing Variables and Determining CPTs (Steps 3 & 4)**
Variables in a BBN can, in principle, represent continuous spectra of a specific feature [9]. We focus on discrete states of BBN variables due to successful implementations in other domains [8] and the increased ease for users. Relating this to our EA context, an exemplary discretization of a variable would be: the EA element "Server" has the two discrete states "up" and "down", or, a "Service" has three possible states for response time, i.e., "fast", "moderate" and "slow". Determining the discrete states can be done conjointly with developers and end-users. All these activities can be summed up in step *3. Discretize variables of the BBN.*

Having determined the BBN structure and discretized its variables, the conditional probability distributions for all variables have to be obtained. This constitutes step *4. Determine CPTs for BN variables* of our method. In case of availability of some data, existing mathematical estimation methods can be applied, such as Maximum-Likelihood estimation (MLE) and related estimation algorithms [13]. In addition to this there are many ways to gather empirical data, see for instance [14] or [15].

The collection of data without using any mathematical estimation methods can be done applying one of the following general methods below:

Direct collection (of technical parameters of the actual EA elements; read out log files, etc.);

1. Indirect collection (of data in data bases at distributed locations that allow to draw conclusions about element dependencies);
2. User-based estimation of causal dependencies (by querying the users – via interview or questionnaire).

The manner of data elicitation may also depend on the individual collection strategy of a company. Methods one and two usually require additional technical efforts beforehand because EA elements have to be enabled to provide adequate information to the probabilistic model. Method three does not require these technical efforts. This approach collects relevant conditional probabilities through interviews with e.g., architecture experts, programmers, and system users as well as through analysis of the participating EA elements. On data collection see also [14]. Having collected all conditional probability distributions the BBN is now fully specified and may serve as decision support for failure impact analysis in EA.

## 4 Scenario-Based Analysis: Creating the Decision Support System

We apply our method to an exemplary Enterprise Architecture to demonstrate the creation of the BBN and its application as decision support for failure impact analysis in EA. The enterprise we chose is a virtual car rental agency with a service-oriented architecture and a real life implementation. The description of the business scenario and the service-oriented architecture and its implementation details can be found in [16]. The core business processes of a car rental agency are 'car reservation', 'car pick-up', and 'car check-in', i.e., returning the rental car. We analyze the business process of returning a rental car back to the agency, i.e., "Check-in car".

The business process "Check-in car" is initiated at the local service by the return of a car. For this, data about the returned car has to be requested from the system. The car is inspected in presence of the customer and claims are recorded. An invoice is then created automatically based on the rental data and the entered claim information. The monetary quantification of claims and retention is based on a claims list mapping claims to amounts. If there are no claims the car is released right away. If claims are asserted, a claim report is generated. This claim report is submitted to the claim settlement department which is responsible for contacting the insurance company and entering regulation information. At the final stage the case is documented and the car is repaired and released. The corresponding EA is modeled with ArchiMate [6, 7] and is depicted in Fig. 2. In accordance with our method defined in section 4 the following steps are executed to create the BBN.

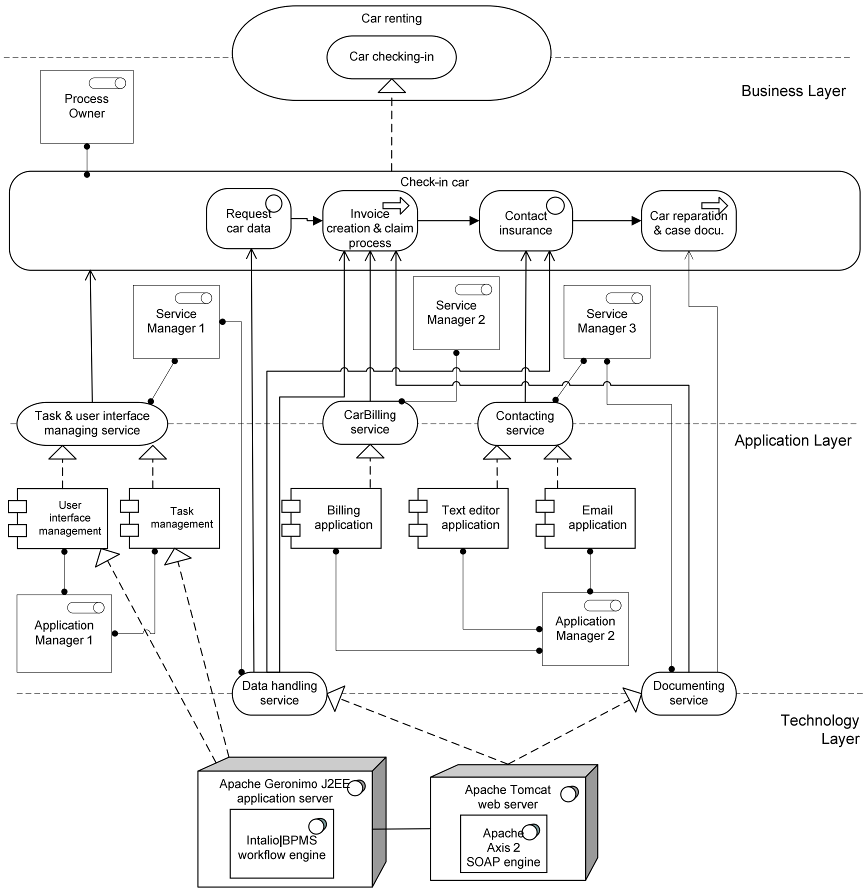


**Fig. 2.** EA model of the car rental scenario, showing all architectural elements involved in the "Check-in car" process

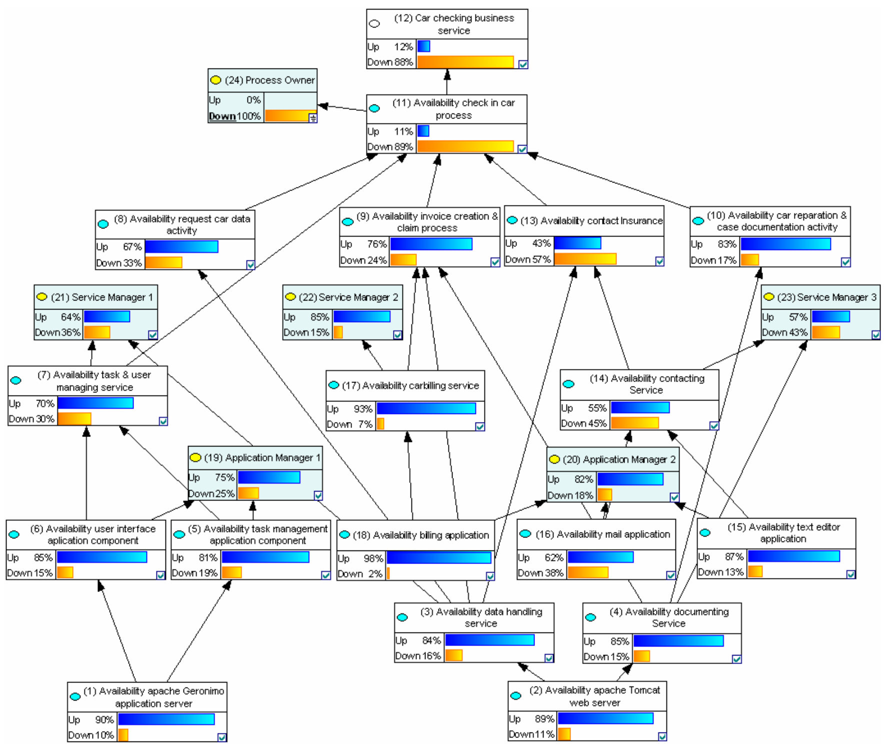

(12) Car checking business service
Up 12%
Down 88%
(24) Process Owner
Up 0%
Down 100%
(11) Availability check in car process
Up 11%
Down 89%
(8) Availability request car data activity
Up 67%
Down 33%
(9) Availability invoice creation & claim process
Up 76%
Down 24%
(13) Availability contact Insurance
Up 43%
Down 57%
(10) Availability car reparation & case documentation activity
Up 83%
Down 17%
(21) Service Manager 1
Up 64%
Down 36%
(22) Service Manager 2
Up 85%
Down 15%
(23) Service Manager 3
Up 57%
Down 43%
(7) Availability task & user managing service
Up 70%
Down 30%
(17) Availability carbilling service
Up 93%
Down 7%
(14) Availability contacting Service
Up 55%
Down 45%
(19) Application Manager 1
Up 75%
Down 25%
(20) Application Manager 2
Up 82%
Down 18%
(6) Availability user interface aplication component
Up 85%
Down 15%
(5) Availability task management application component
Up 81%
Down 19%
(18) Availability billing application
Up 98%
Down 2%
(16) Availability mail application
Up 62%
Down 38%
(15) Availability text editor application
Up 87%
Down 13%
(3) Availability data handling service
Up 84%
Down 16%
(4) Availability documenting Service
Up 85%
Down 15%
(1) Availability apache Geronimo application server
Up 90%
Down 10%
(2) Availability apache Tomcat web server
Up 89%
Down 11%

**Fig. 3.** Structure and parameters of the Bayesian Belief Network mapped from the car rental agency Enterprise Architecture model

**Step 1: Map EA Elements to BBN Variables**

Out of the 27 architectural elements in the EA model (Fig. 2), 24 elements are mapped to the BBN as variables/nodes. All mapped EA elements are depicted in the BBN in Fig. 3. For instance, the EA element “Apache Geronimo J2EE Application Server” is mapped to node “(1) Apache Geronimo application server”, the EA element “Data handling service” between Application and Technology Layer in the EA model is mapped to node “(3) Data handling service” in the BBN, and so on. It has to be noted that not all EA elements have been mapped to the BBN under the assumption that the non-mapped EA elements do not have any causal influence on other elements. This could be because they serve the mere purpose of structuring (e.g., the business service “Car renting”) or are on a very deep technological layer, such as the “Intalio|BPMS workflow engine”, whose causal relations to the hardware would not contribute to a significant better understanding from a business process management perspective. The latter supports the goal of maintaining a manageable view on the BBN.

**Step 2: Map EA Relations to Causal Relationships in the BBN**

In the car rental EA model there are mostly *directed* relations. Those relations are mapped to causal relationships in the BBN, e.g., the “Realization” relation (dotted line, arrowhead not filled) between the “Apache Geronimo J2EE application server” and the application component “Task management” is mapped to a causal relationship between node “(1) Availability Apache Geronimo application server” and node “(5) Availability Task management application component” in the BBN (as in Fig. 3).

There are also eleven *undirected* relations, i.e., those relations between business roles and business processes/services/applications (e.g., between “Process Owner” and the “Check-in car” business process). We say that the people who adopt a specific business role are able to assess the status of the architectural element they are responsible for to a certain extent, e.g., a process owner can make a judgment on the availability of a business process. This observation is based on the actual status of the element. Therefore we can map the undirected EA relations to directed causal relationships going from the element to the observer, indicating that the observation of an element by a person will usually be influenced by the actual status of the element. Having mapped the undirected relations to directed ones, we fulfill the criterion of directedness required by BBN.

Due to the strictly maintained paradigm of service-oriented architecture in the scenario, any cycles in the EA model are absent. Thus, the step in our method which removes any cycles from the BBN is not required here (for cycle removal see [12]). As opposed to the traditional top-down build-up of BBN, we model the causal relationships and nodes upwards – like a bottom-up growing tree – to maintain the resemblance with the EA model.

**Step 3: Discretize Variables of the BBN**
Each EA element that we have identified as a BBN node could be described by various features. Depending on the value of a feature of one variable, which could in principle stem from a continuous spectrum, the feature(s) of other variables are influenced. An important feature of Enterprise Architecture elements that we focus on is the *availability* of these elements [17]. We therefore apply the feature „Availability“ and define two discrete, mutually exclusive values: “up” and “down” (see also [8, 12]). We discretize the value spectrum for our single-feature variables to keep the complexity of the network nodes manageable. Moreover, these two values have shown to be generally comprehensible states of different system elements during talks with system experts and users. Also, for a user it will be generally easier to give estimations on the conditional probability distribution of *two* states of an element, rather than to estimate distributions between three or even more states. The EA model-based BBN structure is depicted in Fig. 3.

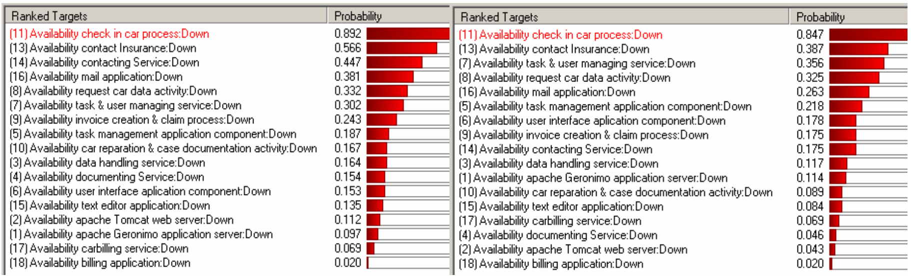
**Step 4: Determine CPTs of the BBN variables**
In this example we have not engaged in the elicitation of data to set the CPTs according to the methods proposed by for instance [13], [15] or [14]. To simulate the actual approach we randomly assigned numbers to the CPTs for the BBN in Fig. 3 above.

## 5 Using the Decision Support System for Failure Impact Analysis

We use the DSS as we have described in the management process in section 3.1 according to diagnostic analysis (section 2.2) to localize probable causes of an observed failure. To demonstrate the usage of the DSS we let the “Process Owner” observe that the business process “Check-in car” is *down*. This relates to the first step in the management process: *Observe failure event*. The next step is to set the observation in the DSS, i.e., the modeled Bayesian Belief Network, according to the observation. This is done in the GeNIe-tool by *setting the evidence* of node “(24) Process Owner” to “down” (see Fig. 3). During the third step, *Conduct Diagnosis*, the actual diagnostic analysis based on the one observation and the BBN model is executed (in GeNIe, the *Test Diagnosis* button initiates this). In our example the cost of observing the availability status of architectural elements is still set to zero, assuming that querying service or application managers is an effortless task. This means that additional observations will always contribute to a better diagnosis since the observation is for free. For a more realistic representation in the future the costs of asking people or having people to publish their observations need to be introduced in the model.

After the observation, diagnostic analysis calculates the a-posteriori probabilities of all target nodes. In Fig. 4, left (screenshot taken from GeNIe [11]), the diagnostic results are depicted as a list of ranked (ranked according to the probability of being *down*) target nodes. The ranked target node list starts with nodes (11): 0.892, (13): 0.566, (14): 0.447… and so forth. This information points an enterprise architect to architectural elements that ought to be attended to immediately – those having high probabilities of unavailability and being the possible cause of the process failure – compared to those elements with lower probabilities. Based on the given information this would be the best order of activities in a repair project.

The order of activities can change when more information of the status of architectural elements is known. In addition to the formerly observed business process “Check-in car” being *down*, we let “Service Manager 3” observe the services “Car documenting service” and “Contacting service” under his responsibility being available, i.e., *up* (Fig. 4, right). The ranking of target nodes significantly changes, after having this additional observation. For instance, node *(14) Availability contacting service: down* had a probability of 0.447. Taking into account the new observation, the probability of node (14) being *down* drops to 0.175. This provides useful information to the enterprise architect who can now initiate differently ordered activities to repair the failure.



| Ranked Targets | Probability |
|---|---|
| (11) Availability check in car process:Down | 0.892 |
| (13) Availability contact Insurance:Down | 0.566 |
| (14) Availability contacting Service:Down | 0.447 |
| (16) Availability mail application:Down | 0.381 |
| (8) Availability request car data activity:Down | 0.332 |
| (7) Availability task & user managing service:Down | 0.302 |
| (9) Availability invoice creation & claim process:Down | 0.243 |
| (5) Availability task management application component:Down | 0.187 |
| (10) Availability car reparation & case documentation activity:Down | 0.167 |
| (3) Availability data handling service:Down | 0.164 |
| (4) Availability documenting Service:Down | 0.154 |
| (6) Availability user interface aplication component:Down | 0.153 |
| (15) Availability text editor application:Down | 0.135 |
| (2) Availability apache Tomcat web server:Down | 0.112 |
| (1) Availability apache Geronimo application server:Down | 0.097 |
| (17) Availability carbilling service:Down | 0.069 |
| (18) Availability billing application:Down | 0.020 |

| Ranked Targets | Probability |
|---|---|
| (11) Availability check in car process:Down | 0.847 |
| (13) Availability contact Insurance:Down | 0.387 |
| (7) Availability task & user managing service:Down | 0.356 |
| (8) Availability request car data activity:Down | 0.325 |
| (16) Availability mail application:Down | 0.263 |
| (5) Availability task management application component:Down | 0.218 |
| (6) Availability user interface aplication component:Down | 0.178 |
| (9) Availability invoice creation & claim process:Down | 0.175 |
| (14) Availability contacting Service:Down | 0.175 |
| (3) Availability data handling service:Down | 0.117 |
| (1) Availability apache Geronimo application server:Down | 0.114 |
| (10) Availability car reparation & case documentation activity:Down | 0.089 |
| (15) Availability text editor application:Down | 0.084 |
| (17) Availability carbilling service:Down | 0.069 |
| (4) Availability documenting Service:Down | 0.046 |
| (2) Availability apache Tomcat web server:Down | 0.043 |
| (18) Availability billing application:Down | 0.020 |

**Fig. 4.** Left: Diagnostic results as a list of ranked target nodes after one observation, and Right: Additional observation: differently ordered ranked target nodes after two observations

## 6 Conclusion

We have proposed a DSS based on a Bayesian Belief Network to address one important concern of today's Enterprise Architects, i.e., conducting failure impact analysis and diagnosis in EAs. The BBN was created based upon an architectural model of an Enterprise Architecture, i.e., the knowledge about causal dependencies between actual architectural elements was exploited to create the BBN nodes and relationships to capture the uncertainties. Particularly for architectures with growing complexity, those approaches which capture this uncertain knowledge and still allow reasoning on it seem suitable. This is supported by situations in which the actual states of system elements cannot be determined, but only *observed* by managers using there experience.

We have designed a detailed method to create the Bayesian Belief Network-based DSS consisting of the mapping of EA model elements to a BBN and eliciting all its required structural features and probabilistic parameters. To demonstrate the general feasibility of the approach we created a DSS for an EA of a car rental agency and conducted an exemplary failure diagnosis in it, giving managers valuable information about what elements could be the probable causes. Even though our exemplary scenario is a complete service-oriented Enterprise Architecture, more complex structures of EA models (e.g., those including loops, non- and bi-directed relations) in other enterprises are definitely possible. In these cases additional mapping rules – from the EA modeling language to BBN parameters – need to be introduced in order to remove irregularities and create a formally correct Bayesian Belief Network.

In further work we will concentrate on the elicitation of probabilities to populate the CPTs of the Bayesian Belief Network. The spectrum of possibilities, e.g., leveraging expert opinions on system element behavior in contrast to exploring automatic ways of using monitoring information about system availability and other qualities, needs to be analyzed considering the trade-off between expressiveness and cost of elicitation.

## References

1. The Open Group: The Open Group Architecture Framework (TOGAF), version 8 Enterprise Edition. The Open Group (2005)
2. Department of Defense Architecture Framework Working Group: DoD Architecture Framework Version 1.0 Department of Defense, USA (2004)
3. Schekkerman, J.: How to survive in the jungle of Enterprise Architecture Frameworks. Trafford, Victoria, Canada (2004)
4. Friedman, N., Linial, M., Nachman, I.: Using Bayesian Networks to Analyze Expression Data. Journal of Computational Biology 7, 601–620 (2000)
5. Buckl, S., Ernst, A.M., Lankes, J., Matthes, F.: Enterprise Architecture Management Pattern Catalogue, Version 1.0. Technische Universität München, München (2008)
6. Buuren, R.v., Hoppenbrouwers, S., Jonkers, H., Lankhorst, M., Zanten, G.V.v.: Architecture Language Reference Manual. Telematica Instituut, Radboud Universiteit Nijmegen (2006)

7. Jonkers, H., Lankhorst, M.M., Buuren, R.v., Hoppenbrouwers, S., Bonsangue, M.M., Torre, L.W.N.v.d.: Concepts For Modeling Enterprise Architectures. Int. J. Cooperative Inf. Syst. 13, 257–287 (2004)
8. Lauría, E.J.M., Duchessi, P.: A Bayesian Belief Network for IT implementation decision support. Decision Support Systems 42, 1573–1588 (2006)
9. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification. Wiley Interscience, Hoboken (2000)
10. Jagt, R.M.: Support for Multiple Cause Diagnosis with Bayesian Networks. Vol. M. Sc. Delft University of Technology, the Netherlands and Information Sciences Department, University of Pittsburgh, PA, USA, Pittsburgh (2002)
11. Decision Systems Laboratory: GeNIe (Graphical Network Interface). vol. 2008, University of Pittsburgh (2008)
12. Tang, A., Nicholson, A.E., Jin, Y., Han, J.: Using Bayesian belief networks for change impact analysis in architecture design. Journal of Systems and Software 80, 127–148 (2007)
13. Neapolitan, R.: Learning Bayesian networks. Prentice-Hall, Inc., Upper Saddle River (2003)
14. Keeney, R.L., Winterfeldt, D.v.: Eliciting Probabilities from Experts in Complex Technical Problems. IEEE Transactions On Engineering Management 38 (1991)
15. Woodberry, O., Nicholson, A.E., Korb, K.B., Pollino, C.: Parameterising Bayesian Networks Australian Conference on Artificial Intelligence. Springer, Heidelberg (2004)
16. Holschke, O., Gelpke, P., Offermann, P., Schröpfer, C.: Business Process Improvement by Applying Reference Process Models in SOA - a Scenario-based Analysis. Multikonferenz Wirtschaftsinformatik. GITO-Verlag, Berlin, München, Germany (2008)
17. International Standardization Organization and the International Electrotechnical Committee: ISO/IEC 13236 - Information technology — Quality of service: Framework. ISO/IEC (1998)