Evaluating a Training Process in a Software Handover Context

Ahmad Salman Khan and Mira Kajko-Mattsson
KTH Royal Institute of Technology
School of ICT
Stockholm, Sweden
askhan@kth.se, mekm2@kth.se

Abstract— **Although there exist some people management process models related to the education and training of software engineers, there are no process models that are adapted to specific software engineering contexts and processes. In this paper, we suggest a set of education and training activities that are applicable in the context of a handover process. We then evaluate these activities within twenty organizations. Although our results reveal great diversity of using these activities, they still show that they are realistic and appropriately mirror the industrial status within a software handover context.**

Keywords-transition; education; knowledge; transfer

I. Introduction

It is common knowledge that people are the most important organizational asset within software companies. The success of the companies is strongly dependent on their employees' knowledge and skills. For this reason, people management process models have been created such as, for instance, *People CMM* and *CM^3: Maintainer's Education and Training* [1], [2]. These models provide guidelines on how to continuously improve the competence of either software developers or maintainers. However, they mainly focus on all the software processes in general. They do not provide any guidelines adapted to a specific software engineering process.

One of very critical and complex contexts is a software handover process (alias software transition process), a process during which a software system is transitioned from developer to maintainer and customer. To assure its success, the process must be treated with great care and caution, and thereby, it must be properly managed. This implies that one must handle and direct it with the right degree of skill, experience, and caution. If not properly managed, it may lead to a communication gap between developer and maintainer, low quality maintenance and customer dissatisfaction. At its worst, it may lead to a delivery failure and loss of customer credibility.

Recently, software handover has become one of the most business critical processes within many software organizations. Due to its diversity, however, it is very challenging to define and implement it. Its design strongly depends on the business context, process lifecycle, product complexity, and customer needs. For instance, its context may range from handing over a software system from developer to maintainer within one and the same company, to handing over from developer in one partner company to another maintainer in another partner company, and finally, to outsourcing the system to a totally unknown party in the clouds [8].

Despite its great importance and business criticality, there are very few software handover process models. To our knowledge, there exists a handful of publications dealing with this process [3, 4, 5, 6, 7]. These are either old or they deal with handover on a very general level. Hence, they do not fulfill current needs of the software industry which today is in a strong need of a well-defined software handover process model [8].

To aid software organizations in developing their handover models, we are in the process of creating *Evolution and Maintenance Management Model* (*EM^3*)*: Software Handover* with the objective of providing guidelines on how to successfully handover a software system from developer to maintainer. On our way towards this journey, as an initial step, we have come up with seven components including activity types that are significant for a successful software transition, and put them into *EM^3 Taxonomy of Handover Activities* [12]. One of its components, *Training* (T), addresses activities for educating and training of the stakeholders involved in software handover. In this paper, we evaluate the *Training* component within 20 organizations. Our goals are (1) to find out whether the component is realistic and whether it meets the needs and requirements of the software industry today, and (2) to create a basis for further research in software handover.

The remainder of this paper is as follow. Section II describes our research method. Section III presents the *EM^3 Taxonomy of Handover Activities*. Section IV evaluates the component using the industrial feedback. Finally, Section V makes conclusions and suggestions for future work.

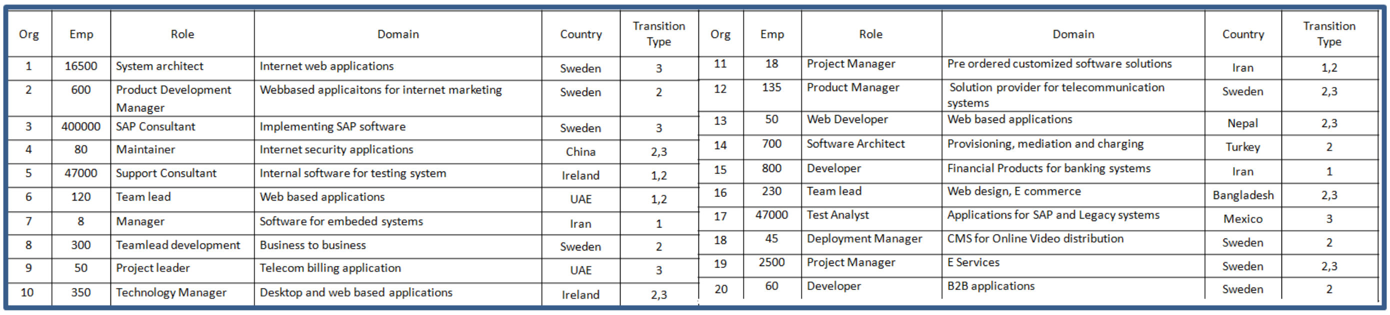
II. Research Steps

In this section, we report on the methods steps taken during our study. We then describe the organizations studied and validity threats.

A. Research Steps

We conducted this research in three major steps. These are (1) *questionnaire design*, (2) *data collection* and (3) *data analysis.* In the *questionnaire design* step, we developed a semi- structured questionnaire discussing five questions to be asked for each activity of the training component. These were: (1) Is this activity performed? (2) Who does exactly perform this activity? (3) When in the lifecycle of the system is it

TABLE I. PARTICIPANT ORGANIZATIONS



Org	Emp	Role	Domain	Country	Transition Type	Org	Emp	Role	Domain	Country	Transition Type
1	16500	System architect	Internet web applications	Sweden	3	11	18	Project Manager	Pre ordered customized software solutions	Iran	1,2
2	600	Product Development Manager	Webbased applicaitons for internet marketing	Sweden	2	12	135	Product Manager	Solution provider for telecommunication systems	Sweden	2,3
3	400000	SAP Consultant	Implementing SAP software	Sweden	3	13	50	Web Developer	Web based applications	Nepal	2,3
4	80	Maintainer	Internet security applications	China	2,3	14	700	Software Architect	Provisioning, mediation and charging	Turkey	2
5	47000	Support Consultant	Internal software for testing system	Ireland	1,2	15	800	Developer	Financial Products for banking systems	Iran	1
6	120	Team lead	Web based applications	UAE	1,2	16	230	Team lead	Web design, E commerce	Bangladesh	2,3
7	8	Manager	Software for embeded systems	Iran	1	17	47000	Test Analyst	Applications for SAP and Legacy systems	Mexico	3
8	300	Teamlead development	Business to business	Sweden	2	18	45	Deployment Manager	CMS for Online Video distribution	Sweden	2
9	50	Project leader	Telecom billing application	UAE	3	19	2500	Project Manager	E Services	Sweden	2,3
10	350	Technology Manager	Desktop and web based applications	Ireland	2,3	20	60	Developer	B2B applications	Sweden	2

performed? (4) How is it performed? (5) Is any activity missing in this component?

In the *data collection* step, we asked students to collect data by conducting interviews in IT organizations. The students were attending a master level course, called *Processes for IT Production* at KTH. This course was offered as part of international master program. Therefore, the students belonged to different parts of the world. Students were free to choose any IT organization working in public or private sector, having small, medium or large size and situated in any part of the world. The only prerequisite was that the selected organization must perform software handover. Since students belonged to different parts of the world, the majority of them selected organizations in the countries of their origin.

Finally, in the *data analysis* step, we analyzed the data collected by the students. We scrutinized answers for any incomplete answers, missing answers and ambiguous information. In case of ambiguities, we asked the students to contact interviewees again for clarifications.

B. *Organizations studied*

This paper includes data collected from twenty organizations from different countries. As shown in Table I, these are Sweden, China, Ireland, United Arab Emirates, Iran, Nepal, Turkey, Bangladesh and Mexico. Eight out of these twenty organizations belong to Sweden.

The organizations studied have different sizes with smallest organization having only eight employees and largest having 400000 employees. They operated in diverse domains such as business to business applications, internet web applications, SAP software, software for embedded systems, telecom billing application, and financial products for banking systems.

C. *Validity Threats*

All case studies encounter validity threats. Regarding external validity, our data sample consisted of 20 small, medium and large organizations working in diverse domains and located in various parts of the world. Although we cannot claim that the results of our study are generalizable, still due to the diverse nature of our sample, we can say with confidence that our results appropriately mirror the current state of the handover process within the organizations studied.

Regarding the construct validity, the risk was that the students might misinterpret the transition process and its results. To minimize this threat, we prepared students for conducting interviews. First, we gave one lecture on transition. We then presented the questionnaire and its purpose and we provided counseling hours. To ensure that all the questions were answered for each transition activity, we created templates listing each question for each activity and arranging space to be filled in with the answers. In this way, we ensured the uniformity of the data and the completeness of all the answers. Finally, to enable additional validation of the answers, we requested that the students provided the contact details of their interviewees.

III. EM^3 TAXONOMY OF HANDOVER ACTIVITIES

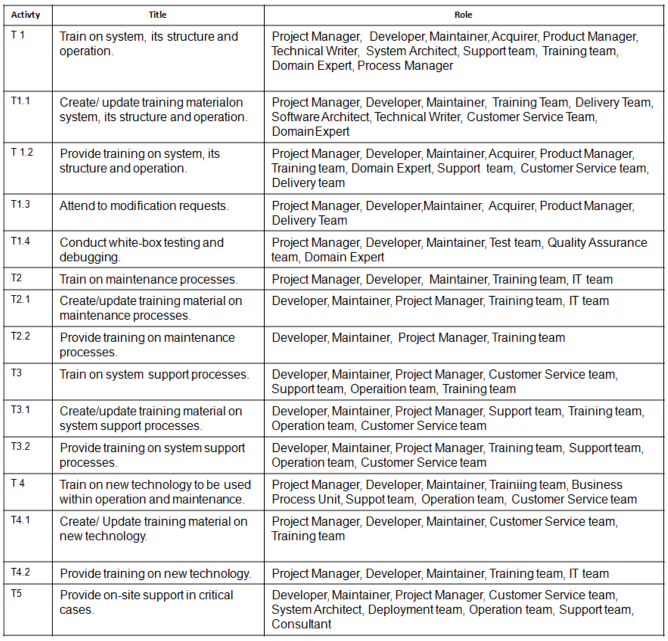
In this section, we first describe all the components in *EM^3: Software Handover*. We then focus on its *Training* component. Due to space restrictions, we cannot fully describe the whole model. We only list its components and briefly describe them. Interested readers are however welcome to study them in [12].

A. *EM^3: Software Handover*

Right now, *EM^3: Software Handover* contains types of activities required for transferring a software system from a development team to a maintenance team. The model has seven components where each component includes a set of highly cohesive activities having a common goal in the transition process. In addition to the training component, *EM^3* includes the following components:

- *Maintenance Environment* listing activities for establishing a maintenance environment at the maintainer's site.
- *Version and Configuration Management* listing activities for tracking the system changes during transition.
- *Deployment* encompassing activities required for installing the system on the acquirer site.
- *Documentation* listing activities needed for developing and transferring documents necessary for future maintenance.
- *Maintainability Management* including activities required for assessing the system and data maintainability.
- *Management and Administration* listing activities required for managing and administrating a transition process. It includes activities for identifying a maintenance team, for establishing a transition team and for developing a transition plan.

TABLE II. TRAINING COMPONENT ACTIVITIES AND ROLES



Activty	Title	Role
T 1	Train on system, its structure and operation.	Project Manager, Developer, Maintainer, Acquirer, Product Manager, Technical Writer, System Architect, Support team, Training team, Domain Expert, Process Manager
T1.1	Create/ update training materialon system, its structure and operation.	Project Manager, Developer, Maintainer, Training Team, Delivery Team, Software Architect, Technical Writer, Customer Service Team, Domain Expert
T 1.2	Provide training on system, its structure and operation.	Project Manager, Developer, Maintainer, Acquirer, Product Manager, Training team, Domain Expert, Support team, Customer Service team, Delivery team
T1.3	Attend to modification requests.	Project Manager, Developer,Maintainer, Acquirer, Product Manager, Delivery Team
T1.4	Conduct white-box testing and debugging.	Project Manager, Developer, Maintainer, Test team, Quality Assurance team, Domain Expert
T2	Train on maintenance processes.	Project Manager, Developer, Maintainer, Training team, IT team
T2.1	Create/update training material on maintenance processes.	Developer, Maintainer, Project Manager, Training team, IT team
T2.2	Provide training on maintenance processes.	Developer, Maintainer, Project Manager, Training team
T3	Train on system support processes.	Developer, Maintainer, Project Manager, Customer Service team, Support team, Operaition team, Training team
T3.1	Create/update training material on system support processes.	Developer, Maintainer, Project Manager, Support team, Training team, Operation team, Customer Service team
T3.2	Provide training on system support processes.	Developer, Maintainer, Project Manager, Training team, Support team, Operation team, Customer Service team
T 4	Train on new technology to be used within operation and maintenance.	Project Manager, Developer, Maintainer, Trainiing team, Business Process Unit, Suppot team, Operation team, Customer Service team
T4.1	Create/ Update training material on new technology.	Project Manager, Developer, Maintainer, Customer Service team, Training team
T4.2	Provide training on new technology.	Project Manager, Developer, Maintainer, Training team, IT team
T5	Provide on-site support in critical cases.	Developer, Maintainer, Project Manager, Customer Service team, System Architect, Deployment team, Operation team, Support team, Consultant

B. *Training component*

The *Training* component includes five activities required to train system stakeholders in a handover context. The stakeholders are maintenance team, support team, acquirer and development team. As shown in Table II, these activities concern training on (1) system structure and operation, (2) on maintenance process, (3) support process, (4) on new technology to be used in maintenance and operation, and finally, (5) on the provision of on-site support.

1) *Training on system, its structure and operaiton*

The first main activity in the *Training* component (*Activity T1*) deals with training on system structure and operation. As shown in Table II, it consists of four sub-activities where the first two deal with the creation of a training material and provision of training to maintainers, support personnel and the acquirer and its end-users whereas the last two deal with supplementary training of maintainers.

The system to be handed over is of interest to many different stakeholders. Stakeholders have different needs, and therefore, they require different levels of system training. For example, the acquirer is only interested in the overall general system presentation while its end-users have to learn how to use the system in greater detail. Support personnel needs detailed training on the overall system structure and detailed information on how to operate the system. Maintenance team, on the other hand, requires an in-depth training on the nitty-gritty details of system structure and operation.

Regarding the first two activities, they are strongly related to each other. To assure that the system training is of high pedagogical quality and that it is relevant for the stakeholder needs, it must be well prepared. For this reason, the first sub-activity deals with the creation and/or update of training material (Activity T1.1). In cases when a totally new system is being created, the training material is created from scratch. In cases when an existing system has been evolved and enhanced with new features, then the training material gets updated. As mentioned above, different stakeholders have different needs. For this reason, this activity may results in several different sets of training material where each set is adapted to different stakeholder needs. After the training material has been created, then the training is provided.

Regarding the second two activities, they may seem somewhat misplaced here, one would think. However, they are not. They are regarded as important training activities of maintenance personnel [6]. The first activity makes maintainers to attend to modification requests, mainly problem reports (Activity T1.3). According to [6], it is an excellent way of not only making them to dig into the system structure and its functionality, but also a good way of making them acquainted with the strong and weak system sides.

Making maintainers to attend to problem reports is also an excellent way of making them prepared for the initial and the most critical and challenging post-delivery phase during which the influx of problem reports coming from end-users is very high. If they are not well prepared to expeditiously solve them, then their credibility may get undermined. By attending to the problem reports before the delivery, maintainers will get well prepared to meet initial post-delivery customer demands. This, in turn, helps them win trust of their future customers.

Attending to modification requests is also beneficial to developers. During transition, they are under pressure to meet the deadline. By attending to modification requests, maintainers relieve them from the burden of attending to all the modification requests. This also gives the maintainers a chance to create a relation with the acquirer, and thereby, to gain their confidence and trust.

The fourth activity (Activity T 1.4) suggests that white-box testing and debugging be conducted. It forces maintainers to look closely at the code structure, algorithms, its paths, decision branches, loops and the data managed by them. This does not only help the maintainers to get acquainted with the basics of the program but also to get acquainted with the programming style and even enable them to evaluate the program from the maintainability perspective.

2) *Training on Maintenance Processes*

The second main activity in the *Training* component deals with training on maintenance process (*Activity T2*). Maintainer needs to know how to conduct maintenance. As claimed in [6],[10], maintainers must understand the objectives behind each process activity. They have to know how to reproduce problem and identify their underlying defects, how to design solutions, how and when to communicate problems both to the management, acquirer and support, and finally, how to react in various unforeseen situations. All this, in turn, leads to better performance and management of maintenance, and thereby, to decreased lead time and increased productivity.

As can be seen in Table II, this activity consists of two sub activities. The first sub-activity suggests that the training material on maintenance processes be created or updated. Although, it is a very important prerequisite, it may be difficult to achieve. As claimed by [6], maintainers may be designated in the last minute, right before the handover takes place, many times, leading to the situations in which maintainers neither have any maintenance process in place, nor are they properly trained for conducting their work. This, in turn, implies that no time and resources have been assigned to create a training material on maintenance process.

Regarding the second sub-activity, it suggests that training on maintenance process be performed. As mentioned above, it is a very crucial activity. Lack of it may lead to mismanagement and mismaintenance of a software system. This, in turn, may cause problems such as corruption of system maintainability, careless and probably unconscious performance of many critical tasks, wrong uninformed decisions and other similar problems.

3) Training on System Support Process

The third activity deals with training on support processes. This activity is just as critical and difficult as any of the above-mentioned activities. Front-end support personnel are the first point of contact towards the customer and the organization's reputation is strongly dependent on how they conduct their work [11]. Support team will have to be operative from the very first day the system has been delivered. Playing the role of an intermediary between the customer and back-end support, they have to be efficient and effective when supporting the daily work of their customers while having had to relieve the maintainers on the back-end support level from dealing with support related tasks. Support personnel must also possess good communication capabilities.

Another reason to why training on support process is critical and difficult is the fact that the tasks of front-end support personnel strongly diverge. As pointed out in [9], front-end support may conduct up to 14 different tasks, where each task requires a separate process. For this reason, front-end support has to have a well-defined front-end support process model that covers the different tasks. This requires that one has to identify the scope of the support tasks and create a training material covering these tasks (Activity T3.1). After that, one may start training (Activity T3.2).

4) Training on New Technology

The fourth activity in the *Training* component deals with training on new technology to be used in the operation and maintenance. As technology evolves, organizations introduce new technology for implementing their systems and for supporting their processes. This, in turn, helps them to increase work efficiency, to improve the quality of work and the quality and reliability of their systems, and thereby, it helps to stay competitive on the market.

New technology solves many problems, but if not properly utilized, then its introduction may create problems, and thereby, lead to lower productivity. If maintainers are not well trained, new technology may backfire and maintainers may spend more time on solving technological problems rather than providing efficient maintenance services. For this reason, to fully benefit from the new technology, the operation and maintenance teams must possess good technological knowledge. This is covered by *Activity T4* in Table II.

Activity T4 contains two sub activities. The first sub-activity (Activity T 4.1) suggests that the training material on new technology be created or updated. This material may be used during (1) group-based teaching form, such as, classes or workshops, or (2) on an individual basis where the teaching material constitutes a self-learning material.

The second sub-activity (Activity T 4.2) suggests that training on new technology be provided. As explained before training on new technology helps the maintenance and operation personnel to perform their duties in a smooth way.

5) Provision of On-Site Support

The last activity *T5* in the *Training* component suggests the provision of onsite support. The need for onsite support arises in two cases. The first case applies when users face problems in using the system. In such a case, support team visits them in order to help users to operate the system properly. The second case applies, when support services are transitioned to the acquirer side. In this case, it may be difficult for the acquirer support team to take on support duties at once. Therefore, a few developers may visit them and help them in resolving daily issues. Developers remain with support team till support team is confident enough to do their work.

IV. EVALUATION

In this section we describe the results of our study. When doing it, we follow the order of the activities as presented in Table II. Due to space restrictions, we could not describe the roles involved in each activity. We have however listed them on the right-hand side column in Table II.

A. Trainig on system, its structure and operaiton

1) Create/update training material on system, its structure and operation (T 1.1)

All except one organization (95%) create and/or update training material on system, its structure and operation. The material is created for both maintenance team and acquirer. It includes user manuals, PowerPoint presentations, functional specification, and wiki material.

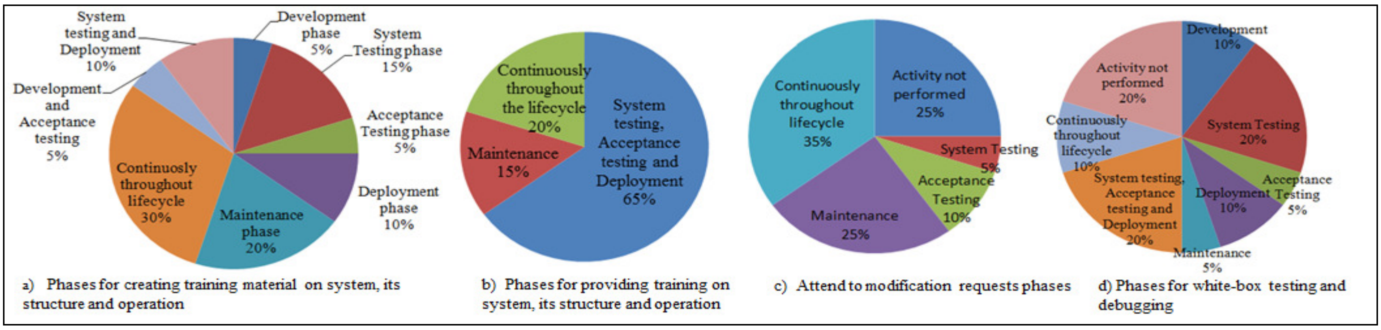
The point in time for creating and/or updating training material varies strongly for the organizations studied. As shown in Figure 1.a, the majority of the organizations studied (6 organizations, 30%) continuously create and update their training material during the whole lifecycle. The second group of organizations always creates and updates their material at the beginning of the maintenance phase (4 organizations, 20%).

Regarding the remaining organizations the point in time varies strongly; three organizations (15%) create and update training material in the system testing phase, two organizations (10%) in the deployment phase, two organizations (10%) begin training in the system testing phase and complete it in the deployment phase, one organization (5%) creates training material training in system testing, one (5%) in acceptance testing and one (5%) in the development phase. Finally, one organization starts creating training material in the development phase and ends it in the acceptance testing phase.

2) Provide trainig on system, its structure and operaiton

All twenty organizations studied provide training on system, its structure and operation. They do it on two different formality levels. Fifteen organizations (75%) conduct it on a formal level by arranging special training sessions. Five organizations (25%) do it rather on an informal level via informal discussions, emails and telephone. The ways of providing system training vary among the companies studied. Below, we briefly list them.

- *Training requires a specially designed environment:* This environment is similar to the operational environment. Without it, the training on system is impossible.



a) Phases for creating training material on system, its structure and operation

b) Phases for providing training on system, its structure and operation

c) Attend to modification requests phases

d) Phases for white-box testing and debugging

Figure 1. Phases for training on system, its structure and operation

- *System is presented on different granularity levels in two consecutive rounds*: The maintenance team gets first an overview of the system. It then receives a detailed training about the structure and its functionality. These training rounds may be provided by different groups, such as, first by project manager and then by the development team.
- *Choice of who trains who varies*: Developers train first maintainers or support personnel (in companies where maintenance stays with development). The maintainers or support personnel then trains the acquirer's end users.
- *Training is context dependent*: In cases when maintenance stays with development, then not much training is done. The developers already possess system knowledge and do not require any special training. They may however require it in cases when developers quit and new people get hired. In this case, the quitter trains the new hire.
- *Training of super users*: Only selected members of the acquirer's support team get trained. These members, in turn, train the end-users in the acquirer organization.
- *Video conference training*: Due to geographical distribution, the travel cost and urgency of training, development organizations transfer system knowledge by organizing video conference training sessions.
- *Internal versus external training sessions*: The *internal* training is conducted for the employees including developers and maintainers. In this case, the employees informally communicate with each other and get help from training material available in the organization's common repository. The *external* training, on the other hand, is provided to the acquirer. Here, developers conduct formal training to the selected users in the acquirer organization.

The point in time when system training is provided varies. As shown in Figure 1.b, thirteen organizations (65%) do it in parallel with the system testing phase and continue with it during the acceptance testing and deployment phases. Three organizations (15%) provide training at the beginning of maintenance phase. Finally, four organizations (20%) consider it a continuous activity starting in the development phase and carrying on during the whole system lifecycle.

3) Attend to modification requests (T1.3)

All except five organizations (75%) involve maintainers in attending to the modification requests as part of training within a handover process (Figure 1.c). The remaining 25% organizations do attend modification requests but they do not consider it as part of training. The context in which attendance to the modification requests takes place varies according to the following:

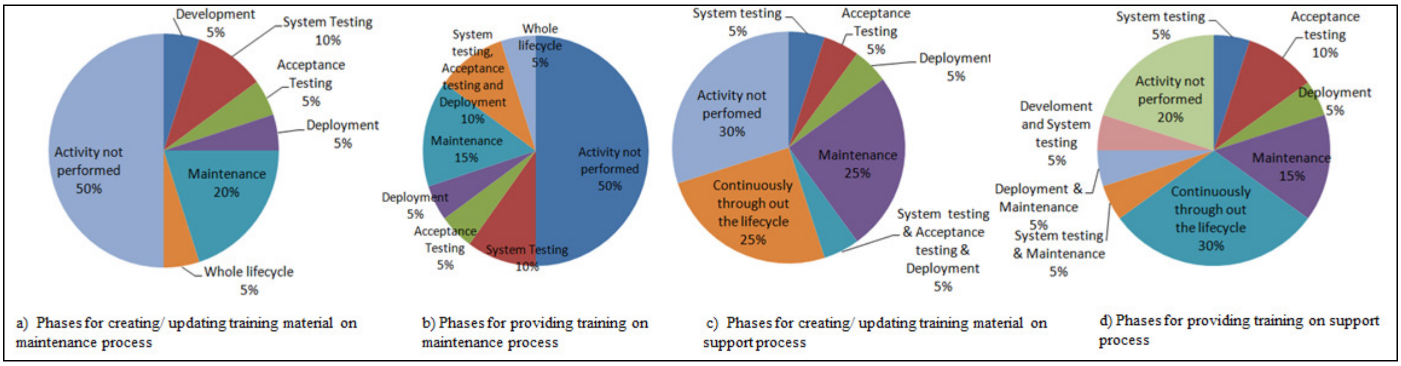
- *Number of modification requests*: If the number of modification requests is low then maintenance team attends to these requests. If it is high, then a delay of the deployment is requested and development team solves them.
- *Modification requests are only attended by new hires*: The newly hired developers attend to all the requests in order to learn the system. This is the case in the companies where maintenance stays with development.
- *The severity and complexity of the modification requests determines the next course of events*: Only minor requests get attended during acceptance testing. If the requests are major, then a new contract is signed with the acquirer.
- *Developer and maintainer together attend to the modification requests*: Both the development and maintenance teams work together to attend to modification requests during acceptance. However, during the deployment phase, the maintenance team takes responsibilities for all the modification requests.

4) Conduct white box testing and debugging (T 1.4)

All except four organizations (80%) claim that they conduct white box testing and debugging as part of the maintainers training process. The way, circumstances and purpose vary among the companies studied. Common to them, however, is that maintainers are assisted by developers.

Usually a *specific environment for conducting white-box testing is created for training purposes.* The changes made in this environment, if any, however, are never uploaded in the operational environment. Only specific problems are white-box tested and debugged by the maintainers for the educational purposes. These problems concern performance and random system errors which cannot be easily reproduced.

The point in time for performing this activity varies in the organizations studied. As shown in Figure 1.d, four organizations do it in system testing (20%), two organizations (10%) do it in development, two organizations (10%) conduct white box testing in the whole lifecycle of the system, and two organizations (10%) in the deployment phase. Finally, one organization (5%) conducts this activity in acceptance testing, and one (5%) in the maintenance phase.



a) Phases for creating/ updating training material on maintenance process

b) Phases for providing training on maintenance process

c) Phases for creating/ updating training material on support process

d) Phases for providing training on support process

Figure 2**.** Phases for maintenance and support processes

B. *Train in maintenance process (T 2)*

1) Create/update training material on maintenance process (T 2.1)

Only 50% of the organizations studied (10 organizations) create and/or update training material on their maintenance processes. The ways of creating the material and making it available varies somewhat. The material is created and/or updated either on a continuous or on an as-needed basis by mainly developers, maintainers and technical writers. The continuous update is done as soon as some change has been made to the process. In this way, one assures that the training material is up-to-date. The "on-as-needed-basis" creation and update is usually realized when major changes have been done to the maintenance process. In all cases, however, the training material is available either online or in some repository.

Regarding the remaining 50% of the organizations, they do not create or update any training material on maintenance processes. Five of them (25%), claim that their developers continue with maintenance and that they already possess process knowledge. Therefore, they see no reason for creating any educational material. Three out of ten organizations (30%) outsource maintenance. Hence, this task is irrelevant in their context. Finally, two organizations have not provided any reason for not creating training material.

The point in time when training material gets created and/or updated varies for the organizations studied. As shown in Figure 2.a, four organizations (20%) create material in the maintenance phase, two in system testing (10%), one organization (5%) does it in the development phase, one (5%) in the acceptance testing phase, one (5%) in the deployment phase, and finally, one organization does it continuously during the whole system lifecycle.

2) Provide training on maintenance process (T 2.2)

The 50% of the organizations that create training material on maintenance processes, also provide training. They do it either via discussion sessions and meetings or arranged training courses. Regarding the remaining 50% of the organizations (10 organizations), they do not perform this activity due to the following reasons:

- Three out of ten organizations outsource maintenance to other maintenance organizations, and hence, they are not concerned with it.
- Five another organizations delegate maintenance responsibilities to developers who already have experience of the maintenance processes. Hence, they do not find any need for getting trained.
- Two organizations conduct informal training only in cases when maintenance process changes.

The point in time when training on maintenance process is provided varies in the organizations studied. As shown in Figure 2.b, 15% of the organizations (3 organizations) provide training at the beginning of the maintenance phase, 10% of the organizations (2 organizations) do it in the system testing phase, 10% (2 organizations) start training on maintenance process in the system testing phase and continue with it during acceptance testing and deployment, 5% (1 organization) do it during acceptance testing and another 5% during deployment (5%). Finally, the last 5% do it continuously in the whole life cycle of the system.

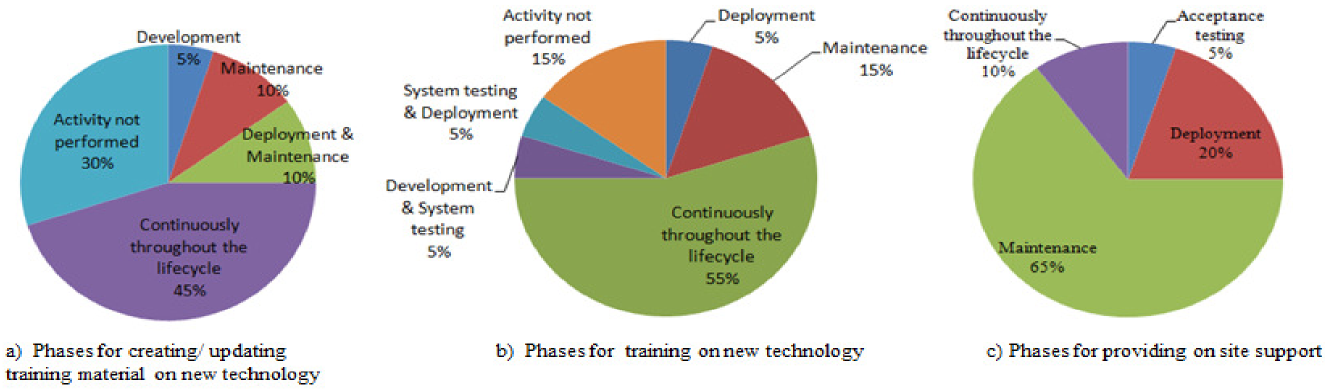
C. *Train on system support processes (T 3)*

1) Create/ update training material on system support processes (T 3.1)

Fourteen out of twenty organizations (70%) create or update material on system support process. Their material provides guidelines on how to report problems and how to handle unexpected situations in the system. This material is available to the whole organization via a central repository. Usually, the organizations studied update their training materials in two cases: either when a new system version is created or whenever a new support problem or difficulty arises. In the first case, a new system version may require new support routines. In the second case, as soon as the organizations have discovered process deficiencies, they attend to them by changing the support process. In both cases, the changes to the support process are documented and communicated to all the support personnel.

Regarding the remaining organizations (6 organizations), that do not create any training material. Five of those six have supported the same systems for a long period of time. They claim that they have good experience in providing support services and they do not require any specific training material. The sixth organization, on the other hand, outsources support services and support process. Hence, creating training on support process is not their concern.

The point in time for creating or updating training material on system support process varies among the



a) Phases for creating/ updating training material on new technology

b) Phases for training on new technology

c) Phases for providing on site support

Figure 3. Phases for training on new technology and provision of onsite support

organizations studied. As shown in Figure 2.c, five organizations (25%) do it continuously throughout the lifecycle of the system and another five organizations (25%) do it in the maintenance phase. One organization (5%) does it in the system testing phase, one in acceptance testing (5%), one (5%) in the deployment phase. Finally, one organization (5%) starts creating training material in the system testing phase and continues on this task during the acceptance testing and deployment phases.

2) Provide training on system support process (T 3.2). All the organizations studied except for four (80%) train their support personnel on system support processes. Forms of training strongly vary. They may either be conducted in a classroom, workshop, meeting or via informal communication channels such as email and telephone. Below, we briefly list the elicited ways of providing training:

- *Scheduled training*: Training on support processes is a frequently scheduled activity. It may be conducted as frequently as on a weekly basis. This raises awareness among the employees about the importance of quality support, and thereby, leads to substantial progress and improved productivity.
- *Needs-based training:* Support personnel are trained according to their needs which are based on the status of their knowledge and learning preferences. Team members may point out their deficiencies and get training on their weak expertise areas or they may get trained as soon as they encounter problems in providing support.
- *SLA-based training*: The training provided is dependent on the scope of the agreement as defined in the Service Level Agreements (SLAs).
- *Two-stage training:* Training is provided in two ways. First, a meeting is arranged with the support personnel during which one provides high-level system information and the process supporting it. If this is not enough, then the personnel may always get additional consultation via email or telephone.
- *Training on support processes is merged with the training on maintenance processes.* In cases when the number of system users is small, front-end support is merged with back-end support. For this reason, the team gets training on both support and maintenance process.
- *Simulation-based training*: Here, the organizations inject defects into the system that are then reproduced by the trainees (the front-end personnel).

The point in time when training on system support process is provided varies among the organizations studied. As shown in Figure 2.d, four organizations (20%) provide training in the maintenance phase, three organizations (15%) do it continuously throughout the lifecycle, two organizations (10%) do it in the system testing phase, two organizations (10%) in the deployment phase and two (10%) begin the training during system testing and continue with it during acceptance testing and deployment. Finally, one organization (5%) does it during system testing and maintenance, and one (5%) performs this activity during the acceptance testing phase.

D. *Train on new technology to be used within operation and maintenance of the system (T 4)*

1) Create/ update training material on new technology (T 4.1)

All except six organizations studied (70%) create or update training material on new technology. The material may be or may be not stored in a central repository. In cases when it is not, then there is a risk that the knowledge stays with individuals. Two of these six organizations outsource maintenance responsibilities. One organization does not introduce new technology due to critical nature of their system. Three organizations do not consider it important to create training material.

The point in time for creating or updating training material on new technology varies. As shown in Figure 3.c, nine organizations (45%) do it continuously throughout the lifecycle of the system, two organizations (10%) do it in the maintenance phase, two organizations (10%) start creating the material in the deployment phase and continue with it in the maintenance phase. Finally, one organization (5%) already creates training material it in the development phase.

2) *Provide training on new technology (T 4.2)*

All except three organizations (85%) train on new technology to be used within operation and maintenance. All of them send their employees on external courses whenever they experience a technological need that they cannot learn on their own. Regarding the internal training, they either conduct it in a seminar form or they assign resources to individuals for the learning activities. The seminars may be led by different groups such as operations team, specifically dedicated technology trainers, developers or other. In the individual learning case, each team member is responsible for studying new technology on his own. However, some enthusiastic members may be assigned the task of learning and then be responsible for disseminating their learning outcomes.

Regarding the 15% of the organizations that do not provide training (3 organizations), two of them outsource the maintenance responsibilities to other organizations, and hence, they do not to train on new technology. The third organization, on the other hand, maintains a financial system. Due to its critical nature, they prefer to use old, reliable and well established technologies instead of introducing the new ones on which they do not dare to rely.

The point in time for training on new technology varies. As shown in Figure 3.b, eleven organizations (55%) do it continuously throughout the system lifecycle, three organizations (15%) do it in the maintenance phase, one organization does it in the deployment phase, and one (5%) starts during development and continues in the system testing phase. Finally, one organization (5%) begins training in the system testing phase and ends it in the deployment phase.

E. *Provide onsite support in critical cases (T 5)*

All the organizations studied provide onsite support in critical cases. However, they do it in different forms. Five organizations (25%) send their experienced consultants to support the maintainer's onsite. Regarding the remaining organizations, they either provide consulting services or arrange video conferences for assisting in critical cases. One of the organizations does it in two consecutive rounds. They first send their deployment team to resolve critical problems. If they fail, then their development team goes to the acquirer site to attend to the problems.

The point in time in the system lifecycle when on-site support is provided varies. However, as shown in Figure 3.c, the majority of the organizations (65%) do it within the maintenance phase. The other organizations do it in the deployment phase (20%), 10% do it continuously throughout the whole lifecycle, and finally, 5% do it during the acceptance testing phase.

V. CONCLUSION

In this paper, we have evaluated the *Training* component in 20 organizations. Our results show that all the activities are implemented by the organizations studied. However, the implementation of these activities strongly depends on the context at hand and the nature of software handover. Due to the strong diversity of our results, it is difficult to make any concrete conclusions. However, we may provide one general conclusion that training starts too late. It generally takes place in the final development phases or initial maintenance phases. Very few companies start within development and do it continuously throughout the whole system lifecycle.

Just as Pigoski claims [6], "*You cannot deliver 1 million lines of code on Friday and start maintaining in on Monday. You must prepare for it!*", so do we. We are of the opinion that the companies provide training too late. This mainly concerns training on system, maintenance and support processes. We believe that the productivity and work quality of maintenance and support groups may be substantially aggravated. However, right now, there is no research on this matter. Hence, we do not know much what is the optimal point in time to start handover training. More studies need to be done to find this out.

REFERENCES

[1] Carnegie Mellon University, Software Engineering Institute. People Capability Maturity Model^{SM}, Technical Report (CMU/SEI-95-MM-02), 1995.

[2] Forssander, S., Olsson, U., Corrective Maintenance Maturity Model: Maintainer's Education and Training, in Proceedings, International Conference on Software Engineering, IEEE Computer Society Press: Los Alamitos, CA, 2001, ISBN: 0-7695-1050-7, pp. 610-619.

[3] April, A., Abran, A.: Software Maintenance Management, Evolution and Continuous Improvement, John Wiley & Sons. 169--174 (2008)

[4] International Organization for Standardization: ISO/IEC Standard 14764, Standard for Information Technology (2006)

[5] MIL-HDBK-347: Military Handbook, Mission critical computer resource software support, (1990)

[6] Pigoski, T.: Practical Software Maintenance, John Wiley & Sons. 117--162 (1997)

[7] Vollman, T.: Transitioning from development to maintenance. In: Software Maintenance Conference, San Diego, CA, USA. (1990) ISBN 0-8186-2091-9

[8] Gustafsson, L.O., Cloud Outsourcing Requires a Proper Handover Process (Position Paper), in Proceedings of International Conference on Advanced Information Management and Service (IMS 2010), IEEE, 2010, in press.

[9] Problems within Front-End Support, Journal of Software Evolution and Maintenance: Research and Practice, John Wiley & Sons, Volume 16, Issue 4-5, July - October 2004, DOI: 10.1002/smr.294, pp. 309-329.

[10] Forssander, S., Olsson, U., Corrective Maintenance Maturity Model: Maintainer's Education and Training, in Proceedings, International Conference on Software Engineering, IEEE Computer Society Press: Los Alamitos, CA, 2001, ISBN: 0-7695-1050-7, pp. 610-619.

[11] Kajko-Mattsson, M., Maturity Status within Front-End Support Organisations, International Conference on Software Engineering, IEEE, Computer Society Press: Los Alamitos, CA, ISBN: 0-7695-2828-7, 2007.

[12] Khan, A. S., Kajko-Mattsson, M.: Taxonomy of Handover Activities. In: Software Maintenance Maturity Model workshop, Limerick, Ireland, S3M (2010), in press.