5-2012

Interface Management: An approach to support IT-Managers in decision making

Lama Balloul
Carl von Ossietzky University Oldenburg, Lama.balloul@uni-oldenburg.de

Jorge Marx Gómez
Carl von Ossietzky University Oldenburg, jorge.marx.gomez@uni-oldenburg.de

Interface Management: An approach to support IT-Managers in decision making

Lama Balloul
Carl von Ossietzky University Oldenburg,
Lama.balloul@uni-oldenburg.de

Jorge Marx Gómez
Carl von Ossietzky University Oldenburg,
Jorge.marx.gomez@uni-oldenburg.de

Abstract

Information systems are critical assets for modern enterprises. They must be updated continuously to reflect business needs, leaving cumulative effects on the system complexity. Information systems are normally composed of multiple subsystems or components. Interfaces arise wherever one subsystem or component interacts with another. This interaction can be achieved through interoperation or integration. However, both require a comprehensive knowledge about existing interfaces, which suffer from lack of documentations. This, as a result, has bad influence on managing the interfaces and could lead to wrong decisions. In our research-in-progress, we point to the problems companies face with their interfaces and present our solution to support IT managers in taking decisions regarding the interfaces in each phase in their lifecycle. Managing the interfaces includes analysing dependencies and the attributes of interfaces with their weight and influence on assessing the related risk.

Keywords

Interfaces, Interface management, Interoperability.

1. Introduction

Information systems constitute the backbone of modern enterprises. They cooperate with each other to achieve the strategic goals of the organization. The IT related success of an organization is not only dependent on the success of an individual system, but rather on the combined work of individuals as well as the success of communication to share information and process it. Therefore, information systems are considered as critical assets for modern enterprises (Comella-Dorda et al. 2000).

An approach like Enterprise Architecture Management (EAM) promises transparency for an enterprise architecture, control of complexity and tracking changes in the architecture (Keuntje et al. 2010) which is done in both strategic and operative dimensions (Niemann, 2006). However, the prerequisite for successful EAM is a consistent knowledge about the current IT landscape, and available resources of a company. The degree of complexity, the budget and the skills in a company, especially in small and medium sized companies, are important factors for the success of EAM (Addicks et al. 2010). We see EAM as an umbrella, under which our work could be classified. However, it is covered foremost by the operative part and focuses on the interfaces between applications: how to construct and design interfaces and more important how to deal existing interfaces and achieve better management for them.

2. Problem Definition

In (Niemann, 2006), the author states the question “Do you currently have access to up-to-date information on the interfaces and dependencies in your environment?” This question can be posed to several companies without having a satisfied answer. Another big question is “Do you know how often and with what degree of reliability these interfaces are used?” The responsibility of interfaces is distributed and with the lack of documentation and the lack of a repository of running and deactivated interfaces, these questions often remain unanswered.

According to interviews we made with a company in Germany, they lack an overview of the interfaces within their IT-landscape. The existing documentation of interfaces is quite various. Sometimes, the description of an interface is very detailed and unstructured which aggravates understanding of the interface, while others are barely described. Even worse, often there was no documentation at all.

The lack of this overview makes it difficult for the IT managers to make right decisions about when to update an interface, and when to rewrite it. However, for them it naturally is important to find out if it is possible to rewrite an interface; if there are any consequences on making changes on an interface and where to apply them.

Problems which are accompanied by e.g. outdated interfaces, lead to failure in executing business processes. This in turn could have potentially drastic consequences on the company according to the importance of affected interfaces.

During the lifetime of interfaces, their costs are relatively high in comparison with the overall IT-budget. According to Gartner group (Ambrose and Morello, 2004), the maintenance and implementation of interfaces cost around 40% of the total IT budget. For this reason, the interfaces which have the highest maintenance costs are potential candidates for reengineering, of course with consideration for other factors.

3. Course of Action

Our research area of Interface Life Cycle Management has its idea from the discipline of Product- and System Lifecycle Management. The life cycle of an interface is not different from the case of a system or a product. It starts with the requirements, to decide the design moving to the configuration. It continues with the run time where it spends most of its time. In parallel to the run time phase, the interface is under monitoring for small changes in configurations or further changes in the requirements. When an interface becomes irrelevant or should be replaced, it moves to the deactivation phase.

ILCM is a systematic procedure which considers the interfaces during their lifetime. ILCM identifies and analyzes the interfaces and their characteristics, and estimates the related risk to aid valuating the interfaces and shows the discretion to act. The course of action consists of five phases which will be described next (see Figure 1):

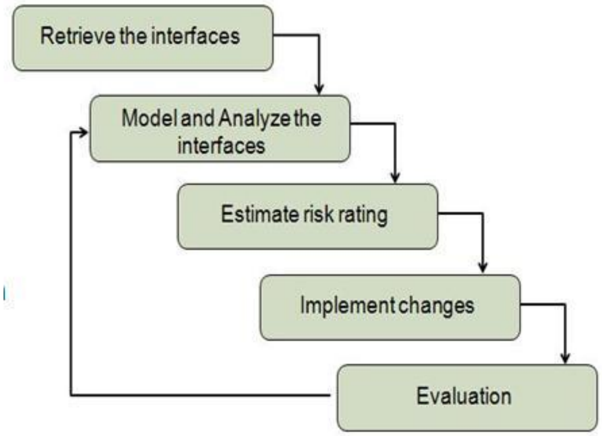

Figure 1: Phases of Interface Life Cycle Management

3.1 Retrieve the Interfaces

According to Giachetti (2010), interfaces can be identified by either examining process input-and-output to determine the source and the goal of the information, or by examining the data flow diagrams and other models which identify interfaces. Unfortunately, the existence of such diagrams cannot be guaranteed in all companies and especially when dealing with older 'Legacy Systems', so we had to think about other means to identify interfaces and retrieve them. There is an interaction-based model presented by Stroulia et al. (1999) which yields an interface map where the transactions between interfaces are shown. After collecting these data, they are filtered to get the basic required data, which we will propose to classify in three layers:

- Business layer, in which the functionality of an interface is determined.
- Logical layer, in which the general characteristics are required: Interface's source, interface's destination, type of exchanged data.
- Physical layer, in which specific data regarding the used technology and the configuration of an interface are collected.

3.2 Model and Analyze the Interfaces

To store the collected data, a matrix, database, or graph database can be used. We chose the graph database for several reasons: Calling up the data from a two dimensional matrix is insufficient with very low performance. Storing data in a database requires extra rows to store the relations between interfaces which can be done easily with graph database. Graph databases map more directly to the structure of object-oriented applications and can scale to large datasets (Angles and Gutierrez, 2008). They are not restricted to a rigid schema like a normal database and graph theory algorithms can be applied to them (Degener et al. 2010). In our work we use the 'GRaph Indexing based on Pre- and Postorder numbering (GRIPP)' for its high efficiency and scalability. It can answer reachability queries on graphs with 5,000,000 nodes in less than 5 milliseconds (Trißl and Leser, 2007).

The analysis starts by defining interface's characteristics based on our three layers.

- Business Layer

  Costs: High maintenance costs can be an argument to think about migration.

  Business value: The operational area decides what to do in the case of failure happening. It is noticed that interfaces which are related in the financial field, have a higher priority.

Business risk: When an interface is not able to complete its mission, the consequences and aftereffects which it has on the daily business should be listed here.

- Logical Layer

  Weightiness: The number of connected interfaces. As this number increases, the risk rate will consequently increases, too, because if this interface contains a failure or produces a failure, all depending interfaces can be affected and generate further failures. The connectivity metric can be calculated using the following equation (Connectivity Metric = number of connected interfaces / total number of interfaces in the enterprise)

  Dependency: if the former interface necessitates a change, this can cause a necessity for a change in the following interface. The same can happen in the instance of failure occurring.

  Type of transmitted data: Synchronous connections often demands faster processing than asynchronous.

- Technical Layer

  Protocol: if a failure appears in several interfaces which use the same protocol, this could be a sign that the mistake is in the underlying protocol. Therefore, all interfaces which use the same protocol should be identified to estimate their priority and business risk.

  Frequency of call: the more often an interface is called, the higher probability of the failure appearance.

  Data volume: The bigger this volume, the higher the interface's priority.

3.3 Estimate Risk Rating

The risk could be associated with keeping an interface, adopting migration, failure appearance, or establishing reengineering process. The percentage, which each attribute contributes in calculating the risk in each case, is our actual work. The reasons for causing a risk are concluded in the following points (Schorn, 2008):

- Incomplete or badly defined requirements
- Lack of internal know-how
- Proprietary interfaces (in-house formats)
- Over-engineering, too many and too complex interfaces
- Standard interfaces which have been customized to such a degree that they no longer conform to the standard
- Inadequate documentation
- Insufficient testing

Examples of risks which companies can face:

- Business processes could not be completed which has its consequences on the repute of the company and/or penalty
- Security risk when an interface is still accessible when it is not supposed to be
- Increasing costs for the project which could exceed the project budget

4. Conclusion

The last two phases of this work are our current activity. Changes will be implemented in the light of a prototype according to the weight of interfaces' metrics and this prototype will be evaluated from IT managers. As mentioned before, this is a research-in-progress and future results will be published later in full paper.

References

Addicks, J.S., Appelrath, H.J., Gringel, P., Postina, M., Steffens, U. and Zillmann, C. (2010) “A Component-Oriented Tool-Approach to Enterprise Architecture Management” in Schuhmann, M., Kolbe, L.M., Breitner, M.H. and Frerichs, A (eds.) *Multikonferenz Wirtschaftsinformatik*, Göttingen: Universitätsverlag, pp. 31-32.

Ambrose, C. and Morello, D. (2004) Designing the Agile Organization: Design Principles and Practices. Working Paper, Gartner Group.

Angles, R. and Gutierrez, C. (2008) Survey of graph database models. ACM Computing Surveys (CSUR), v.40. n.1, pp.1-39.

Comella-Dorda, S., Wallnau, K., Seacord, R. and Robert, J. (2000) A Survey of Black-Box Modernization Approaches for Information Systems. In Proceedings of the International Conference on Software Maintenance (ICSM'00), Washington, DC: IEEE, pp. 173-.

Degener, J., Scott, M. M., Giannandrea, J. and Michener, B. (2010) Optimizing schema-last tuple-store queries in graphd. In Proceedings of the 2010 international conference on Management of data (SIGMOD '10). New York, NY, USA: ACM, pp. 1047-1056.

Giachetti, R. E. (2010) *Design of Enterprise Systems: Theory, Architecture, and methods*. Boca Raton, USA: CRC Press.

Keuntje, J. H.; Barkow, R. (2010) *Enterprise Architecture Management in der Praxis: Wandel, Komplexität und IT-Kosten im Unternehmen beherrschen*. Düsseldorf: Symposion Publishing.

Niemann, D. K. (2006) *From Enterprise Architecture to IT Governance: Elements of Effective IT Management, 1st edition*, Berlin: Vieweg & Sohn Verlag.

Schorn, P. (2008) “Interfaces @ Supply Chain Event Management” in Ijioui, R., Emmerich, H., Ceyp, M (eds.) *Strategies and Tactics in Supply Chain Event Management*, *Berlin*: Springer-Verlag, pp. 51-64.

Stroulia, E., El-Ramly, M., Kong, L., Sorenson, P. and Matichuk. B. (1999). Reverse Engineering Legacy Interfaces: An Interaction-Driven Approach. In Proceedings of the Sixth Working Conference on Reverse Engineering (WCRE '99). Washington, DC, USA: IEEE Computer Society, pp. 292-.

Trißl, S. and Leser, U. (2007) Fast and Practical Indexing and Querying of Very Large Graphs. In Proceedings of the 2007 ACM SIGMOD international conference on Management of data, Beijing, China: ACM, pp. 845-856.