Combining Network Measures and Expert Knowledge to Analyze Enterprise Architecture at the Component Level

Alixandre Santana[1], Daniel Simon[2], Kai Fischbach[3], Hermano de Moura[1]
[1]Center of Informatics, Federal University of Pernambuco, Brazil
[2]Scape Consulting GmbH, Germany
[3]University of Bamberg, Germany
atfs2@cin.ufpe.br

***Abstract*— Enterprise architecture (EA) network analysis has been attracting researchers' attention lately. The main source of information is the structural components, including the relations among them and how they might be structurally arranged. These relations are studied to generate valuable information for EA professionals. However, to the best of our knowledge, ours is the first attempt to combine structural information with a second source of information: expert´s tacit knowledge. We believe combining these sources employing two new methods – what we call cognitive-structural diagnosis analysis and attribute check analysis – can refine the expert´s knowledge about the architecture. To demonstrate these methods´ feasibility, we apply them with two application architecture datasets collected in two different organizations. We also offer a classification schema for enterprise architecture network analysis at the component level, our focus. Our conclusions indicate that the cognitive-structural diagnosis analysis method minimizes analysis subjectivity while validating important components and also suggesting important structural ones to be further analyzed by experts. The attribute check analysis offers further contributions by helping in the investigation of particular attributes of applications in important architectural positions.**

Keywords*— *Network measures, Enterprise Architecture, analysis, method.

I. INTRODUCTION

Enterprise architecture (EA) as an intertwined system of, for example, strategic goals, business processes, applications, and infrastructure components [1] subject to a variety of inherent relations and dependencies. The management of all these components, their dependencies, changes, and evolution allows changes be coordinated and aligned with the mid-term to long-term company objectives [2]. This is not trivial task; rather, it is a challenge for EA specialists. At companies that deal with vast information technology landscapes supporting several business processes, the task of gleaning additional value from current EA models is made even more complex by the lack of suitable analysis tools [3].

EA analysis has, though, attracted researcher attention in the last decade[3]. The literature contains several paradigms for EA analysis, such as probabilistic relation models [4], EA intelligence [5], complexity management [6], ontology-based analysis [7], and network analysis [8]. In the latter, researchers model EA components as networks/graphs and apply measures and/or methods grounded in network analysis [9] to identify important structural components, considering the relations among them. We label this paradigm EA network analysis.

The idea behind EA network analysis is then to extract valuable information from the EA component's relationships to support the analytic process. A similar analytic principle has also been used in other areas such as product engineering to analyze parts, components, and relationships at a more granular level, checking aspects like network density, central components, and modularity [10, 11]. Besides, some works do already exist that address EA network analysis to some extent: [12], aiming to define principles of complexity for EA; [8], studying the impact of changes in EA; [6], working with structural complexity; and, finally, [13], proposing the hidden structure method to classify application architecture in different topologies with their respective implications for cost of change analysis. However, the application of network measures in EA analysis still seems to be quite expandable [3, 14]. In this paper, we aim to contribute in this direction.

At a general perspective, [15] is the only work that has tried to classify the EA analysis research field and covers all the paradigms mentioned before. The authors present a schema with five dimensions: body of analysis, time reference, analysis technique, analysis concern, and self-referentiality. With the analysis techniques dimension, they define three categories: expert-based, rule-based, and indicator-based. They point out that the expert-based analysis techniques depend on the experience and expertise of the executing person. That is, one or more experts – the enterprise architects – analyze properties of the EA along appropriate architecture views, such as, for example, reports or graphical visualizations. Reference [15] found only two papers in this category, which they discuss in their literature review. In these two cases, the experts analyze the EA models based on their tacit knowledge and used no analytical tools or other knowledge sources in drawing their conclusions. This scenario is depicted in Cycle I in Fig. 1. In addition to the expert-based techniques, the work of [15] also discusses two other types: rule-based, performed at an increased level of formalization and potentially automated (e.g., EA analysis based on ontologies); and indicator-based,

This work was partially supported by CAPES through the Doctorate 'Sandwich' Program (PDSE,) process 99999.003702/2014-06, Brazil.

978-1-4673-9885-5/16/$31.00 ©2016 IEEE

used to assess quantitatively architectural properties such as complexity or costs. The authors make no mention of the possibility of simultaneously combining two types of analytic techniques, as we aim to do in the present work.

This is essentially what is done in the Cycle III, after the Cycle II modeling of EA as a network. Considering the EA network analysis paradigm only, there were no analysis strategies that combined at the same time expert (i.e., knowledge about components) and structural information [3]. This analysis scenario is represented in Cycle IV in Fig. 1.

We aim to minimize the subjectivity of informal analysis done by experts by combining it with components' structural information (indicator-based analysis). To that end, we introduce and evaluate two methods for EA network analysis at the component level: cognitive-structural diagnosis analysis and attribute checking. We use cognitive-structural diagnosis analysis to provide complementary information for expert-based assessments within the EA decision process. With attribute checking, we make considerations about components with certain attributes of interest that occupy important structural positions. Our research question is as follows: How can we combine expert information with network measures to support EA domain analysis? In addition, our paper provides a foundation for a normative classification schema of EA network analysis at the component level.

In this paper, we make two contributions: aid cognitive-structural diagnosis analysis by assessing the architect's perception of components together with network measures' output, which would allow experts to validate or refine their knowledge about their EA components; and introduce a classification schema for EA network analysis methods at the component level as a guide for further research.

The remainder of this paper is structured as follows. Section II presents the theoretical background of EA, network analysis, and our research classification schema. Section III then explains our two methods for EA network analysis. Section IV presents and discusses our results, using two small cases. Finally, Section V presents our conclusions and considerations for future research.

II. THEORETICAL BACKGROUND

A. Enterprise architecture

EA is defined in a variety of ways; we adopt the definition proposed in the literature review of [12], which is also supported by [16]: EA is a system formed by four subsystems: Business (or Business Architecture [BA]), Data/Information (or Data Architecture [DA]), Application (or Application Architecture [AA]), and Infrastructure (or Infrastructure Architecture [IA]). In this paper, we model one of these EA subsystems (or EA layers/domains) - the application layer - as networks/graphs.

B. Enterprise Architecture as a complex network

The study of networks in many disciplines, ranging from computer science and communications to sociology and epidemiology, has experienced a strong upsurge in recent years [17]. In the context of network theory, according to [9], a graph or network is a set of components (nodes or vertices) and links (edges or relations).

This representation has several interesting properties, as it can encapsulate local neighborhood relations as well as global characteristics of the data [18]. In the prominent work of [9], the author uses measures from the social network analysis field to investigate how a component (individual) is immersed in a (social) structure, how this structure rises from micro-relations among components, and how it can affect specific context variables (e.g., individual performance). In the view of [14], these network measures applied to an EA modeled as a network could represent more reliable indicators to describe EA structure, helping experts guide its analysis process and evolution. The network measures applied to our two datasets are presented in Section III.

With this conception of EA modeled as a network, we can choose different levels of analysis at which to apply network measures [19]. First is the actor or component level, at which it is common for researchers to apply centrality measures such as degree or betweenness and use these values to identify central components in terms of structure [9]. The second possibility is to analyze the component and its neighborhood; for instance, we might perform a dyadic or triadic analysis. Another possible perspective is at the modular or group level, which might identify cliques or clusters.

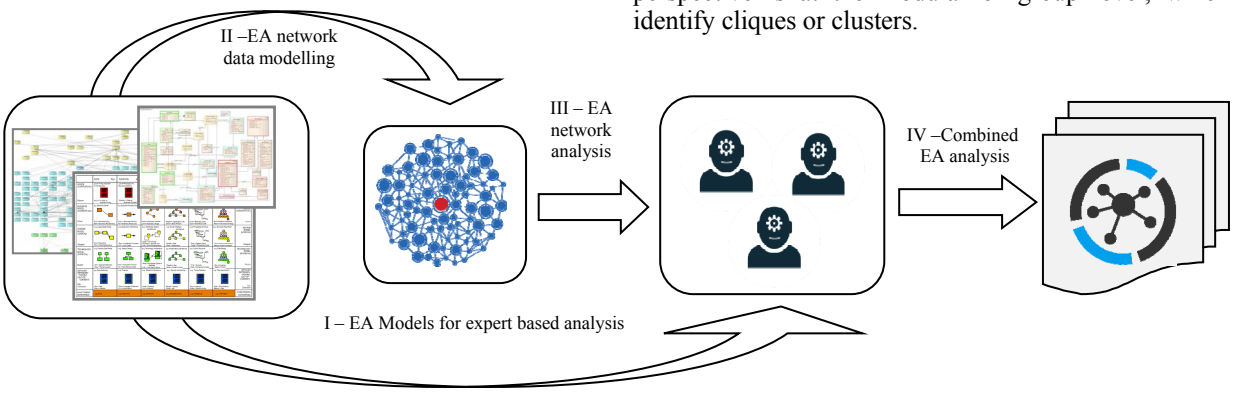

Fig. 1. Combined EA analysis. Cycle I: EA models are analyzed with expert-based techniques; Cycle II: EA models are converted into EA network data; Cycle III: EA network analysis is performed using network measures; Cycle IV: Combined EA analysis)

Finally, looking at the network level, some measures such as connectedness, diameter, centralization, density, and clustering coefficient [9] can depict attributes of an entire network and thus, in our context, an entire EA domain. In this paper, we focus on the EA network analysis at the component level.

C. *Enterprise architecture network analysis at component level*

In the EA context, analysis processes should extract relevant information from the architecture models to provide solid and structured information that will be valuable for making business and IT decisions [20]. In this sense, we suggest network measures can function as support tools for architects, helping them analyze their practical concerns. In an organizations' daily life, given limited resources, it is important that architects identify a subset of these components in the overall architecture to control or actively manage. To some extent, experts may be able to identify intrinsically important components that may, for example, support critical business tasks or have a large number of users and are thus often the focus of decision makers. In practice, these checks are often completed using rough and subjective estimations. This is where network measures may offer help.

Reference [3] identifies 67 network analysis initiatives (network measures such as degree centrality, cluster algorithms, and other methods) applied in the EA context. We took that set of initiatives and, to classify them according to the strategies of analysis adopted by the researchers, created a classification schema comprising four categories (depicted in Fig. 2), one of which is a main subject of the present work: the expert-combined analysis. By unimodal domain analysis (Fig. 2), we mean an analysis of a unimodal network (or one EA domain). A domain is any EA subsystem, as discussed in Section II.A (e.g., application architecture [AA]), which has only one kind of component – i.e., applications). The study of modeling and analysis methods for bi- or multi-modal networks or networks comprising two or more EA domains (e.g., networks with applications that support business processes) is beyond the scope of the present work, as is the analysis at dyadic, group, and network levels. In the following, we discuss the five categories of EA analysis at the component level.

1) Structural analysis. This category comprises works that try purely to identify architectures' structural components. In this category, papers highlight components with structural roles acting like bridges or hubs or are connected with other well-connected ones. Input for this analysis is limited to information about the relations among the components. Statistical models are excluded. Reference [21], who investigate the component's centrality degree, and [22], who apply degree and betweenness centrality in their analysis, are examples of such studies.

2) Statistical analysis. This category comprises research aimed at building statistical models or finding correlations based on network measures. Reference [23], for example, uses discriminant functions based on several network measures to classify business processes – core, information intensive, flexible, or automatable processes – in categories according to type.

3) Proxy based analysis. The independent variables are composed of network measures only and the researcher is interested in using network measures as proxys for EA concens such as cost of change. The work of [13] is an example of such an approach.

4) Complementary analysis. This category includes papers that use two complementary types of information as input to investigate an EA concern. Its main difference with proxy-based analysis is the use of a component's properties (e.g., whether an application is self-developed, cloud-based, or off-the-shelf) together with network measures. For instance, [24] uses application attributes such as real-time requirements and operating cost together with outdegree centrality to evaluate the gap between the current and future EA states. In the work of [25], the Process-PageRank (PPR) algorithm ranks business process for improvement using degree centrality and a need for improvement (a business process attribute) to calculate the network-adjusted need for improvement. So far, there has been no discussion regarding the extent to which the structural information could be used together with expert knowledge (e.g., top-ranked applications perceived by the expert). Experts might want to know, for example, whether applications that occupy important structural positions are hosted in house or

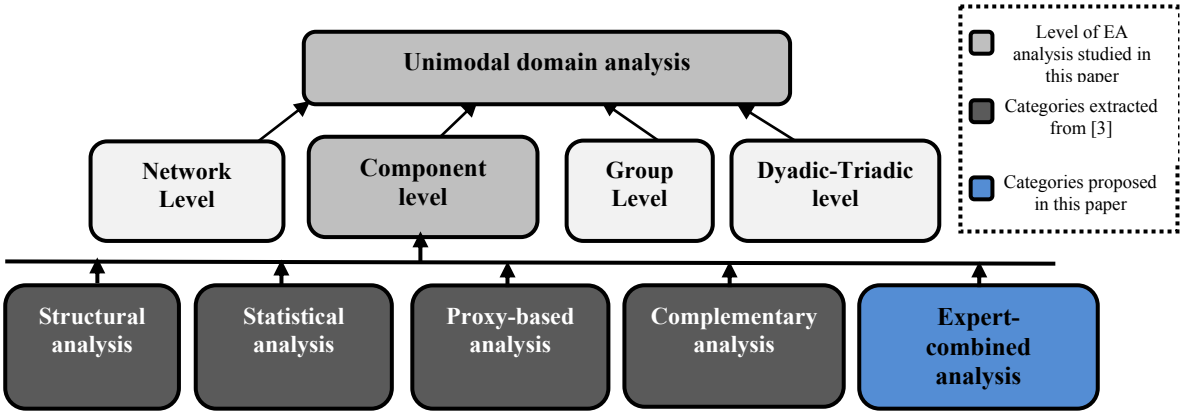

Fig. 2. A classification schema of methods for EA network analysis at the component level.

are outsourced. This discussion leads us to the next category of analytic methods.

5) Expert-combined analysis. This category includes papers that aim to investigate how the value extracted from network measures can be combined with tacit expert knowledge. In this paper, we use this tacit expertise combined with information obtained from structural analysis of the application architecture. We assume that one can use non-functional characteristics of EA components, such as business criticality or strategic relevance as judged by experts, with structural information. A second strategy under this category is to check whether a specific attribute value is present in components that occupy important structural positions. For example, architects might be interested to know what kind of software (e.g., self-developed, cloud-based, or off-the-shelf) is running on applications with the highest eigenvector values. Section III.C details this method. Table I presents our classification schema for EA network analysis at the component level, with the characteristics of each proposed category summarized to facilitate discussion.

Of course, the list of categories in Table I is not exhaustive. However, this schema helps us contextualize the kinds of methods we are proposing among other EA network analysis possibilities. Section III that follows describes the methodological aspects for two proposed expert-combined methods and presents their validation with empirical data.

III. METHODOLOGICAL ASPECTS

This work can be classified mainly as exploratory and applied research. As for research methods, we have adopted a design science research approach [27]. We use the outputs of applying quantitative measures from network theory along with information provided by EA experts from two sample cases to design two artifacts for performing EA network analysis based on the combined data. More specifically, we compare the main application nodes identified by the measures with those pointed out manually and subjectively by the experts. We refer to this artifact as *cognitive-structural diagnosis analysis*. Our hypothesis is that network measures will confirm the majority of applications perceived as important by the experts and thus validate their perceptions, while also shedding light on potentially important components that were at first overlooked by the experts. This method is explained in Section III.C. We also operationalize a method to verify which kinds of components occupy important structural positions in the network, while observing these components' features. We call this second artifact *attribute check analysis.* Both artifacts are validated empirically. The rationale behind these two methods is that when an enterprise has a set of applications like the one depicted in Fig. 3, which represents the architecture of one of our cases, it may be intuitive for a person to point out some applications based on their license´s price, number of users, or any other intuitive heuristic. However, as can be seen from the colored components, different centrality measures can highlight different structural aspects that are not possible to detect simply by "eyeballing" the network. We believe that this additional source of information may generate important insights for the expert.

In the following, we present the two cases, the used network measures, and our approaches to combine expert knowledge with structural information.

A. The two datasets analyzed

We apply our methods to two application architectures in two different organizations: the first operates in the media industry and employs several thousand people at its headquarters in Germany and about twenty international sites; the second one is a large multi-industry player also headquartered in Germany and operating in a few other European countries.

The EA management function of the first organization was established more than five years ago. We were provided a dataset with 236 applications (components), but only 201 with in-degree and out-degree values greater than zero, representing nodes with connections (around 85% of the original dataset). The 35 isolated applications were removed since the used network measures can obviously provide valuable information only when components are connected. The second organization's EA management function is about two years old. The corresponding dataset originally had 377 applications. Removing the isolated ones left 228 applications, or about 60 percent of the original dataset. We refer to the resulting datasets from these two organizations as Dataset01 and Dataset02, respectively (due to anonymity reasons, we cannot be more specific and, e.g., provide network figures with named nodes in this paper). Both datasets come with the same semantics for the relations between the applications, which is that one "application provides-data-to another application." As depicted in Fig. 4, if an application A sends data to a second application B, we link A with B, using a directed and unweighted relation from A to B. The first two cycles of the EA combined analysis proposal (see Fig. 1) are related to external knowledge acquisition and network data modeling activities, respectively. Regarding the first cycle, we obtained the data with the attributes´ values (and components' inter-relationships) in datasheets.

TABLE II. EA NETWORK ANALYSIS AT THE COMPONENT LEVEL: CLASSIFICATION CATEGORIES

Network measure	Additional knowledge type		Statistical based or supported	EA concern
	Expert	Component attribute		
Statistical analysis	Yes	Yes	Yes (Strongly)	Several
Structural analysis	No	No	No	Only structural
Proxy-oriented analysis	No	No	No	Several
Complementary analysis	No	Yes	No	Several
Expert combined analysis	Yes	Yes	Yes/No	Several

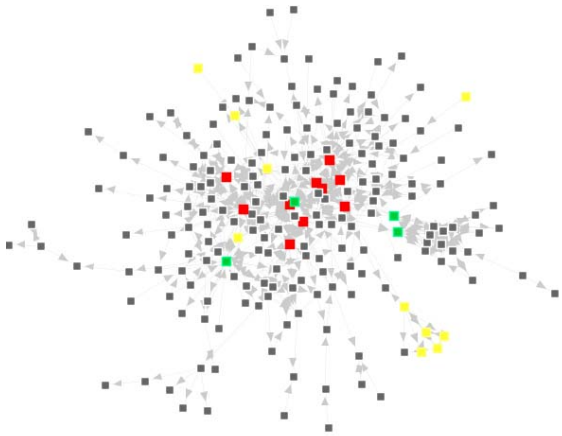

Fig. 3 An application architecture modeled as a network (red nodes represent the 10 highest eigenvector centrality values, yellow ones have the 10 highest closeness, and green ones have the 10 top betweenness values).

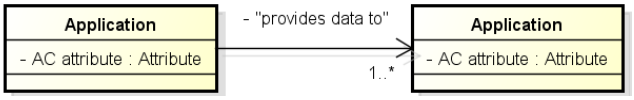

Fig. 4 Semantical modeling adopted for the two datasets

Table II presents the attributes of the applications we analyzed. Some attributes in the table require further explanation. For example, in Dataset01, we refer to "type of development." Its possible values are "standard" (representing purchased commercial software), "self-developed" (internally developed applications), and "others." In Dataset02, "core application" is a qualitative attribute that materializes the expert´s perception about whether an application belongs to the core set of the organization's applications and thus has top priority from a business point of view. In the middle column, we describe the ratio of maintained values for each component attribute or relation.

Subsequently, we modeled manually the relations among applications with help of the ORA tool[1]. In general, this task might benefit considerably from a software plugin capable of converting data from architecture models to networks. Once the network models were finished, we used the UCINET software to run the chosen network measures in Cycle III(explained in Section III.B). In Section III.C, we introduce the methods used in Cycle IV.

B. *Network measures used*

Table III shows the set of network measures we use, chosen according to two criteria: (1) they are well-known centrality measures often employed in related studies; and (2) they address different aspects of the network. Reference [26] argues that centrality degree, betweenness, and closeness are related to different aspects of centrality such as control, brokerage, or independence. Eigenvector centrality is the fourth measure of centrality. Hence, the measures of centrality then offer four different ways of identifying how network structure might differentiate the roles of components. Our two networks are two unimodal networks formed by (only) applications components. We used the UCINET software to calculate those measures with our data.

C. *Approaches to EA analysis at component level*

As discussed in Section II.C, we present a new category for EA network analysis: expert-combined analysis. In addition, to demonstrate the idea behind the category, we propose two methods – explained below – that both try to take advantage of combining structural and expert knowledge.

1) Cognitive-structural diagnosis analysis

As pointed out earlier, EA analysis still depends strongly on human expertise due to the lack of development of proper tools [3]. In the case of vast IT landscapes, human perception – which is limited and subjective – might decrease the confidence in the quality and coverage of EA analysis. Applications that support strategic business processes, for example, are natural candidates to be listed by the experts as important EA components. Experts might not notice other applications that do not fit these criteria, despite that the applications play important structural roles in the information flow. Hence, considering EA analysis as a complex process, one might want to use additional knowledge sources in order to add confidence to the analysis. Combining the information generated by applying network analysis measures (structural criteria) together with expert knowledge (subjective, in nature), we believe we can design a more robust method for EA analysis. One important assumption for this method is that network analysis and expertise are both important factors for the decision of identifying critical components. To apply this method, given a component C (e.g., application, business process), we first identify an "attribute of interest" (AI) (e.g., application's availability, business process strategic value) or any other EA analysis concern for this component.

TABLE II. CHARACTERIZATION OF THE TWO DATASETS

Dataset01	**% Available data**	**Possible Values**
Total number of applications: 236	81% (201/236) (not isolated applications)	
Total number of relations: 482	100	
Number of attributes: 2		
Type of development	84.6	Others; Self-developed; Standard
Availability	25.3	Normal; High; Very high
Dataset02	**% Data available**	**Possible Values**
Number of components: 377	60% (228/377) (not isolated applications)	
Number of relations: 637	100	
Number of attributes: 2		
Business criticality	88	Very critical; Critical; Not critical; Blank
Core application	100	Yes; No

[1] http://www.casos.cs.cmu.edu/projects/ora/

TABLE III. NETWORK MEASURES USED IN THIS RESEARCH AND THEIR CONTEXTUALIZATION TO THE CASE STUDIES

Network measure	Contextualization of the measure
In-degree	In-degree centrality (applicable as our networks are directed ones) indicates the degree to which applications depend on data provided by others,
Out-degree	Out-degree centrality (again, applicable as our networks are directed ones) indicates the degree to which applications provide data to others.
Betweenness	Betweenness can identify applications that work as bridges and connectors [9] in the application network.
Out-closeness	Closeness has a significant effect on how coupled a component is to all other components, not just to those that integrate with it directly [8]. Outcloseness, in a directed network, can indicate the range of reachability of an application to the entire application network. We apply out-closeness in our datasets.
Eigenvector	Eigenvector centrality can be considered an extension of degree centrality that privileges components connected to other well-connected components [9]. Thus, this measure can identify applications that together constitute global structural points.

Second, we store the network measure's outputs for the components in the matrix "network measure values" (NMV) (e.g., eigenvector and degree centrality values for all the components). Thus, we investigate whether there is a correlation, at least at a moderate level (rho >0.3, Spearman correlation, $p<0.05$), between AI values and those in the NMV matrix. If there is no correlation, one should review whether this is due to the fact that the given attribute of interest is naturally independent from the component's structural position or that the expert might have wrongly missed including a component's structural context in the analysis altogether. If there is at least a moderate correlation, we select the number nc of components with the highest NMV values (we call these outliers, which form a set with size=nc).

From this outliers set, the goal is to find the components that might validate the expert's perception about "important components" (true positives) and/or add new ones not perceived previously by experts (false positives). Thus, we list those components for further evaluation and analysis report. Typical questions that may be answered with this method are:

- Can the "*x*" highest-ranked network measure values (outliers) identify the "*y"* most-important components as per a certain attribute (like *strategic value*)?
- Can the "*x %*" highest-ranked network values identify the "*y*" components with "*high*" and "*very high*" availability (attribute) values?

For both questions, two parameters need to be defined. One is the amount "x" of selected components from the outliers' set to be used in the cognitive-structural diagnosis analysis. "x" can be an exact amount or a percentage of the outliers' set ("x%"). The second is "y": the amount of components from the set of components of interest (components already identified as important ones by the expert). Thus, we compare the subsets of x and y components and compute recall and precision values, which results in the four areas Fig. 5 depicts. Precision and recall are not considered as the central performance indicator for the method, but they are used to acknowledge that network measures overlap with the expert knowledge at a certain level. However, we are more interested in the groups generated with the F- measure: true positives, representing reinforcement of the components' importance from a structural and expert-based (non-structural) perspective (Fig. 5, Area 3); false positives (Fig. 5, Area 1); and false negatives (Fig. 5, Area 2).

The true positives represent components classified by experts as "strategic," "critical," or any other intrinsic value of interest that had its structural importance confirmed. The false positives represent components with no particular value for the attribute of interest (e.g., "core application" = no) but detected by network measures as having structural importance; they must be discussed further to review and validate the expert's classification with respect to the detection of this structural importance detected (which may represent valuable information). The next group represents components with no structural relevance to the network measures, but is important for the experts for different reasons (false negatives). Table IV contextualizes the recall and precision results. The pseudo algorithm in Fig. 6 summarizes the method.

2) Attribute check analysis.

Our assumption is that combining each network measure's contribution will allow us to identify components that play the different strategic roles indicated in Table III. In this sense, the goal of this method is to take the outliers among the components (i.e., those with highest values for any centrality measure) and check whether they have specific attributes or features of interest, such as type of software license (e.g., SaaS, self-developed), type of application hosting (e.g., cloud, internally, etc.), type of technology (mainframe, client-server, distributed, etc.), vendor (e.g., IBM, Oracle), and so on (see Fig. 7, which presents the algorithm for the *attribute check analysis* method). Any of these features might influence an expert in the case of changes, acquisitions, mergers, and so on.

More specifically, this method would answer a typical question such as: How are "individual" (self-developed) and "standard" (procured) software distributed over the most important (i.e., top 10%) structural components? One could also use this method to verify the presence or dominance of a specific software supplier in key structural applications. Another possibility would be to create triggering events that might warn experts when key components assume specific values for a target attribute (e.g., availability).

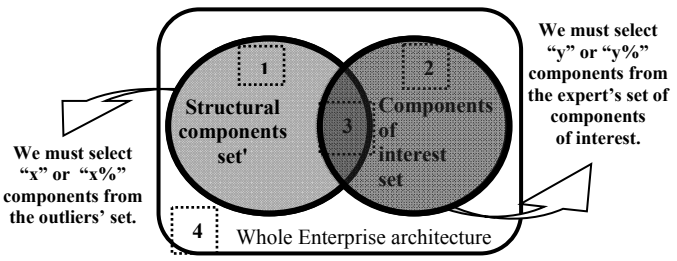

Fig. 5 Cognitive-structural diagnosis analysis: defining the samples to compute recall and precision.

TABLE IV. RECALL AND PRECISION CONTEXTUALIZED IN EA NETWORK ANALYSIS.

Attribute: Business Criticality; Values of interest : Very critical and Critical Used network measures: e.g.: outdegree and betweenness		
	Important to the expert	**Not important to the expert**
Structurally important	**True positives**	**False positives**
Not structurally important	**False negatives**	
Recall =	How many relevant items are selected? Recall = Region 3/(Region 3+Region 2)	
Precision=	How many selected items are relevant? Precision = Region 3 / (Region 3+Region 1)	

Cognitive-structural diagnosis analysis – algorithm

1: **Set** m = number of applied network measures
2: **Set** c = number of components in the application architecture
3: **Set** y = number of components of interest (classified by the experts)
4: **Set** $NMV_{c,m}$ = the matrix to save m network measures values for all the c components
5: **Set** CNMc = the list of network measures correlated with an attribute of interest
4: **For each** network measure NM_i (NM_1=eigenvector, NM_2=closeness centrality,…, NM_m)
6: **For each** component C_j (C_1=APP_1, C_2=APP_2…, C_n=APP_c)
7: $NMV_{i,j}$ ← value of NM_i for C_j
8: **From** the component type C, **select** an attribute of interest AT_C
9: **For each** network measures $NM_{i(i=0, m)}$
10: **If** AT_C and NMV_i are correlated (spearman) **then**
11: CNM_c ← CNM_c + NM_i
12: **Calculate** Recall and precision (set of "y" components of interest **and** set of "x" or "x "%-highest outliers for each CNM)
13: **Return** true positives set, false negatives set and false positives set

Fig 6. Cognitive-structural diagnosis analysis method – pseudo algorithm.

Attribute check analysis – algorithm

1: **Set** m = number of applied network measures
2: **Set** c = number of components in the application architecture
3: **Set** y = number of components of interest (classified by the experts)
4: **Set** $NMV_{c,m}$ = the matrix to save m network measures values for all the c components
5: **For each** component Ci (i=0, nc)
6: **For each** measure NMj (j=1,m)
7: $NMV_{c,m}$ ← Value of NMj for Ci
8: Calculate the x or x% of outliers from $NMV_{c,m}$
9: **From** the component type C, **select** an attribute of interest AT_C
10: **Check** the particular values for ATc among the outliers

Fig 7. Attribute check analysis method – pseudo algorithm

In general, this method is suitable for contexts in which the attribute of interest does not have at least a moderate correlation with the network measures. The method's output is descriptive and qualitative information that may enhance the expert's knowledge about the EA and support architects in their decision processes. In the end, though, the conclusions are the responsibility of the expert.

All in all, we have introduced two exploratory and generic approaches that might be applied to any set of network measures and any target attribute when the expert needs or wants to compare his tacit knowledge with the structural dimension of the architecture. It is worth mentioning that past work using network analysis did not explore this potential, focusing mainly on the pure structural value of the EA components.

IV. RESULTS AND DISCUSSION

For Dataset01, we run the cognitive-structural diagnosis analysis method. We use m=5 (five network measures described in Table III), c=201 components, and y=5. We thus aim to find five components with a particular value ("very high" and "high") for the attribute of interest availability. First, we analyze the correlation between the attribute and each of the 5 network measures. There is a moderate correlation ($0.7>rho>0.35$, $p<0.05$) for outdegree, eigenvector, and betweenness centralities. We then try to find the 5 (y=5) components of interest among the 5 (x=5) highest outliers of each measure among the 201 components (c=201). Since this sample is small, corresponding to less than 3 percent (5/201) of the application architecture, after applying the method we find only one component that matches with the components of interest. However, when we expand our analysis to select the 10 percent (x=10%) that are the highest outliers for each measure, we find all 5 components of interest. Finally, we calculate recall and precision and obtain the results in Table V.

The results of the recall and precision computation for this case show it is possible to identify that the five components of interest are related to important structural positions in the application architecture. The method also identifies four other components as false positives, with similar network values compared to the true positives, but not considered as components with high or very high availability by the experts. With this information, architects might become aware that there are four components with occasional downtimes (since they have only moderate availability) that are well connected and may thus affect other components in case of failure.

TABLE V. RECALL AND PRECISION VALUES FOR AVAILABILITY.

Attribute of interest: Availability. Values: high or very high Network measures: Outdegree, eigenvector and betweenness. 10% outliers selected		
	High/very high	**Normal**
Structural important	**5**	**4**
Not Structural important	0	42
Recall =	1 or 100%	
Precision=	0.55 or 55%	

This is true, for instance, for an application that manages accounting-related data supplied on a daily basis to several connected applications; in case of a longer downtime, these applications may thus no longer operate on current data. This information can be used to elaborate adequate mitigation plans.

Dataset02 contains 228 applications, each with two attributes of interest: "business criticality" (a proxy for how critical the application's uptime is for the business process it supports) and "core application" (a subjective value assigned by the experts to those applications considered to be top-ranked in terms of priority). First, we perform the cognitive-structural diagnosis analysis with the "business criticality" attribute. After removing some blank instances (26/228), we create a dummy variable isCritical ("1" for critical or very critical values; "0" otherwise) and test the Spearman correlation with the network measures in Table III. Criticality had moderate Spearman correlation with outdegree (rho=0,379, $p<0.05$) and betweenness (rho=0,313, $p<0.05$) measures, which were selected.

Following the algorithm, the next step is to look for the critical and very critical applications (y=150) and the 150 highest outliers (x=150) for each selected measures. Both measures overlap in 118 of a total of 165 highest outliers analyzed. Finally, we calculate the Recall and precision for the outliers. Among the highest 150 outliers of each selected measure, we can find most (80.66%) of the "critical" and "very critical" applications.

We also have 32 false positives. It is worth mentioning the importance of looking through the 32 false positives to verify whether their business criticality can now be reevaluated by the experts, considering their structural importance. For example, one of the false positives is an application that maintains operationally significant data such as bills, receipts, and other records. Malfunctioning of this application and the corresponding impact on the connected applications may result in major operational issues in, for example, logistics processes.

As Section III.B.2 discusses, we can also perform the cognitive-structural diagnosis analysis with a percentage of the outliers´ set. Rather than using x=150 as before, we now investigate the outputs of the two network measures selecting the 20-percent highest outliers to identify among them critical and very critical components. Table VI shows the results. The sample of 20 percent of the outliers from each measure resulted in 60 different components, with an overlap of 29 components that belong to both outliers´ set.

TABLE VI. RECALL AND PRECISION VALUES FOR BUSINESS CRITICALITY, USING OUTDEGREE AND BETWEENNESS.

Attribute of interest: business criticality. Values: critical or very critical. Network measures: outdegree and betweenness. 20% outliers		
	Correct	**Not correct**
Selected	**56**	**4**
Not selected	94	48
Recall =	56/(56+94) = 37.33%	
Precision=	=56/(56+4)= 93.33%	

In this case, we have a good precision rate when we consider 20 percent of the highest outliers as a sample. Note that we allowed the experts to decide the percentage of selected outliers (10%, 15%, or 20%), as there is no predefined percentage. In this case, 20 percent of the outliers resulted in 60 different components, which means we are considering only part of the 150 "high" and "very high" critical applications (components of interest set). For this piece of the application architecture, the selected network measures produce a precision rate of 93.33% (i.e., business criticality of components is present among the top 20% outliers of betweenness and outdegree), while the low recall value is due to the reduced sample size that was selected to be analyzed (60 of 202 component).

Our last cognitive-structural diagnosis analysis is performed within the Dataset02, attribute core application. This attribute can assume the values "yes" (63) or "no" (165). Four network measures are correlated: indegree (rho=0.355, $p<0.05$), outdegree (rho=0.366, $p<0.05$), eigenvector (rho=0.372, $p<0.05$), and betweenness (rho=0.373, $p<0.05$). We chose to analyze the 63 "core" applications (y=63) and select the 63 (x=63) highest outliers of each measure. We obtain a recall rate of 69.84 percent for the core components and a 43.13 percent precision rate. As a consequence, we identify most of the core components among the outliers set. However, many (58) false positives are returned.

The experts should look at the selected components individually, especially the false positives, and also investigate the criteria used to classify the components as core ones. For example, among the false positives is a quality management application that maintains specifications against which to evaluate product quality. Quality management will not work properly if these specifications are not provided to other related applications; while this may not interrupt business operations, it may put reputation at risk, since products of insufficient quality may get into distribution. Table VII shows the results of the Recall and precision calculation for Dataset02.

Our final analysis is the attribute check for the attribute type of development from Dataset01. This attribute can assume three values: "self-developed," "standard," and "others." For this kind of analysis, we do not need a correlation among the attribute and network measures, since we are performing an exploratory analysis among the outliers that looks for some specific values of interest. We choose the eigenvector and betweenness centralities for the reasons explained in Table II. Considering the 10 percent eigenvector outliers that had type of development identified as a value (18 applications), we find

TABLE VII. RECALL AND PRECISION VALUES FOR "CORE APPLICATION" USING INDEGREE, OUTDEGREE, EIGENVECTOR AND BETWEENNESS

Attribute of interest: core application. Values: yes and no. Network measures: indegree, outdegree, eigenvector and betweenness. 63 outliers		
	Correct	**Not correct**
Selected	**44**	**58**
Not selected	19	107
Recall =	=44/(44+19)= 69.84%	
Precision=	=44/(58+44)=43.13%	

that among the 18 most important components in terms of eigenvector centrality, almost all are "self-developed" software components and only one is "standard." Regarding the outdegree values, we have only 3 standard software components among the 19 highest ones.

Altogether, the fact that the most structurally relevant components (pointed out by eigenvector centrality) and the greatest providers of interfaces (applications with high outdegree) belong to the "self-developed" category might indicate less external dependency on software vendors. On the other hand, the more self-developed components are among the most well-connected ones (as it is present in this case), the more likely may be a variety of self-developed interfaces, which in turn may reduce the architectural agility and impede any integration with partners outside the organization.

The results of the cognitive-structural diagnosis analysis demonstrate the method's potential as an auxiliary tool for experts that can help them refine their tacit EA knowledge. In addition, it was possible to investigate how the values of a certain attribute (type of development) are distributed among the most important structural points of the network with the attribute check analysis.

V. CONCLUSIONS

In line with the objectives we formulated earlier, the two methods presented in this paper to combine network measures and expert knowledge as support mechanisms for EA analysis, applied in two empirical datasets with more than 200 applications in each of those. We also proposed a classification schema for network analysis initiatives at the component level.

This research makes two main contributions. First, we were able to perform the cognitive-structural diagnosis analysis and thus demonstrate how it can allow experts to refine their knowledge about the application architecture. The cognitive-structural diagnosis analysis and attribute check methods accept as inputs virtually any network measure and EA concern. For instance, we found that, for the analyzed context, business criticality of components is correlated with the network measures we used, which allowed us to validate the experts' initial, inherently subjective perceptions for most components (88.66%), on one hand, and on the other hand suggest a set of potential candidates for further analysis. We found similar results for the cognitive-structural diagnosis analysis of availability, in which all five components of interest were identified by the method and, given their structural importance, four additional components were suggested for further analysis. Regarding the cognitive-structural diagnosis analysis for core application, 63 of 228 applications, the method identified 69.84 percent correctly. However, since the network measures are individually correlated with the attribute of interest, each of them also bring applications with similar structural values (false positives) to further analysis. The precision rate (43.13%) calls for an individual reconsideration of the selected false positives, which were suggested by, at least, one of the four network measures. We need to highlight the importance of false positives, since they represent components that have network values similar to the true positives but were not classified as components of interest by the experts. We also tested the attribute check approach with type of development values from Dataset01. With this kind of descriptive analysis, it was possible to identify structural positions in the main architecture occupied by self-developed applications. Simple checks like this can warn experts about undesired configurations and help them monitor the EA evolution. It should be noted that both the cognitive-structural diagnosis analysis and the attribute check analysis method accept as inputs virtually any network measure and EA concern.

Our second contribution is a classification schema that will allow the community to organize and position research using its nomenclature. The list of proposed categories is not exhaustive and was based on [3], with two new ones added in this paper.

There are limitations of this study. As for the methods´ validity, the definition of the attribute of interest is the most important threat to the construct validity. To facilitate the comparison of results from different organizations, in future studies we may create a minimum criteria set to define an application as "TOP", for example. A second threat may be the correctness of the modeled network, which depends on the quality of EA documentation and on the process used to build the network. We recommend researchers always double-check their modeled networks. All in all, we do not propose to generalize the results for all organizations, since the definitions of some concepts of interest (e.g., business criticality and core application) we employed in this study may suffer from an interpretive bias and may vary across organizations. Thus, further validation by analyzing application architectures from additional organizations (with different sizes and structures) is desirable for future research.

Despite the originality of the approach, our analysis is concentrated at the application domain - a strategy followed in other works (e.g., [8], [13], [14], and [24]). As other layers, such as business architecture, are also subject to inherent relations and dependencies (cf. [28], displaying a business capability dependency map), we see the use of our approach not as generally limited to the application architecture but also valid elsewhere in the EA to compare subjective perception with structural information in a given layer (e.g., business capability degree with strategic value assigned). Interlayer dependencies, though, have not yet become a subject of our analysis.

In addition, because the attribute check analysis method is an intuitive approach, further examples are needed to provide more evidence of its utility. As for the cognitive-structural diagnosis analysis, it would benefit from future research that includes a case-by-case analysis of the false positives outliers pointed out by each network measure. This analysis consumes considerable time and effort of representatives from the end-user organization. Although we did not have the chance to perform it for all false positives detected, we do recognize its importance. In addition, in our cognitive-structural diagnosis analysis algorithm, every component identified as an outlier by a network measure is selected to be part of the outliers set (it needs to be identified by at least one network measure to be considered as an outlier). To refine the outlier selection (and

possibly improve the precision rate), researchers could test more "robust" strategies with which components have their structural importance confirmed by at least two measures, for example.

Finally, we conclude by stressing again that our cognitive-structural diagnosis approach should be seen as a heuristic to support the expert in identifying key applications and not as a classifier system or regression model to predict the significance of future applications. It is also important to highlight that recall and precision values for the method are not the focus of our analysis, since they are only indicators of overlapping between the two knowledge sources; rather, the aim is to support with the heuristic's output the expert's qualitative analysis. In this sense, a supportive approach should be adopted regarding the benefits of the two proposed methods. They should be seen as a guide or supporting instrument in a specific context more than as substitutes for expert assessments. Nevertheless, we believe that the more complex the application landscape, the more the pure "EA analysis by hand" approach can benefit from our methods' support, as they can both substantiate and complement the former.

ACKNOWLEDGMENT

We thank the anonymous reviewers for their helpful comments.

REFERENCES

[1] R. Winter and R. Fischer, "Essential layers, artifacts, and dependencies of enterprise architecture". Proc 10th IEEE International Enterprise Distributed Object Computing Conference Workshops, EDOCW2006. doi:10.1109/EDOCW.2006.33

[2] F. Ahlemann. Strategic Enterprise Architecture Management: Challenges, Best Practices, and Future Developments (F. Ahlemann, E. Stettiner, M. Messerschmidt, and C. Legner, eds.), Springer Heidelberg Dordrecht London New York, doi: 10.1007/978-3-642-24223-6.

[3] A. Santana, K. Fischbach, and H. Moura, "Enterprise Architecture Analysis and Network Thinking : a Literature Review." In Proc. HICSS 2016.

[4] M. Buschle, J. Ullberg, U. Franke, R. Lagerström, R. and T. Sommestad, A tool for enterprise architecture analysis using the PRM formalism. Lecture Notes in Business Information Processing, 72 LNBIP, 108–121. doi:10.1007/978-3-642-17722-4_8

[5] R. K. M. Veneberg, M. Iacob, M. J. van Sinderen, and L. Bodenstaff, "Enterprise Architecture Intelligence: Combining Enterprise Architecture and Operational Data". 2014 IEEE 18th International Enterprise Distributed Object Computing Conference, pp. 22–31, IEEE. doi:10.1109/EDOC.2014.14

[6] A. W. Schneider, M. Zec and F. Matthes, "Adopting Notions of Complexity for Enterprise Architecture Management". Proc. Twentieth Americas Conference on Information Systems, 2014.

[7] G. Antunes, M. Bakhshandeh, R. Mayer, J. Borbinha, and A. Caetano,"Using ontologies for enterprise architecture analysis," in Enterprise Distributed Object Computing Conference Workshops (EDOCW), 2013 17th IEEE International, pp. 361–368.

[8] D. Dreyfus and G.M. Wyner, "Digital Cement: Software Portfolio Architecture, Complexity, and Flexibility". Proc. 17th Americas Conference on Information Systems, Detroit, Michigan, USA, 2011

[9] J Scott. Social Network Analysis. Newbury Park, CA: Sage, 1992.

[10] Q. D. V. E. Hommes, "Comparison and application of metrics that define the components modularity in complex products", Proc. of the ASME 2008 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, 2008.

[11] O. Mazhelis, J. A. Lehto, J. Markkula and M. Pulkkinen, "Defining Complexity Factors for the Architecture Evaluation Framework", Proc. 39th Hawaii International Conference on System Sciences, 2006.

[12] A. Schütz, T. Widjaja and J. Kaiser , "Complexity in enterprise architectures: conceptualization and introduction of a measure from a system theoretic perspective," Proc. 21st European Conference on Information Systems, 2013.

[13] R. Lagerström, C.Y. Baldwin, A. MacCormack, and S. Aier.,"Visualizing and Measuring Enterprise Application Architecture: An Exploratory Telecom Case". Harvard Business School Working Paper, no. 13-103, 2013.

[14] D. Simon and K. Fischbach, "IT Landscape Management Using Network Analysis." In G. Poels (ed.), LNBIP (Vol. 139), Springer Berlin Heidelberg, pp. 18–34 (doi: 10.1007/978-3-642-36611-6_2).

[15] S. Buckl, F. Matthes, and C. M. Schweda, "Classifying Enterprise Architecture Analysis Approaches", Proc. 2nd IFIP WG5.8 Workshop on Enterprise Interoperability (IWEI'2009), pp. 66–79.

[16] The Open Group (2011). TOGAF Version 9.1. Van Haren Publishing.

[17] M. Mitchell. "Complex systems: Network thinking". Artificial Intelligence, 170(18), pp. 1194–1212. doi:10.1016/j.artint.2006.10.002

[18] G. Morais and R.C. Prati, "Complex Network Measures for Data Set Characterization". Proc. 2013 Brazilian Conference on Intelligent Systems (pp. 12–18). IEEE. doi:10.1109/BRACIS.2013.11

[19] S. Wasserman and K. Faust. Social Network Analysis. Cambridge: Cambridge University Press.

[20] S. Aier and M. Schönherr, "Model Driven Service Domain Analysis". In D. Geor-gakopoulos, N. Ritter, B. Benatallah, C. Zirpins, G. Feuerlicht, M. Schoenherr, and H. Motahari-Nezhad (Eds.), Service-Oriented Computing ICSOC 2006 SE - 17 (Vol. 4652, pp. 190–200). Springer Berlin Heidelberg. doi:10.1007/978-3-540-75492-3_17

[21] D. Naranjo, M. Sánchez, and J. Villalobos, "PRIMROSe - A Tool for Enterprise Architecture Analysis and Diagnosis". Proc. 16th International Conference on Enterprise Information Systems, pp. 201–213.

[22] J. E. Bartolomei, D. E. Hastings, R. de Neufville, and D. H. Rhodes, "Engineering Systems Multiple-Domain Matrix: An organizing framework for modeling large-scale complex systems". Systems Engineering, 15(1), 41–61. doi:10.1002/sys.20193

[23] O. Levina and R. Hillmann, "Network-Based Business Process Analysis". Proc. 45th Hawaii International Conference on System Sciences, pp. 4356–4365. doi:10.1109/HICSS.2012.447

[24] M. Postina, I. Sechyn, and U. Steffens, "Gap analysis of application landscapes". Proc. 2009 13th Enterprise Distributed Object Computing Conference Workshops (pp. 274–281). IEEE. doi:10.1109/EDOCW.2009.5331980

[25] M. Lehnert, J. Seyfried, M. Siegert, and M. Röglinger, "ProcessPageRank – A Network-based Approach to process to prioritization pecisions", Proc. European Conference of Information Systems, ECIS, 2015. doi:10.18151/7217408

[26] L. C. Freeman (1978). Centrality in social networks conceptual clarification. Social Networks, 1(3), 215–239.

[27] A. Hevner, S. March, J. Park and S Ram. Design Science in Information Systems Research. MIS Quarterly, v. 28, p. 75–105, 2004

[28] A. P. Apthorp, "Business Architecture for Change Program Design and Planning". In D. Simon, C. Schmidt (Eds.), Business Architecture Management – Architecting the Business for Consistency and Alignment. Springer, 2015, pp. 179–201.