FOUNDING ENTERPRISE SYSTEMS ON ENTERPRISE PERFORMANCE ANALYSIS

Ian Douglas
Learning Systems Institute and Department of Computer Science, Florida State University, 320A, 2000 Levy Avenue Innovation Park, Tallahassee, Florida, 32310, USA.

Keywords: Domain analysis, Service-oriented computing, Reuse

Abstract: Information, knowledge and learning systems are developed with the implicit belief that their existence will lead to better performance for those using them, and that this will translate into better performance for the organisations for which the user works. One important activity that must occur prior to requirements analysis for such systems is organisational and human performance analysis. One key software application that is missing from most organisations is an integrated enterprise system for analysing performance needs, determining appropriate support solutions, monitoring the effect of those solutions, and facilitating the reuse and sharing of the resulting knowledge. A model for such a system is presented, together with a prototype demonstrating how such a system could be implemented.

1 INTRODUCTION

The developers of information systems normally go about their work with the belief that the deployment of such systems will result in an improvement in the operation of an organisation and the results it achieves. There are many instances when this does not prove to be the case. In his analysis of software failure, Flowers (1996) locates the main cause not in software development, but rather in its conception. Managers often suppose that a computer system is a cure-all for operational problems in business practice. In many instances, there is a rush to develop a solution before there is an understanding the problem.

While information systems are one solution type that can be used to improve the performance of humans and organisations, other systems also promise such improvement, e.g. business process re-engineering, training, job aids, e-learning, knowledge management. Many of these solutions are also implemented without careful analysis of their contribution to supporting organisational need.

The entire organisational system, not just the computer components, must be considered prior to solution selection. It is crucial for an enterprise to understand how the performance of its individual employees and teams contributes to its goals and results prior to the development of any solution system. The "solutions-oriented" thinking in many organisations needs to be replaced by more holistic "problem-oriented" thinking. Traditionally, when "analysis" is done, it is often framed with a particular solution in mind. Performance analysis and human performance technology have emerged as means of focussing on the overall performance of organisational systems (Gilbert, 1996, Robinson and Robinson, 1995, Rosset, 1999). Underlying this approach is general systems thinking (Wienberg, 2001).

There is a great deal of technology available to assist solutions design and construction, e.g. software engineering case tools, but there is relatively little available to assist performance analysis. Such technology facilitates understanding of the organisational performance factors that the solution is intended to address, prior to the development of any solution, and also facilitates the collection of baseline metrics required to determine additional value created by any solution that is implemented.

Douglas and Schaffer (2002), present a methodological framework for technology supported organisational performance improvement. The framework requires the reporting of the analysis in terms of reusable knowledge components, which are stored in repositories. The framework also incorporates the need for visual modelling of performance, collaborative analysis, rationale management and configurable support systems.

Douglas I. (2004).
FOUNDING ENTERPRISE SYSTEMS ON ENTERPRISE PERFORMANCE ANALYSIS.
In *Proceedings of the Sixth International Conference on Enterprise Information Systems*, pages 588-591
DOI: 10.5220/0002657105880591

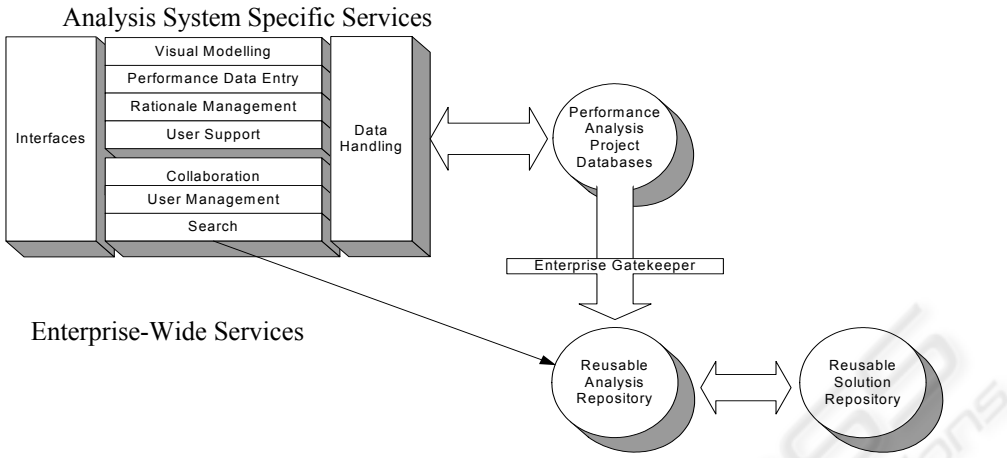

Figure 1: Architecture for the performance analysis support software

This paper describes a model for the technology support for the framework and a prototype system which demonstrates how performance analysis can be facilitated by a web-based enterprise software system. The system is focussed first on the goals of the organisation, what processes need to occur to meet these goals, and what roles are defined to carry out the activities involved in a process. The analysis system is used to determine the need for support and help in the selection of the appropriate solutions (e.g. computer programs, training courses, job aids).

A repository of this analysis knowledge would be created which is organised by discrete performance goals identified in analysis. The analysis repository can be linked to a repository of reusable solution components. Thus, if a new project discovers in the analysis repository some prior analysis of relevance to the current problem, they will also discover any solutions that resulted from the analysis.

2 A PERFORMANCE ANALYSIS SUPPORT SYSTEM

A working prototype of an organisational performance analysis system has been completed and is under evaluation with both the US army. The prototype is entirely web-based and supports all the elements of the framework noted in the introduction. An important concept embedded in the design of the prototype is configurability (Cameron, 2002), i.e. tools should not be fixed to a particular methodology, but be adaptable to the specific needs, methodologies and terminology used in different organizations and groups. The intention is to create a set of configurable tools and methods, which have a shared underlying representation of performance analysis knowledge. The system architecture is based on the emerging new paradigm of service-oriented software (Yao, Lin and Mathieu, 2003). This allows custom interfaces to a continuously refined shared repository of knowledge on human performance. Each version of the performance analysis system will have core components (see figure 1), but the specific version of the components will vary from organization to organization. In the current system a third party collaboration tool called Collabra has been tied into the systems to handle the collaboration component. If a different organization used a different collaboration tool this would be ‘ plugged in’ in place of Collabra. Likewise if different data types were collected in another organizations methodology (or different terminology used), different data entry templates could appear. The user support component can be tailored to the specific methodology employed by an organization.

The components of an analysis (models, data and rationale) are stored in a project specific database from which analysts and stakeholders can retrieve, view and comment on the contents. These can then be transferred to a repository of performance analysis knowledge. Some organizations may wish to have a gatekeeper function to check the quality of the components entered into the central repository. An integral part of the tool is an automated search of this repository. Thus, as soon as an analysis team on a new project begins to enter data, it is matched against existing data in the analysis repository to alert the user to possible sources of existing knowledge. Ye and Fischer, (2002) argue for the need for this type of automated task-aware, context-sensitive search to encourage reuse. It is likely that

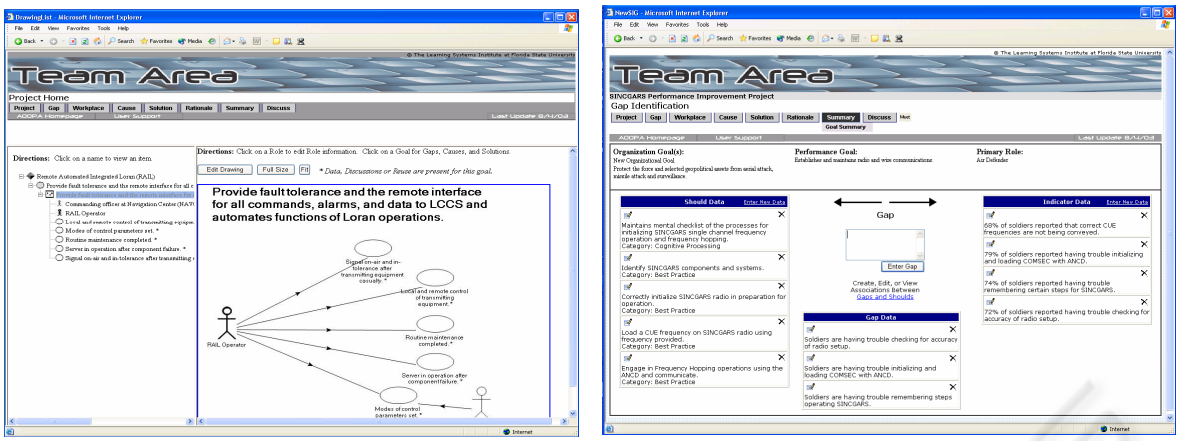

Figure 2: Screenshots of one instance of an analysis tool conforming to the framework

the user will be discouraged from using a resource that requires a lot of browsing in order to determine if relevant information exists.

Figure 2, illustrates the prototype that has been constructed to demonstrate one instance conforming to the framework and the architecture illustrated in figure 1. The modelling component is a key focal point and provides a shared reference and navigation aide throughout a project. The current prototype uses performance case modelling, which is an adaptation from unified modelling language (UML) Use Case notation is widely used in object-oriented software systems analysis (Cockburn, 1997) and has been adapted for more general systems analysis (Marshall, 2000). Performance case notation provides a simple, end-user understandable means of defining a problem space.

A performance diagram is a graphic that illustrates what performers do on the job and how they interact with other performers to reach performance goals. A role is a function that someone as part of an organizational process (e.g., mission commander, radio operator, vehicle inspector).

A primary role is the focus of a project. Secondary roles are entities that interact with the primary role, and may be included when looking at team performance. The primary role is likely to achieve several performance goals, e.g. a mission commander would have to successfully plan, brief, execute and conduct an after action review. High level performance goals decompose into lower level diagrams containing sub-goals. Performance goals represent desired performance at an individual level and each should be directly linked to an organisational level performance goal.

Facilitated by the groupware component the analysis team works collaboratively to create and edit the performance diagram. The analysis team will use the diagram to develop a shared understanding of a domain and identify performance cases where there is a gap between desired on-the-job performance and current on-the-job performance. It allows the organization to pinpoint a specific performance discrepancy that could be costing time, money, and other resources. Those performance cases will be subject to a more detailed analysis.

There are a variety of data collection templates that could be attached to the performance case to assist in this. The current version of the prototype uses a gap analysis template (see right side of figure 2) in which data is collected about current and desired performance in the tasks that are carried out in pursuit of a performance goal. Where a gap is found, for example if 100% accuracy is required on a task and only 60% of those assigned to the task are able to achieve this, then a cause and solution analysis will be initiated. The ultimate goal of problem-solving analysis is to close or eliminate this gap in the most cost-effective manner. In a cause

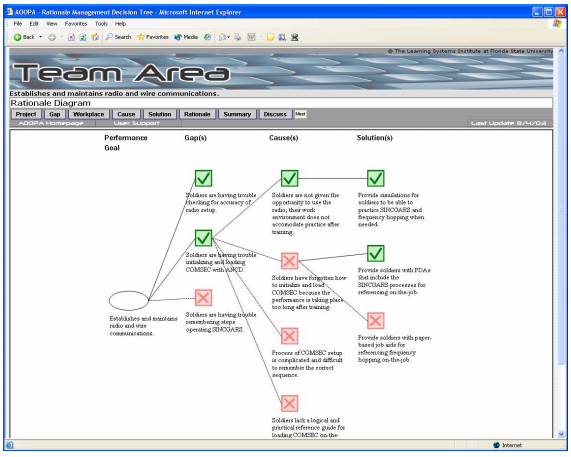

Figure 3: An automatically generated rationale diagram

analysis, stakeholders review gap data, brainstorm

possible causes, put them into cause categories, rate them by user-defined criteria, and select which ones to pursue. The prototype allows users to categorize causes so the recommended solutions are more likely to address the underlying causes. The specific process used in this version is described in more detail in Douglas et al, 2003.

The focus of this system is the actual roles people perform in an organisation (as opposed of their position titles) and the goals they are expected to achieve. This is modelled in an analysis system with the models providing a framework for performance metrics. The gaps in organisational performance evident in these metrics are used to initiate support systems development (software tools, training courses).

Rationale management (Moran and Carroll, 1996, Burge and Brown, 2000) is integrated into the system. Rationale management allows auditing of decision making when solutions resulting from analysis fail to make an impact on organisational performance. In addition to the capture of informal rationale information, through archiving of online discussions, a rationale diagram can be automatically generated from the data entered into a system. Figure 3 illustrates the rationale diagram generated by the current prototype. For each performance goal, gaps in performance leading to the goals are entered, and those selected for further analysis are indicated by a tick. For each gap selected, the potential causes for the gap are indicated. For each selected cause for the gap, the potential solutions considered are indicated, and a tick will show the solutions chosen for implementation.

3 FUTURE WORK

The concept of configurability is an important part of the work carried out to date. The framework on which the model is based is meant to provide a structure for a variety of methods that can be tailored to specific groups or situations. The same is true for software architecture. A fixed tool based on one specific methodology is likely to be of limited use.

This concept is difficult to demonstrate and test when there is only one instance conforming to the model. A second prototype is being constructed, which is conformant to the framework, but is customised to the specific data collection methods, terminology and collaborations tools used by the US Coast Guard's Human Performance Technology Centre. Once more than one instance of an organisational performance analysis tool is available, it will be possible to investigate the possibility of translating performance analysis data between different tools conformant to the framework. Domain ontology will be used to facilitate this.

REFERENCES

Burge, J. and Brown, D. C., 2000. Reasoning with design rationale. *In John Gero (ed.) – Artificial Intelligence in Design '00*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 611-629.

Cameron, J., 2002. Configurable development process. *Communications of the ACM*. (Vol. 45, No. 3), 72-77.

Cockburn. A., 1997. Structuring use cases with goals. *Journal of Object Oriented Programming*, 10 (7), 35–40.

Douglas, I., Nowicki, C., Butler, J. and Schaffer S., 2003. Web-based collaborative analysis, reuse and sharing of human performance knowledge. *Proceedings of the Inter-service/Industry Training, Simulation and Education Conference (I/ITSEC)*. Orlando, Florida, Dec.

Douglas, I. and Schaffer, S., 2002. Object-oriented performance improvement. *Performance Improvement Quarterly*. 15 (3) 81-93.

Flowers, S., 1996. *Software failure: management failure. Amazing stories and cautionary tales.* Chichester, New York: John Wiley.

Gilbert, T., 1996. *Human competence: Engineering worthy performance* (Tribute Edition). Amherst, MA: HRD Press, Inc.

Marshall, C., 2000. *Enterprise Modeling with UML: Designing Successful Software through Business Analysis.* Reading, MA: Addison-Wesley.

Moran, T.P. and Carroll, J.M., 1996. *Design rationale: concepts, techniques, and use*. Mahwah, NJ: Lawrence Erlbaum Associates.

Robinson, D. and Robinson J.C., 1995. *Performance Consulting: Moving Beyond Training*. San Francisco: Berrett-Koehler. References

Rossett, A., 1999. *First Things Fast: A Handbook for Performance Analysis*. San Francisco: Jossey-Bass Pfeiffer.

Weinberg, G., 2001. *An introduction to general systems thinking*. New York: Dorset House.

Yao, C., Lin, K.J., and Mathieu R.G., 2003. Web Services Computing: Advancing Software Interoperability. *IEEE Computer*, October, 36 (10), 35-37.

Ye, Y., and Fischer, G., 2002. Supporting reuse by delivering task-relevant and personalized information. *Proceedings of the Twenty-fourth International Conference on Software Engineering*, 513-523.