Demarcating the Scope of a Handover Process

Ahmad Salman Khan, Mira Kajko-Mattsson
KTH School of ICT
Sweden
askhan@kth.se, mekm2@kth.se

Abstract—Despite the fact that a handover process is just as frequently performed as any development process, little is known about it. Still, it is regarded as one of the lifecycle processes that is not well explored and defined. In this paper, we study the handover process within eighteen companies with the purpose of demarcating its scope within software lifecycle. Our goal is to find out how industry understands handover process and how it places it within software lifecycle. As a result, we have identified seven different scope contexts for the handover process. We have also provided evidence of its wide lifecycle span and its overlap with development, predelivery and postdelivery maintenance processes.

Keywords-transition, predelivery maintenance, transition, shift, rollout, deployment.

I. INTRODUCTION

All software organizations hand over software systems from development to maintenance in one way or the other. At a minimum, developers hand over software systems from self to self. At a maximum, developers hand over software systems to several different maintenance organizations. Irrespective of who transfers to who, a handover process, alias transition, is a very critical element in a software lifecycle [2, 6, 7]. Despite this, very little attention has been paid to the handover process within the academia. Even the software industry has not stated any clear requirements for exploring this important domain. Hence, handover process is in a desperate need of more research.

In this paper, we demarcate the scope of a handover process. We do it within fifteen companies. Our goal is to find out how industry understands handover and how it places it within software lifecycle. This paper related to [5] dealing with predelivery maintenance. Hence, the results presented in this paper are strongly related to the results presented in [5].

II. METHOD

In this section, we present the method taken in this research. Section II.A describes our research steps. Section II.B discusses the sampling and validity of the results presented herein. Just because this paper is closely related to [5], the method presented herein is very similar to the method presented in [5].

A. Research steps

Our research method consisted of four main steps. These are (1) literature study (2) questionnaire design (3) data collection and (4) data analysis. Below we briefly describe them.

In the literature study step, we studied the scientific publications and reports on handover maintenance. Our goal was to gain as much knowledge as possible about this domain. Unfortunately, we only found very few publications dealing with the subject. These are [6, 7, 10]. They provide a platform for the work presented in this paper.

During the second step, the questionnaire design step, we created a semi-structured questionnaire. The questionnaire was developed using evaluation criteria for the purpose of exploring whether, how, when and by who the handover activities were conducted in the software industry. The evaluation criteria and the questionnaire are going to be further discussed in Section IV.

To optimize the interview results, we conducted the questionnaire design and data collection steps interchangeably in two iterations. We did so with the intention to assure that the order of questions was relevant and that the questionnaire was complete.

The iterative approach between the questionnaire design and data collection steps implies that we interviewed and collected data in two consecutive iterations. As outlined in Fig. 1, we first created a preliminary version of a questionnaire and using it, we made interviews within three organizations. Our goal was to evaluate that the interviews went smoothly with the initial questionnaire design and to improve it by using experience from the first three interviews. We did so because we used students within the final data collection step.

The initial interview resulted in an addition of one question – Question 9. In its first version, we had eleven

Figure 1. Questionnaire design and data collection steps.
questions. In its second version, we had twelve questions. For this reason, the results presented for Question 9 will only correspond to the results as elicited by the students. It will not include the results elicited in the first three interviews.

Before conducting the second interview, we had a step between the questionnaire design and data collection steps during which we prepared students for conducting the interviews. Finally, in the data analysis step, we compiled the answers for each question and synthesized and analyzed them. It is these results that constitute the body of this paper.

B. Sampling and Validity

In this study we used the convenience sampling method [9]. This means that we did not control the choice of the organizations involved in our study. Regarding the choice of the first three organizations, we chose those who were available. Regarding the organizations chosen by the students, we allowed the students to choose just any organization (large/medium/small and/or private/government) in any country. The only requirement was that the organizations studied should conduct handover process.

Many of our students, coming from an international master program in Sweden, chose organizations in their own countries. Hence, the countries represented in this report are Pakistan, Germany, Turkey, Bangladesh, Iran, France, Thailand, and Sweden. Table 1 presents the details of the organizations studied. Due to the sensitivity of the material presented herein, we do not name these organizations. Some of them however are major well-known multinational companies.

We are aware of the fact that the validity of the results may be strongly jeopardized when one involves students. To optimize the interview results and to secure their credibility, we did the following:

- **Ordered questions**: We made the effort to assure that each question transitioned smoothly from previous questions. Questionnaires that jump from one unrelated topic to another feel disjointed and are not likely to result in effective interviews. We also ordered questions with respect to their ease.

- **Used figures to support the interviewing process**: To improve the communication between the interviewers and interviewees, we created three lifecycle roadmaps and numbered them. These are presented in Fig. 4. In this way, we secured that the two groups talked the same language.

- **Prepared students for the interviews**: To assure the quality of the data collection process performed by the students, we prepared them in several consecutive steps. We first gave a lecture on the subject. We then had three seminars during which we presented the questionnaire and explained and motivated each of its questions. To ensure that the questions were not misunderstood by the students and the interviewees, we included guidelines for asking each question in the questionnaire. Due to space restrictions, we cannot show these guidelines in this paper. However, we may provide them to our readers on request.

- **Required personal data from the interviewees**: To ensure the traceability of the interview results and the authenticity of the collected data, we requested that the interviewees provided their contact information.

- **Scrutinized student results**: We scrutinized all the answers provided by the students. In case of ambiguities in the answers, the students were asked for clarifications and complimentary additions.

Our study was of a strong explorative character. Here, we wished to find out as much as possible about the scope of the handover process. The sampling method, however, does not allow us to generalize our results. Despite this, we believe that the results presented herein strongly contribute to increasing the visibility into the handover process.

III. HANDOVER PROCESS

Handover or transition is a process of transferring responsibilities for a software system from development to maintenance [6, 7, 10]. In general, it implies the transfer of a software system and its data, transfer of responsibilities for this system, and transfer of system knowledge and experience.

It is not possible to view software transition as an isolated process. It is tightly related to other lifecycle processes such as predelivery development and postdelivery evolution and maintenance. Activities performed during development have great impact on a successful transition and the activities taking place after the transition are strongly impacted by what happens both before and during the transition process. Hence, as shown in Fig. 2, software transition should always be considered in the context of Predelivery, Transition and Postdelivery lifecycle stages.

IV. EVALUATION CRITERIA

To reach our exploration goal, we have defined a set of evaluation criteria. These are (1) respondents and their
organizations, (2) process usage and terminology (3) placement within lifecycle, and (4) handover activities and roles involved. These criteria are materialized in a questionnaire shown in Fig. 3. Below, we briefly describe and motivate them.

A. Respondents and their organizations

We asked about the interviewee’s organization and the roles they performed within these organizations. Using Questions 1-4 in Fig. 3, our primary objective was to assure that the right roles were interviewed. Our secondary objective was to understand the domain of the organizations studied and ensure the credibility of the answers. Therefore, we asked about the interviewees’ name, email, contact telephone number and their role within their respective organization. We also requested some basic information about the organization.

B. Process usage and terminology

Before conducting study in any organization, it was essential to figure out that it performs handover activities. Therefore, we asked about whether the organization performs transition (see Question 5 in Fig. 3). Our goal was to include only relevant organizations in our study. Since organizations adopt different terminology according to their needs, we also asked about the term used for transition (see Question 6 in Fig. 3).

C. Placement within lifecycle

We presented three lifecycle roadmaps to our interviewees and asked them to place maintenance on them. Just because there are many different ways of presenting lifecycle, we presented three different roadmaps. These are presented in Fig. 4. In this way, we ensured that our interviewees at least pointed out one handover/transition in one of the roadmaps.

The roadmaps show different perspectives of software lifecycle. These are (1) product perspective illustrating product transfer between major and minor revisions (see Question 7) [2], (2) process phase perspective visualizing the transfer between the phases in the whole software lifecycle (see Question 8) [8] and (3) development cycle perspective visualizing the phases within a software development cycle (see Question 9).

D. Transition activities and roles

We asked the interviewees about the organizational groups and activities involved in the transition phase. Our goal was to find out what activities were performed within a handover process and what roles performed them. Regarding the organizational groups, we first inquired about the participation of a developer, maintainer, acquirer and any other groups involved in transition (see Question 10). We then continued by eliciting the activities performed by these organizational groups (see Question 11). Finally, we inquired about the roles performing transition activities for all the organizational groups involved in the process (see Question 12).

V. RESPONSES

In this section, we present the interview results. When doing it, we follow the order as defined in our questionnaire.

A. Process usage and terminology

All of the organizations studied perform transition. However, they do not always call it transition or handover. They may name it as rollout process, shift, deployment, übergabe (in German), acceptance and handover, handover to local support, handing over process or they may have no name for this process at all.
In fifteen of the organizations studied, transition is conducted on an internal basis. This means that one transfers the system from one group within the organization to another group within the same organization. In the remaining three organizations, transition takes place among separate organizations. Two of these three organizations are the maintenance organizations, the ones that take over systems from the development organization.

B. Placement within lifecycle

Our interviewees were asked to point out handover on at least one of the three lifecycle roadmaps in Fig. 4. Respondents could choose the roadmap relevant to their organizations and contexts. However, in most cases, they pointed out transition on more than one roadmap. Below we present the results for each roadmap.

1) Product lifecycle perspective

Using Roadmap 1, the majority of the organizations studied (15 out of 18) place handover as an activity taking place between major versions. Regarding the remaining three organizations, they do not place transition at this point in the lifecycle.

The majority of the organizations studied (10 out of 18) regard transfer from major version to minor revision as part of the handover process. Regarding the remaining organizations, only two of them justified the reason for not placing transition at this point. In the first organization, the maintainer is not responsible for making changes to the product; he only supports the product. In the second organization, being an ERP solution provider, one does not release minor revisions at all.

Our interviewees were requested to identify other points in Roadmap 1, during which transition takes place. Six out of 18 organizations provided the following answers:

- Handover occurs whenever one changes the system. Hence, transition occurs at every interface in Roadmap 1.
- Handover depends on the complexity, size and customer demands in the project. In this case, each version should provide added business value and customer should approve the functionality of the version. Only then, a handover may take place and system can move on to another version.
- Handing over a system to maintenance is an economical and political issue. The system is transitioned only if it is no longer enhanced with new functionality and when only less costly maintenance services remain to be done.

2) Process lifecycle perspective

Using Roadmap 2, the organizations were requested to place transition on it. The majority of them (14 out of 18) place transition between the initial development and evolution phases. Three of these 14 organizations also place transition after every evolution iteration. Two other organizations place transition between evolution and servicing phases. Regarding the remaining organizations, they had difficulties to place transition on this lifecycle roadmap. However, their interviewees identified three major factors originating handover. These are (1) change in the system, (2) maintenance cost, and (3) market demands.

3) Development lifecycle perspective

Using Roadmap 3, the interviewees were requested to place the handover process on the development cycle. Here, the interviewees had divided opinions. They were:

- Five out of fifteen organizations studied start transition process during the integration and system testing phase.
- Four organizations start transition after the integration and system testing phase.
- Two organizations begin transition after the requirements definition phase.
One organization starts transition between design and implementation phase.

One organization conducts transition in three major stages: (1) the stage between requirements definition and system design during which both development and maintenance teams discuss and agree on the deliverables to be handed over, (2) the stage between the implementation and integration and system testing where the proposed solution is transitioned to maintainer and System Line Architect for approval, and (3) the stage after acceptance testing during which the system is moved on to production and maintainer takes over the responsibilities for the system.

### Handover activities and roles

In this section, we describe roles and activities performed by developer, maintainer and acquirer during the handover phase. We also present the roles involved on each side.

#### Developer’s activities and roles

All the organizations studied listed the activities performed on the developer side during transition. These activities are listed in Fig. 6. They span from development of maintainable architecture to resolving post deployment software problems.

<table>
<thead>
<tr>
<th>Developer</th>
<th>Maintainer</th>
<th>Acquirer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Roles</strong></td>
<td><strong>Roles</strong></td>
<td><strong>Roles</strong></td>
</tr>
<tr>
<td>Project Manager</td>
<td>Transition Manager</td>
<td>Business Analyst</td>
</tr>
<tr>
<td>Release Manager</td>
<td>Release Manager</td>
<td>Product Manager</td>
</tr>
<tr>
<td>Development Manager</td>
<td>Change Manager</td>
<td>Network Engineer</td>
</tr>
<tr>
<td>Test Manager</td>
<td>Trainer</td>
<td>Operation Engineer</td>
</tr>
<tr>
<td>System Architect</td>
<td>System Administrator</td>
<td>System Configuration Manager</td>
</tr>
<tr>
<td>Domain Expert</td>
<td>Test Coordinator</td>
<td>Super User</td>
</tr>
<tr>
<td>Analyst</td>
<td>Maintenance Manager</td>
<td></td>
</tr>
<tr>
<td>Application Developer</td>
<td>Coordinator</td>
<td></td>
</tr>
<tr>
<td>Test Engineer</td>
<td>Project Manager</td>
<td></td>
</tr>
<tr>
<td>Quality Engineer</td>
<td>Quality Engineer</td>
<td></td>
</tr>
<tr>
<td>User Interface Designer</td>
<td>Configuration Manager</td>
<td></td>
</tr>
<tr>
<td>Technical Writer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trainer</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Activities</strong></td>
<td><strong>Activities</strong></td>
<td><strong>Activities</strong></td>
</tr>
<tr>
<td>Signoff requirements.</td>
<td>Communicate with development team during transition.</td>
<td>Confirm the operational needs of the system.</td>
</tr>
<tr>
<td>Identify major modules and map requirements against the developed modules.</td>
<td>Work with developer to clearly define the criteria for transition termination.</td>
<td>Establish operational, functional and security requirements.</td>
</tr>
<tr>
<td>Develop technical documentation.</td>
<td>Approve design documents.</td>
<td>Check existence of identical systems.</td>
</tr>
<tr>
<td>Create a list of system enhancements.</td>
<td>Assess and satisfy manpower requirements.</td>
<td>Perform business impact analysis of the new system.</td>
</tr>
<tr>
<td>Guide technical writer for development of release notes.</td>
<td>Create maintenance plan together with acquirer.</td>
<td>Make initial cost estimation and confirm funding.</td>
</tr>
<tr>
<td>Develop installation notes.</td>
<td>Accept service level agreement.</td>
<td>Create maintenance plan together with maintainer.</td>
</tr>
<tr>
<td>Develop user manuals.</td>
<td>Gain system knowledge.</td>
<td>Develop service level agreement.</td>
</tr>
<tr>
<td>Develop IT security documentation.</td>
<td>Understand code.</td>
<td>Sign the handover protocol.</td>
</tr>
<tr>
<td>Develop user manuals and administrative manuals.</td>
<td>Understand backup recovery.</td>
<td>Approve IT security procedures.</td>
</tr>
<tr>
<td>Develop training material.</td>
<td>Review system functionalities.</td>
<td>Finalize system requirements.</td>
</tr>
<tr>
<td>Develop system architecture material.</td>
<td>Understand key process areas and their effect on the system.</td>
<td>Establish test plan for the system.</td>
</tr>
<tr>
<td>Assist in documentation of development process.</td>
<td>Attend training sessions.</td>
<td>Attend product training courses.</td>
</tr>
<tr>
<td>Perform unit testing.</td>
<td>Analyze bug reports.</td>
<td>Perform acceptance testing.</td>
</tr>
<tr>
<td>Prepare the handover protocol and forward it to acquirer.</td>
<td>Write system instructions for service desk.</td>
<td>Assess the readiness of using the new system.</td>
</tr>
<tr>
<td>Communicate with the operators and maintenance teams.</td>
<td>Review and update support documentation.</td>
<td>Give approval to make the system operational.</td>
</tr>
<tr>
<td>Upload the modules into test environment.</td>
<td>Participate in integration testing.</td>
<td>Report software defects after deployment.</td>
</tr>
<tr>
<td>Train maintenance and operation teams.</td>
<td>Participate in acceptance testing.</td>
<td>Approve that no more post live (warranty period) support is required.</td>
</tr>
<tr>
<td>Train maintainer for debugging and troubleshooting techniques.</td>
<td>Perform blackbox testing.</td>
<td>Provide suggestions for future enhancements.</td>
</tr>
<tr>
<td>Transfer application knowledge to maintainer.</td>
<td>Perform SQA testing.</td>
<td></td>
</tr>
<tr>
<td>Train acquirer for utilizing system.</td>
<td>Assess the maintainability of the system.</td>
<td></td>
</tr>
<tr>
<td>Explain and clarify acquirer’s doubts.</td>
<td>Obtain acquirer production environment information.</td>
<td></td>
</tr>
<tr>
<td>Conduct domain sharing session.</td>
<td>Perform readiness assessment for support.</td>
<td></td>
</tr>
<tr>
<td>Perform readiness assessment.</td>
<td>Setup maintenance environment.</td>
<td></td>
</tr>
<tr>
<td>Prepare a list of outstanding faults and functionalities.</td>
<td>Participate in system deployment.</td>
<td></td>
</tr>
<tr>
<td>Baseline source code.</td>
<td>Work as first contact person during system delivery.</td>
<td></td>
</tr>
<tr>
<td>Build, bundle and deploy software artifacts.</td>
<td>Setup help desk.</td>
<td></td>
</tr>
<tr>
<td>Handover code to maintainer.</td>
<td>Understand all outstanding faults and functionalities.</td>
<td></td>
</tr>
<tr>
<td>Perform defect fixing.</td>
<td>Baseline source code.</td>
<td></td>
</tr>
<tr>
<td>Conduct extra helping sessions with maintenance team on demand.</td>
<td>Provide level 1 and level 2 support.</td>
<td></td>
</tr>
</tbody>
</table>

**Other roles**

- System Architect
- Data Architect
- Database Administrator
- Business Analyst
- Software Configuration Manager
- Human Resource Team
- IT Coordinator
- Educational Instructors

**Other roles activities**

- Plan and analyze the handover process with development and maintenance managers.
- Develop a transition schedule.
- Estimate manpower and cost requirements.
- Coordinate project activities.
- Coordinate during whole handover process.
- Develop organizational charts, new roles and job descriptions.
- Evaluate the skills and competency required.
- Accomplish new staff requirements.
- Ensure that release performed according to release plan.
- Setup database configuration.

Figure 6. Roles involved and activities conducted in the transition phase.
The portfolio of handover activities is very wide ranging from requirements signoff to documenting the system, to providing training, and finally, to resolving post deployment software problems.


2) Maintainer’s activities and roles

The activities as performed by the maintainer role slightly differ than those as performed by the developer. As shown in Fig. 6, they mainly focus on realizing some of the predelivery activities, on assisting developer in various activities such as testing and in preparing and realizing transition. Our interviewees have identified twelve different roles on the maintainer’s side. These are (1) Operation Manager, (2) Transition Manager, (3) Release Manager, (4) Change Manager, (5) Trainer, (6) System Administrator, (7) Test Coordinator, (8) Maintenance Manager, (9) Maintenance Coordinator, (10) Project Manager, (11) Quality Engineer, and (12) Configuration Manager.

3) Acquirer’s activities and roles

The majority of the interviewees (11 out of 18) have even listed activities on the acquirer’s side. These are presented in Fig. 6. As can be seen there, they range from establishing requirements to performing acceptance testing, and to, finally, approving the system.

The organizations studied have identified five different roles on the acquirer side. These are (1) Super User, (2) Operation Engineer, (3) Network Engineer, (4) Product Manager, and (5) Business Analyst.

4) Other roles activities

Our interviewees (12 out of 18) have also mentioned other groups involved in the transition phase and listed activities performed by them. The roles they have mentioned are (1) System Architect, (2) Data Architect, (3) Database Administrator, (4) Business Analyst, (5) Software Configuration Manager, (6) Human Resource Team, (7) IT Coordinator and (8) Educational Instructor.

As can be seen in Fig. 6, the activities as listed by our interviewees mainly focus on planning and coordination between parties involved in transition, estimation of manpower and cost requirements, and development of transition schedule.

VI. CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

In this paper, we have studied the scope of a handover process. We did it within eighteen organizations. Our results indicate that handover is a very complex process involving a wide range of roles and activities. The list of roles and activities as elicited in this paper is not exhaustive. It only represents a subset of handover process activities. However, this initial list already provides evidence that the transition process spans over the entire development and some initial parts of postdelivery maintenance.

Using Fig. 7, we place the elicited handover activities on Roadmap 3. As can be seen there, the activities start during the requirement definition phase by establishing and finalizing requirements and terminate after successful delivery of the system.

Roadmap 3, however, is not sufficient to exhibit the span of handover activities. Therefore, we use business cycle roadmap taken from [3] to show the range of handover activities. In this roadmap (see Fig. 8), handover activities start during the Identify Business Opportunity phase by
confirming operation needs of the system and terminate at
the beginning of the postdelivery maintenance phase.

Our study also shows that the activities in the handover
process overlap with some of the activities in the
development, predelivery maintenance and postdelivery
maintenance. We illustrate it with a Venn diagram in Fig. 9.
Intersection between these processes exhibits common
activities. For instance, development of system architecture
documentation and mapping requirements to modules is part
of development as well as handover. Similarly gaining
system knowledge during development is part of predelivery
and handover. Post live warranty period support is part of
handover and postdelivery maintenance. Non-intersecting
parts in the Venn diagram, on the other hand, represent the
activities that entirely belong to each process. For instance,
handover protocol development and approval belong to
handover process only.

In this study, we have observed that handover
complexity depends upon the nature of the organizational
structure and their processes. We have identified seven
different scope contexts for the handover process. These are
visualized with bold arrows in Fig. 10. We present these
perspectives below.

- **Context 1**: Handover happens only between major
  versions. Minor revisions include corrective
  maintenance changes only.

- **Context 2**: Transfer from major version to major version
  and to minor revision is considered as handover. Major
  versions include enhancive maintenance changes
  whereas revisions include minor improvements and
corrective changes.

- **Context 3**: Transition takes place from initial
development to evolution stage. System evolution is
  part of maintenance.

- **Context 4**: Handover occurs after every evolution
  iteration. System evolution is considered to be part of
development. Maintenance includes only corrective
activities.

- **Context 5**: Handover occurs after initial development
  and every evolution stage. Initial development and
evolution are regarded as part of development.
Handover is originated for acquiring low cost
maintenance services after ensuring that the product
will no longer be enhanced with new functionality.
Only corrective maintenance activities are executed
after handover.

- **Context 6**: All the activities from requirement definition
till system testing are part of development. Handover
gets initiated during acceptance testing and is executed
in parallel with deployment.

- **Context 7**: Transition is implemented in three major
  stages (1) the stage between requirements definition
  and system design, (2) the stage between the
  implementation and system testing, and (3) the stage
  after acceptance testing.

Figure 8. Placing transition activities on business cycle roadmap. D, M and A stand for Developer, Maintainer and Acquirer.

Figure 9. Venn diagram illustrating process relationships.
Our study exhibits a wide span and complexity of a handover process. It shows evidence that to develop a generic handover process model will be difficult considering its strong intersection with other processes, its wide lifecycle span and the different scope contexts in which it may be placed.

This study is one of the initial steps in our research towards developing a framework for a handover process model. We feel that we need to conduct more studies to explore the process in various contexts. Probably, we will have to conduct separate studies for each context at hand and then identify some common generic patterns. Right now, we do not know what is the most optimal step to be taken next. What we know, however, is the fact that a great big challenge is ahead of us.

REFERENCES