# A Unified RE Approach for Software Product Evolution: Challenges and research agenda

Pnina Soffer
MIS Department, Haifa University, Carmel Mountain 31905 Haifa, Israel, spnina@is.haifa.ac.il
Leah Goldin
Golden Solutions, P.O.B 6017, Kfar Saba, Israel, L_goldin@computer.org
Tsvi Kuflik
MIS Department, Haifa University, Carmel Mountain 31905 Haifa, Israel, tsvikak@is.haifa.ac.il

**Abstract**
Requirements engineering plays an important role in the evolution of software products, where a product evolves through new releases that are developped considering market prospects rather than a specific customer. The challenges posed in this situation, which differ from customer-specific RE challenges, have hardly been emphasized in the RE literature. The paper presents an evolution scenario, outlining concurrent RE cycles that take place within this scenario. The cycles relate to the sales process of the current release, the development processes of the next release and possibly an of an interim customer-specific release, and the planning process of future releases. The paper characterizes the RE processes that comprise these concurrent cycles and their relationships, and identifies the need for managing the entire scenario in an integrative manner. This forms a basis for specific challenges as an agenda for future research.

**Keywords**: RE for product software, Software evolution, Requirements management

**Submission category**: position paper

## *1. Introduction*

As of today, the main business trend of software companies is the development of off-the-shelf software products. Two major requirements engineering phases exist in the life of off-the-shelf software products. The first one is an initial development of a new product. There may be no real customers at this time, and requirements are defined on the basis of market analysis and expectations (Potts, 1995). The second phase is an ongoing one, when a product already exists and evolves as new releases are developed periodically. This phase acknowledges the success of the product, as there is an increasing number of customers purchasing it, and these customers require more and more features as part of their growing taste. As well, targetting new market prospects as well as the availability of new technologies may lead to additional requirements.

The main focus in the RE literature so far has been towards requirements posed by a specific customer, for a specific contract, through interaction with such customer (e.g., [2]). This does not explicitly correspond with the reality of RE for software product evolution, where requirements are collected and combined from various sources, then analysed and prioritized in order to generate the content of the next product release, in parallel with the preparation of (possibly) few other releases to follow. As well, traditional RE approaches take a formal view and are developed by and for software engineers, while requirements in product maunfacturing companies are frequently defined by marketing people, taking business-oriented considerations.

Some attempts to address this situation have been made during the last decade. Potts [10] described the product RE situation of “imaginary requirements for imaginary customers”. [9][13][15] indicated the various challenges of this situation, attempting to create a focus of the research community on these challenges. Specific issues that gained some attention are requirements prioritization [11][12][12], commonality and variability management [7], requirements dependency [1][9], and release planning [1][3][4][5] including product vs. project tradeoffs [8]. Other issues, which are addressed also in the context of “traditional” RE, such as requirements traceability and configuration management, are of relevance to the product evolution situation. However, these attempts address specific issues one at a time, and do not take a holistic view, which we claim is essential, due to the relationships among all the activities which comprise an RE process.

Recently, attempts to empirically study and characterize the product evolution situation have been made, addressing Market-Driven RE in general [3][6][9], priritization [11], and release planning [3]. We view these attempts as very important steps towards gaining an understanding of the practice and needs in this situation. However, a full understanding is yet to be achieved.

This paper proposes a scenario of product evolution, outlining concurrent RE cycles that take place within this scenario. The scenario is based on a combination of information obtained from the literature and observations from our accumulated industrial experience with a number of companies, both as a customer and as a requirements engineering consultant, working with software vendors. It characterizes the RE processes that comprise these concurrent cycles and their relationships, and identifies the need for managing the entire scenario in an integrative manner.

## *2. Product Release Lifecycles*

A common scenario of a product developing organization is as follows. At present there exists a current release (denoted as release N), which is being sold as an off-the-shelf product. The next release (N+1) is being developed and is planned to become commercial at a known time in the future. At the same time, requirements for future releases (N+2, N+3..) are already being collected.

Figure 1 presents a typical schema of a software product release lifecycle in a product development company, that is a continuous evolution of the product with respect to market needs. During the development of release (N), Marketing was engaged in learning the market, identifying business opportunities, and learning about their competitors. All that information should be available at the beginning of the next product release development (N+1), which starts with Marketing producing the Marketing Requirements Document (MRD) that contains the market requirements for release N+1.

Once the MRD is handed-over to the Research & Development (R&D) group, the R&D analyses the Software Requirements Specifications (SRS) based on the current product release (N). The SRS is then used as the basis for implementation and testing of the new product release (N+1).

Note that the MRD is a result of planning and analysis, rather than a set of "raw" requirements. The further analysis towards the SRS is of a more technical nature. [5] refers to this by distinguishing market requirements from business requirements, which are the result of analysis.

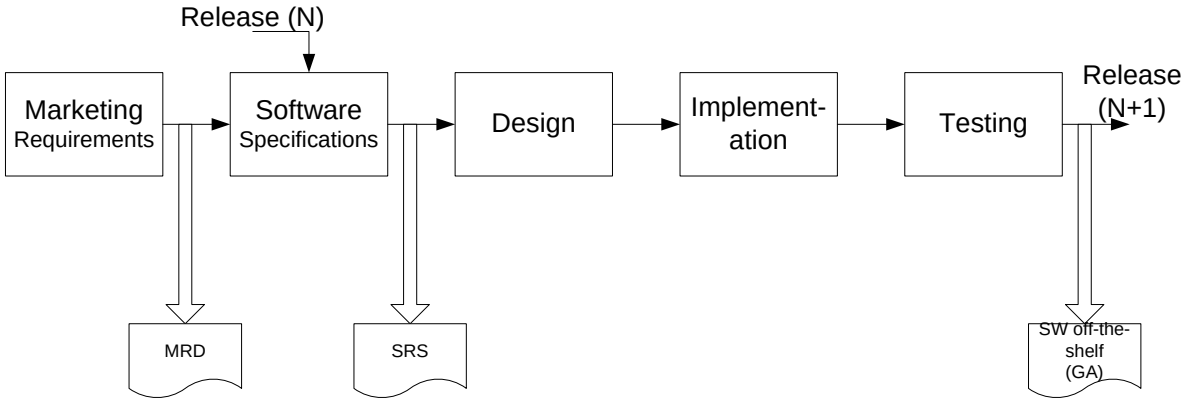


Figure 1: software product release lifecycle

As presented above, the company is occupied at the same time with release N (currently sold to customers), release N+1 (under development, with specific delivery date), and future releases (requirements identification, prioritization, and scoping for future versions).

This development process is only a representative part of a larger process of product roadmap planning and evolution. The overall product road map is illustrated in Figure 2. The release (N+1) lifecycle is normally called project lifecycle and is bounded by scope, schedules and budget.

At this situation assume a strategic customer demands specific adaptations to be made to the product to meet his special needs. This means another project development cycle (release N') in parallel to the release (N+1) lifecycle. Unlike the other development projects, here the requirements address a specific customer, and are based on a gap analysis between the current release (N) and the customer's needs. This gap analysis originates in the pre-sales process, where the current release (N) is being sold to customers. During this process, the specific customer's requirements (e.g., [14]) are captured in view of the current product release. The aim of RE in the pre-sales process is to elicit customer's requirements and accommodate them within the current release (N) features. However, if major gaps are found, they may result in a customer-specific version of the product (N'). The requirements for release N', once approved, are sometimes termed "commitments".

Note that although these requirements are included in release N', they are not a part of the product roadmap, and should be considered and evaluated for deciding whether they will be included in a future release (and which one).

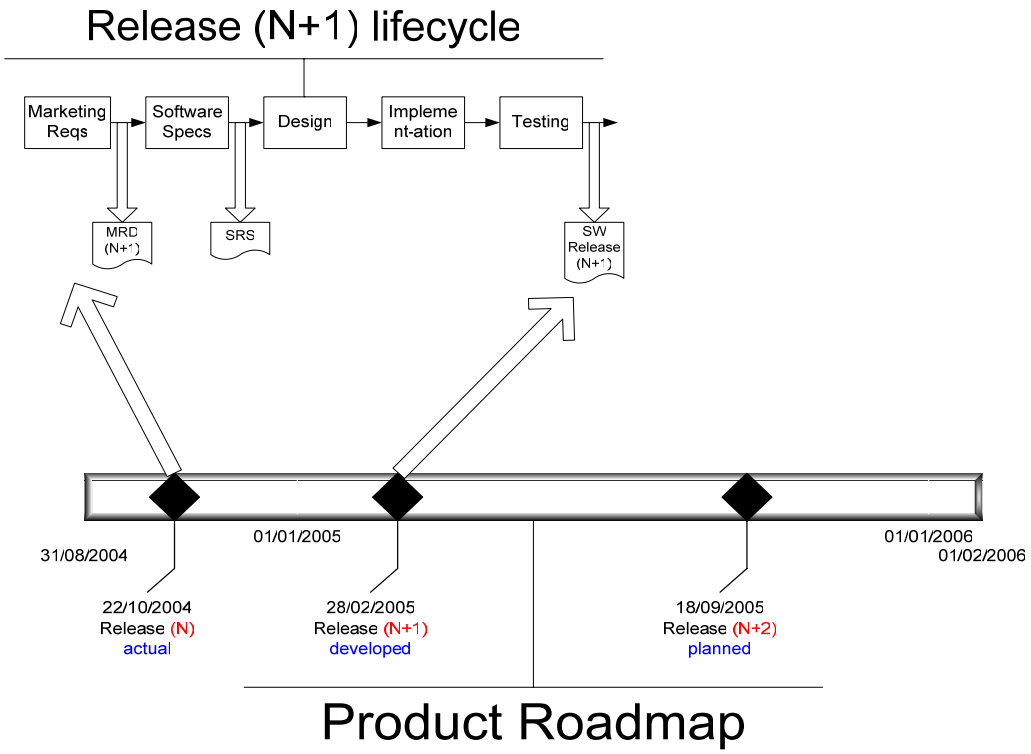


Figure 2: Product roadmap

## *3. RE Cycles in Product Evolution*

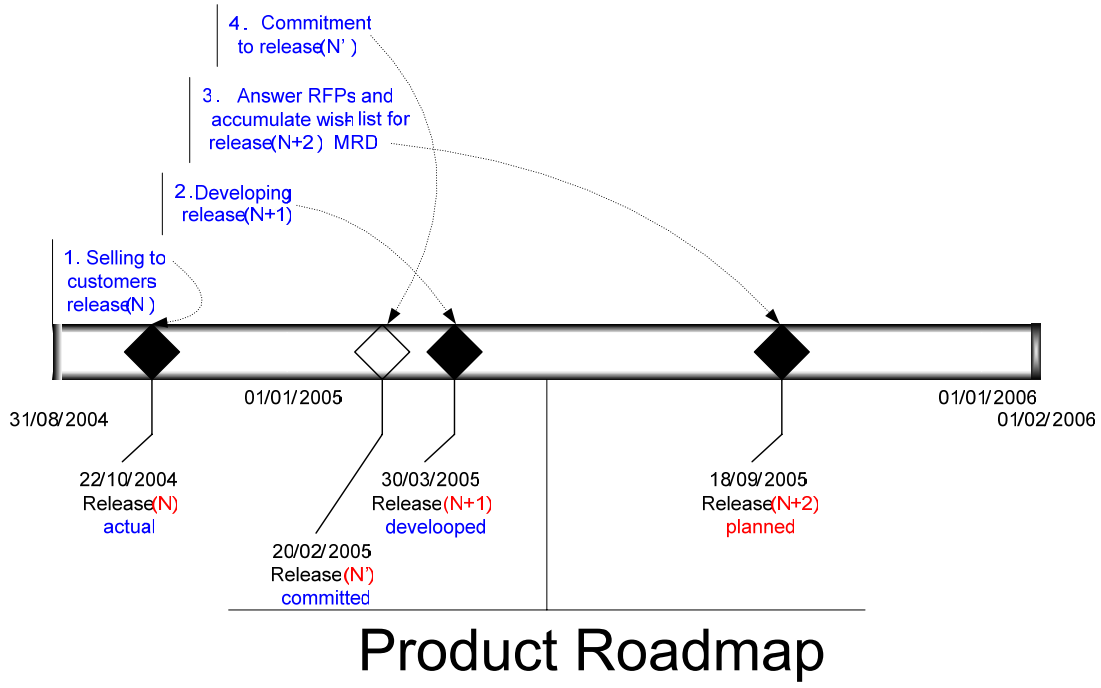


Figure 3: Four RE cycles related to product roadmap

From the product road-map point of view, all the above-mentioned RE cycles (N, N', N+1, N+2) are related to each other. Marketing & Pre-sales forces keep an ongoing effort of replying to requests for proposals (RFPs) to different prospects. Answering an RFP is mandatory for the business in order to stay in the market, even if the bid does not win. During that time, a list of market needs is accumulated into a wish-list which will then be used as input for preparing the MRD for a product release that will provide the required solution, which will hopefully be N+2 or later, but may also require interim releases, to support business constraints. Note that at this stage the MRD for release N+1 is already determined and is being used. The development based on requirements that are within the scope of release N+1 is in process, so any "urgent" new requirements will necessarily casuse an interim, unplanned release (N').
Based on the above description, we can identify four concurrent RE cycles, as illustrated in Figure 3.

## *4. Unified RE Approach*

RE processes, as traditionally described in the literature, are elicitation, analysis, specification, and validation [2]. since requierments tend to change, management becomes an important component as well. In practice, the requirements elicitation process is asynchronous, hence management becomes a crucial issue [5]. Since requirements need to be realized in timely fashion, release planning has been introduced as a substantial process as well.
All in all, the traditional activities of requirements engineering, together with the specific additions need to be integrated into a complete framework to support RE for product evolutionary development. This section discusses the individual components and what is required from an integrated RE approach.

### *4.1 Requirements Elicitation*

Elicitation with respect to future releases is of a different nature compared with elicitation for releases N and N'. Thus, requirements Elicitation should be divided into three different cases:

1. Requirements elicitation regarding release N addresses the specific needs of a customer with respect to the existing product version. It starts with a requirements document prepared by the customer, usually in a form of an RFP, for a first match with the existing product capabilities. Although the customer requirements are presumably specified in the RFP, elicitation proceeds through pre-sales and sales sessions. The purpose of elicitation in this case is not to identify requirements for further evolving the product. Rather, it is to gain an understanding of the customer needs in order to identify the existing product features that meet these needs. A second outcome of this process is unsatisfied requirements, which can be documented and considered for future releases.
2. If substantial gaps are identified when attempting to sell release N, these may lead to requirements for a committed release N'. Requirements elicitation in this case resembles "traditional" elicitation, since it addresses a specific known customer.
3. Elicitation for future releases is the elicitation usually referred to as involving "invented requirements" [10]. This is an on-going process, where requirements are gathered from various sources, such as customer visits, customers feedback on current releases, sales personnel inputs, and to a lesser extent, market surveys [6]. Moreover, unsatisfied requirements when selling the current release (N), as well as the specific requirements elicited for release N' are also considered as requirements sources for future releases. A major challenge here is to avoid requirements duplication and overload [7][13].

### *4.2 Requirements Analysis*

Analysis traditionally deals with elaboration aimed at removing ambiguity, conflict identification and resolution, feasibility analysis, as well as resource and cost estimation. The most important aspects of the analysis in market oriented RE are requirements prioritization [6][11], dependency identification [1][6][3], and resource and cost estimation [1][4][6]. Note that dependency among requirements can either be functional dependency or value dependency, where the customer value of a certain requirement can be fully realized in the presence of another requirement [1]. Regarding our four RE cycles, resource and cost estimations are of importance to all the four. However, RE for future releases emphasizes prioritization and dependecy analysis [1][3][6][11][9][12] , as a basis for release planning. Gap analysis, which is not usually discussed with respect to requirements analysis, is particularly important, both for releases N and N', and as a basis for time / cost estimates for future releases requirements. The analysis, specifically regarding release N and N', can reuse knowledge gained in the past in similar projects.

Analysis of requirements for future releases may also involve variabilty management. Generic requirements are generalizations of specific customer requirements and a result of domain analysis. These specific requirements, in turn, may be expressed as certain variants and options to be included in the product.

### *4.3 Specification & Documentation*

Requirements Specifications is the written (documented) interpretation given to requirments in order to use them as a basis for softeware development. These specifications are subjected to requirements quality criteria, such as clarity, completeness, understandability, verifiability, etc., in order to assure a well defined specifications leading to a quality software product.

Regarding our RE cycles, it is mandatory to have well-defined specifications per each release. As indicated by [6][7], in practice documentation is frequently in the form of informal textual requirements before a complete release specification is produced. The release specifications themselves are commonly written as delta documents per each release, that indicate the incremental part of specifcations with regards to the referenced release, N'→N, N+1→N, N+2→N+1. After several releases it may become impossible to understand the full picture of the current product specifications from the different delta documents. A good requirements management process can sort this problem. The specifications should provide a complete definition of a system at whatever development stage we are interested – N, that exists, N+1 that is currently under development and N+2 that is planned.

### *4.4 Requirements Validation*

Validation of requirements seeks to discover errors in the specifications [2]. Validation for releases N and N' is similar to "traditional" requirements validation, since a specific customer is involved and can validate the requirements. As indicated by [6][7], validation of requirements for future releases is somewhat more difficult due to the absence of real customers. Beta versions are used for validation (of release N+1), but at a rather advanced phase in the release life-cycle. Here traceability to the origin of a requirement may play an important role. Some of the requirements for future releases are results of gaps identified when selling release N (or past releases). These requirements were originally raised by real customers, who probbably validated them in the past, and track can be kept both for the test plan and for results found. As well, it may be possible that these customers, intending to upgrade their current product release, will be interested in validating their requirements to be included in future releases

### *4.5 Release Planning*

Release planning has started to attract attention recently, indicated by various theory-based approaches [4][12] as well as practice reports [3][6][11]. Nevertheless, all the reported approaches rely on common building blocks, which are requirements prioritization, roadmapping, dependency analysis, and resource allocation in view of time/cost estimates. Strategic market planning is sometimes applied, but usually rather as external guidelines than as an integral part of this activity [6].

Release planning naturally relates to requirements for future releases only and not for releases N and N', although awareness of future release plans may affect these requirements, knowing that certain solution are expected to be available in the near future, i.e., mamaging product roadmap may eliminate N' efforts.

### *4.6 Requirements Management*

In general, requirements managements is about centerally storing, maintaining, and using the products of all the above discussed requirements processes.

In the situation described, requirements management is the centeral control, that integrates all the products, provides the needed connectivity and enables the efficient use of knowledge gained in the various processes, keeping traceability and configuration control.

Requirements management should employ a centeral repository designed to meet the specific challenges of the various processes in the concurrent RE cycles taking place [8]. It should store requirements of the following categories: (a) satisfied requirements (in support of the current release sales-oriented RE process), (b) requirements included in customized releases (N'), that can be considered for inclusion in future releases, (c) unsatisfied requirements already included in future release plans, (d) unsatisfied requirements yet to be planned, and (e) requirements that were rejected in the past, together with the reasons for rejecting them (in case they are raised again for reconsideration).

Different kinds of requirements groupings may be needed. Two important ones are (a) functional grouping, as a basis for possible generalization, and (b) dependency-related grouping.
Traceability should, naturally, be kept, e.g., from generic requirements to specific requirements and to the specific customer (if exists).

### *4.7 Unified RE Discussion*

Table 1 summarizes the above discussion of the RE processes in view of the the four concurrent RE cycles, i.e. N, N', N+1, N+2 ahead. The colomns in the table represent the product releases, where the raws represent the traditional domains of RE processes, i.e., requirements elicitation, specification analysis, release planning and requirements management as discussed above. Note that release N+1 is considered as being currently developped. Hence, elicitation is never aimed at it, but rather at release N+2 and ahead, to be planned afterwards.

Table 1: RE processes in the concurrent RE cycles

| **Release Process** | **N** | **N'** | **N+1** | **N+2 ahead** |
|---|---|---|---|---|
| **Requirements Elicitation** | Requirements to be accomodated within current release, gap detection | Elaboration of gap analysis | | Multiple sources, both specific and non-specific customer |
| **Requirements Analysis** | Gap analysis | Gap analysis, time / cost estimation, | Dependency detection, time / cost estimation, generalization and variability management | Prioritization, dependency detection, time / cost estimation, generalization and variability management |
| **Requirements Specification** | | Delta documents | Delta documents, full system spec | Delta documents |
| **Validation** | Along with the customer | Along with the customer | Beta version; with specific customers | |
| **Release planning** | | | Based on resource allocation, dependencies, prioritization | Based on resource allocation, dependencies, prioritization, roadmapping |
| **Requirements Management** | Current satisfied requirements, traceable unsatisfied requirements | Traceable commitements and current requirements | Traceability of requirements and plans, configuration management | Status-managed requirements, traceable to origin and to planned releases, configuration management |

## *5. Discussion: Challenges of a Unified RE Approach*

RE in software product evolution is crucial. In practical situations, even though at the initiation of the product the requirements are frquently well handled, market pressures as well as schedule and resource limitations may lead to deterioration in the RE processes during the evolution of the software product. This may be viewed as a paradox, that during a software product evolution, the more the RE process is required the less it is done.
Considering the above discussion, it seems that a key issue in effectively utilizing the information gathered and created through the four concurrent RE cycles is to be able to capture their inter-relations. This is particularly important with respect to the N and N' cycles, which are usually regarded as separate activities, not related to roadmapping and release planning. In many cases these two cycles are performed by different organizational units (sales vs. marketting) or even outsourced. Nevertheless, the information these cycles can provide is valuable and should be available for the entire RE effort performed in the organization.
We believe that this should be addressed by appropriate requirements management processes and infrastructure. Specific RE activities discussed above, such as analysis, verification, and release planning, should be designed when the entire scope of information and traceability is taken into account.
At the same time, each of the four RE cycles requires different capabilities, tools, and views of the requirements database. We therefore identify the following challenges, posed mainly on the requirements management process and infrastructure, to support a unified RE approach.
*Challenge 1: unification*. Defining a requirements management process that intersects between the products of the different RE processes in the different cycles. Such process should enable the utilization of the

massive amount of information gathered and produced while avoiding overload. It should be supported by a requirements management infrastructure, incorporating a requirements database that allows full traceability and configuration management of both requirements and releases.
*Challenge 2: distinction*. Defining distinct views of the requirements database and support functionality required for the different activities in the four cycles. The challenge is to separate and distill the specific RE activities and deliverables per each product release during its evolution. Per each cycle we need the specific RE scenario that aggregates all and only the RE activites required for that specific release type. It seems that for future releases in the product roadmap (N+2 ahead) the emphasis is on elicitation, analysis, specification, and planning. However, in current release N, RE should produce its benefits for assisting in the sales activities, consuming minmal RE effort, and producing information for future utilization. As for releases in development (N', N+1), the main RE effort will be in requirements management, specifically configuration control, and in traceability-enabled verification.

## *6. Conclusion*

The motivation of this paper was to analyse the RE activities within software product evolution. This led us to the concept of Unified RE Approach that by different RE scenarios we can handle better the requirements during the evolution of a software product releases.
From the above it is clear that in practice, requirements are gathered, anlalyzed, specified and validated in a continuous process. Product releases are dynamically planned, supporting evolving market/customers needs. In a unified RE approach each scenario per every release type can be viewed as a temporal "snapshot" or view of the product requirements repository. This dynamic situation in turn requires complicated mechanisms for requirements traceability in order to keep track on the actual content of the various products releases.
In order to support such dynamic RE approach , RE scenarios need to be defined and evaluated, covering all the above-mentioned RE attributes and activities. The unified RE approach should be based on already existing concepts and methods, while utilizing the inter-relations between the different processes and deliverables for achieving a better support to each distinct activity.

## *7. References*


[1] Akker J.M. van den, Brinkkemper S., Diepen G., Versendaal J., 2005, Determination of the Next Release of a Software Product: an Approach using Integer Linear Programming, *Proceedings of the CAiSE'05 Forum*, p. 119-124, Porto, Portugal.

[2] Bray I. K., 2002, *An Introduction to Requirements Engineering*, Addison-Wesley.

[3] Carlshamre P. and Regnell B., 2000, Requirements Lifecycle Management and Release Planning in Market-Driven Requirements Engineering Processes, *Proceedings of the 11th International Workshop on Database and Expert Systems Applications (DEXA'00)* p. 961-966, Greenwich, London, U.K.

[4] Carlshamre P., 2002, Release Planning in Market-Driven Software Product Development: Provoking an Understanding, *Requirements Engineering* 7, p. 139-151.

[5] Dag J. N. och, Gervasi V., Brinkkemper S., 2005, A Linguistic-Engineering Approach to Large-Scale Requirements Management, *IEEE Software* 22(1), p. 32-39.

[6] Dahlstedt A.G., Karlsson L., Persson A., Dag J. N. och, Regnell B., 2003, Market-Driven Requirements Engineering Processes for Software Products – a Report on Current Practices, *International Workshop on COTS and Product Software: Why Requirements Are So Important (RECOTS)*, held in conjunction with the 11th IEEE International Requirements Engineering Conference, September 2003, Monterey USA

[7] Deifel B., 1999, A Process Model for Requirements Engineering of CCOTS, *10th International Workshop on Database & Expert Systems Applications,* p.316-321.

[8] Grynberg A., Goldin L., 2004, Product Management in Telecom Industry – Using Requirements Management Process, *Proceedings of SwSTE'04*, Tel Aviv, Israel.

[9] Karlsson L., Dahlstedt A., Dag J. N. och, Regnell B., Persson A., 2002, Challenges in Market-Driven Requirements Engineering – an Industrial Interview Study, *8th International Workshop on Requirements Engineering: Foundations for Software Quality*, Essen, Germany.

[10] Potts C., 1995, Invented Requirements and Imagined Customers: Requirements Engineering for Off-the-Shelf Software, *Proceedings of Second IEEE International Symposium on Requirements Engineering (RE'95)*, p. 128-130, IEEE Computer Society.
[11] Regnell B., Host M., Dag J. N. och, 2001, An Industrial Case Study on Distributed Prioritization in Market-Driven Requirements Engineering for Packaged Software, *Requirements Engineering*, 6, p. 51-62.
[12] Regnell B., Karlsson L., Host M., 2003, An Analytical Model for Requirements Selection Quality Evaluation in Product Software Development, *Proceedings of the 11th IEEE International Requirements Engineering Conference*, IEEE Computer Society.
[13] Sawyer P., 2000, Packaged Software: Challenges for RE, *Proceedings of the 6th International Workshop on Requirements Engineering: Foundation for Software Quality*, Stockholm, Sweden.
[14] Soffer P., Golany B., Dori D. and Wand Y., 2001, Modelling Off-the-Shelf Information Systems Requirements: An Ontological Approach, *Requirements Engineering* 6(3), p. 183-199.
[15] Xu L, Brinkkemper S., 2005, Concepts and Research Framework of Product Software, *Proceedings of PHISE 2005 Workshop*, Porto, Portugal.
[16] Yeh, A.C., 1992, Requirements engineering support technique (REQUEST): Assessment of a market driven requirements management process, *Proceedings of the Second Symposium on Quality Software Development Tools*, p. 211-223, New Orleans, LA, USA.