Journal of Universal Knowledge Management, vol. 1, no. 2 (2006), 123-129
submitted: 15/5/06, accepted: 15/6/06, appeared: 28/9/06 © J.UKM

The Role of Learning in RE – Position Paper –

Frank Houdek
(DaimlerChrysler AG, Germany
Frank.Houdek@DaimlerChrysler.com)

Abstract: Organizational learning plays an important role in requirements engineering (RE) – probably more than in some other software engineering subjects, as RE needs to be highly domain- or context-specific. But which organizational learning technique is the most suitable one? In this position paper, I identify a number of well-known techniques, reflect them for RE, and recommend a combination of techniques for organizational learning in RE.

Keywords: requirements engineering, organizational learning, domain specific learning
Categories: D.2.1, K.3.2

1 Introduction

We learn cooking in a different way than we learn to discuss a mathematical function. The way we learn best depends on many factors, including the nature of the subject we want to learn. This observation holds also for different subjects that are necessary in a soft-ware organization. Clearly, the differences are not that intense than in the example mentioned above. But in fact, it is relevant whether we want to learn and improve us as a company in programming device drivers, setting up a configuration management system, or performing requirements engineering (RE). In the first case we might acquire detailed know-how from external sources. In the last case, external training (including education at universities) is bound to teach general concepts and principles. It is hardly possible to become a good requirements engineer for a particular domain (e.g. automotive) without learning in that domain.

In this position paper, I elaborate on the concept of subject specific learning techniques and try to identify a suitable set of learning techniques for RE.

2 Learning in Software Engineering

On our way to differentiate suitable organizational learning techniques for particular software engineering topic areas (in our case: RE) from less suitable techniques, we first have to identify factors that might characterize the commonalities and differences of software engineering topic areas with respect to learning.

One facet is *problem-to-solution*: The whole process of software development is about transforming real-world problems into machine solutions. And every real-world problem is represented by humans (even if there is a merely technical problem to solve). The more we are on the problem side of the spectrum, the more differentiation

we see. Aspects of differentiation include processes, notations, (technical) language, quality management techniques, level of abstraction and many more.

Example: Modern warships (which are multi billion dollar projects) are typically described in 2500 to 3500 requirements [Hoppe, 05]. A single automotive ECU[1] (project cost might by about 1/1000 of the warship project costs) is specified by up to several 10000 requirements. Clearly, the nature of the requirements is completely different. But this difference affects the skills that a requirements engineer must have and that the organization wants to teach her.

On the solution-faced end of the spectrum, we see significantly less differentiation. Of course, there are many programming languages, libraries, compilers, etc. available, but compared to differences in the problem space, the variation is low. The consequence for individual and organizational learning is that on the solution-faced end general training might fit perfectly. On the problem-faced end general training usually fails to teach the required domain- or problem-specific skills.

To illustrate this: In my view, this is why nearly all of us have learned programming during our professional education, but nearly none was teached how to be a good requirements engineer.

Another facet is *open-to-standardized*: A number of activities in system engineering are more or less regulated. Amongst the more standardized (by means of legal standards or de-facto standards) ones, we see for example electronic circuit definition or project cost accounting. Some less regulated activities are project planning and software architecture definition.

Clearly, the problem-to-solution facet is not completely independent from the open-to-standardized facet. The more we move towards the solution end (the machine), the higher is typically the degree of standardization. This is due to the fact that solutions are realized on machines that are precisely described. The problem side tends to be more individual (see example above), but there are exceptions.

Example: In telecommunications we see many regulations like protocol standards. In this domain, negotiating product requirements often means identifying parts of that protocol that should be supported in the next product. Due to the high standardization, training on these protocols can be acquired by external training vendors.

Let us come back to RE. With respect to our two facets, RE tends to be on the problem and often on the open sides of the spectrums. As illustrated above, this means that it is unlikely to acquire the necessary knowledge externally and as a consequence a company must have a vital interest in organizational learning in this subject.

3 Techniques for Organizational Learning

Given this need for organizational learning in RE, which learning techniques are most suitable? There are a number of known techniques that might be used for organizational learning in general. In the Section 3.1, I introduce some of them briefly and reflect their specific fitness for organizational learning in RE. The selection of techniques is determined by my own observations in that area. Section 3.2 recommends a combination of learning techniques for RE.

[1] Electronic Control Unit

3.1 Review of learning techniques

Experience repositories / Experience databases are electronic repositories that contain experience knowledge in a somehow structured format. Typical examples are

- Experience Databases (which might be generic or topic specific [Schneider, 01]).
- (electronic) Books of Knowledge [SWEBOOK, 06]
- Pattern Collections (see, e.g.[Hagge, 06], [Houdek, 00] or [Houdek, 97])

RE Reflection: The mere organizational learning in RE by means of databases usually does not work. One positive exception are pattern collections, but not in the sense of a repository itself but they are an excellent means for analyzing real-world observations and distilling recommendations for future work.

(Organized) Experience Exchange within the organizational unit. In such meetings, representatives from different business units (plus external speakers) report about their activities, their lessons learned, and future steps. Main benefits are increased awareness and sometimes joint activities. Typical forms are:

- Regular experience interchange meetings (special interest groups, working groups)
- Situational experience interchange meetings (with attendees from several departments/business units)
- Specific forms of experience exchange are newsletters that are issued somehow regularly.

RE Reflection: These techniques are applied quite often. The main benefit is making people aware of current activities and identifying experience sources.

Often experience exchange happens on a more individual basis (during coffee breaks, etc.) To encourage such kind of exchange, **Experience Brokers** [Johannson, 99] might be used.

RE Reflection: This concept augments organized experience exchange. As an institution of its own it is usually too expensive; but as a by-product of coaching activities this works pretty well.

By transforming **knowledge into work instructions**, (new) knowledge becomes a vital part of the organization. Again, we see several sub-forms:

- Process handbooks are mostly straight-forward. However, there is the problem of following process instructions. Depending on the organization's culture in following process definitions this might work better or worse.
- Standard-Templates (e.g. standard specification template) provide a more concrete means to apply corporate knowledge. New insights are incorporated in such templates that might be mandatory for new projects.
- "Toolification" of procedures (e.g. Quality Gate checklists). Some organizations made good experience by building new procedures into tools. These are often IT-based (e.g. in a project tracking system), but might be manually used (e.g. a review checklist), too.

RE Reflection: Process handbooks or generic work instruction typically do not lead to the intended effect. One reason for this is the missing level of detail. As ar-

gued in Section 2, RE tends to be problem-oriented and individual which contradicts standardized instruction sets.

More work-related instructions like templates or even toolification have significantly better effects; crucial is here (a) a healthy update rate and (b) control of use of these instruments (e.g. the use of standard-templates is checks during document reviews).

Coaching means that a fixed set of – usually internal – consultants provide concrete assistance. Often coaching happens even on a 1-by-1 basis. By the fact that these consultants support different projects, transfer of knowledge and best practice happens automatically. Internal research groups, consulting groups or SEPGs might fulfil this role.

RE Reflection: In my experience, this technique is the most powerful one. This approach works the better, the more similar projects or departments the coach has supported before. Examples from other, similar groups presented by the coach help to understand; reflections and comments by the coach help to internalize knowledge.

Internal **Training** means that colleagues from one or more departments or business units are trained by internal trainers. Clearly, the borderline between coaching and training is smooth. In this context, I would require at least 5 to 10 attendees with inhomogeneous project background.

RE Reflection: In my experience, the effect of training in RE is not so much in the area of dissemination or even internalizing of knowledge, but in creating awareness, demonstrating people that they face similar problems, and that they all have to change something.

Empirical investigations [Basili, 86] require a somehow thorough planning of a case study or experiment, measuring relevant variables (e.g. applied RE technique, invested effort, and cased SW-architecture issues), and analysis of the collected data with respect to the goal of the investigation. This type of learning (which has to be accompanied by suitable dissemination techniques) is quite expensive, but helps to improve an organization's knowledge pool in a systematic way. In the end, this idea leads to the full **experience factory** concept [Basili, 02].

RE Reflection: In general, empirical investigations are a good means to acquire knowledge on effects of RE activities. Especially from an analytic point of view, empirical investigations provide most evidence on causal relationships in RE. But there are two drawbacks: first, empirical investigations are pretty expensive; second, empirical investigations are not well-accepted by practitioners as a means to learn from [Alexander, 05].

As last form, I'd like to mention "**Living Process Handbooks**", which might be implemented as project specific Wiki systems [Leuf, 01]. The basic idea is to have an online process handbook that is augmented by project specific details. This makes the information very attractive (because currently relevant information like the next specification review date, the contact information of the project manager, her current deputy, or the review checklist can be found there). If information is no longer accurate, everyone is allowed/encouraged to modify the information.

RE Reflection: So far, I have no own experience, only promising observations. But I expect much potential here. Wiki systems allow the combination of general process descriptions (like the RE process) and project specific instantiations. If modifications are not accepted automatically, but a central Wiki manager is in the loop, she has also the ability to (a) ensure that no irrelevant information is entered into the system and (b) interesting concepts that might be introduced in one branch of the information universe is made known to others, too. So far, Wiki systems seem to overcome some shortcomings of traditional electronic experience repositories:

- the repository is visited frequently (as relevant and up-to date project information is stored there) [Wieser, 00]
- information is accurate and up-to-date (as everybody is allowed to correct it without any burden)
- information is detailed enough (as information is project specific).

3.2 Recommended combination of RE learning techniques

As a consequence of the RE reflection in Section 3.1, I recommend the following combination of techniques:

- *Organizational experience interchange.* This technique is the basis of organizational learning in RE. Regular events create awareness, keep attention on the topic, and bring people together.
- *Coaching by internal coaches.* This is the most effective learning technique. A generic coaching process might look like this:
    - Understanding the particular needs of the new customer, explaining solutions by means of best practices that are used by other groups.
    - Transfer and – if necessary adapt – solutions.
    - Demonstrate the new practices by means of examples from the new customer.
    - Explain and instruct the new customer so that he starts to apply "his" new process
    - Act as a reviewer for work products the new customer has created by means of his new process.
- *Training.* This activity is used in parallel to coaching and is intended to inform a larger number of people. Thus it is between experience interchange meetings on the one hand and coaching on the other hand.
- *Work instructions transformed from knowledge*. This technique should be mainly used for experience that is used infrequently. Very typical examples are standard specification templates that contain basic requirements, that are necessary for the business, but that are out of the daily focus of an engineer.

Figure 1 illustrates the interplay of these techniques graphically. Whereas experience exchange and work instructions are used all the time, coaching and training are timely focused. Living process handbooks might be used after such a learning period (as shown in the figure). Some aspects of living process handbooks might be beneficial without explicit learning periods as well.

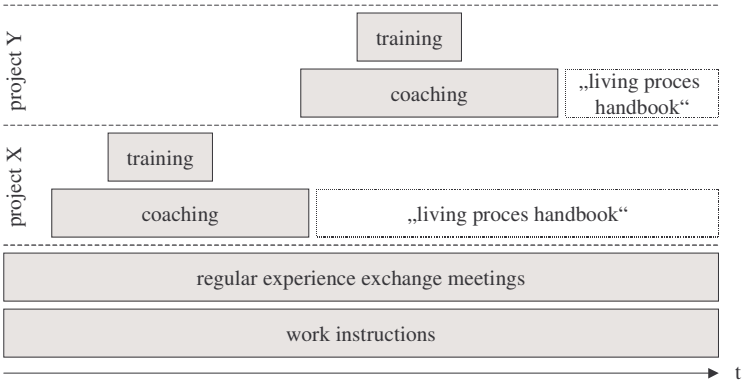

Figure 1: Interplay of RE learning techniques.

4 Conclusions

The more context- or domain-specific a software engineering subject is, the more demand is on organizational learning as there is no sufficient specific knowledge available to benefit from. Requirements engineering is among these subjects.

However, this leads to the question, which learning technique is the most appropriate one. The paper presented a number of commonly used techniques in RE learning and recommended a combination of four (plus one) techniques.[2]

References

[Alexander, 05] I. Alexander, S. Robertson, N. Maiden. What influences the requirements process in industry? A report on industrial practice. In Proc. 13th IEEE International Conference on Requirements Engineering, 2005, pp. 411-415.

[Basili, 02] V.R. Basili, G. Caldiera, H.D. Rombach. Experience Factory. In J.J.Marciniak (ed): Encyclopedia on Software Engineering (2nd edition), John Wiley, 2002, pp. 511-519.

[Basili, 86] V. R. Basili, R. W. Selby, D. H. Hutchens. Experimentation in software engineering. IEEE Transactions in Software Engineering, vol. 12, no. 7, 1986, pp. 733--743.

[Hagge, 06] L. Hagge, F. Houdek, K. Lappe, B. Paech. Using Patterns for Sharing Requirements Engineering Process Rationales. In B. Paech et al. (eds). Rationale Management in Software Engineering, Springer, 2006 (to appear).

[Hoppe, 05] M. Hoppe, personal correspondance, 2005.

[Houdek, 00] F. Houdek, C. Bunse. Transferring and Evolving Experiences: A Practical Approach and Its Application on Software Inspections. In G. Ruhe, F.Bomarius (eds): Learning Software Organizations. Methodology and Applications, Springer, 2000, pp. 210-226.

[2] The four techniques training, coaching, regular experience meetings, and work instructions. The ‘plus-one’ technique is a living process handbook.

[Houdek, 97] F. Houdek, H. Kempter. Quality patterns – An approach to packaging software engineering experience. ACM Software Engineering Notes, vol. 22, May 1997, pp. 81-88.

[Johannson, 99] C. Johannson, P. Hall, M. Coquard. Talk to Paula and Peter – They are Experienced. In Proc. 1st International Workshop on Learning Software Organizations, 1999, pp. 69-76.

[Leuf, 01] B. Leuf, W. Cunningham. The Wiki Way: Quick Collaboration on the Web, Addison-Wesley 2001.

[Schneider, 01] K. Schneider, T. Schwinn. Maturing Experience Base Concepts at Daimler-Chrysler. Software Process Improvement and Practice, vol. 6, 2001, pp. 85–96.

[SWEBOOK, 06] Guide to the Software Engineering Body of Knowledge. IEEE, http://www.swebok.org/ (last visit 25.3.2006)

[Wieser, 00] E. Wieser, F. Houdek, K. Schneider. Push or Pull: Two Cognitive Modes of Systematic Experience Transfer at DaimlerChrysler. In G. Ruhe, F.Bomarius (eds): Learning Software Organizations. Methodology and Applications, Springer, 2000, pp. 186-204.