Preliminary Results of A Systematic Review on Requirements Evolution

Juan Li[1], He Zhang[2], Liming Zhu[2,3], Ross Jeffery[2,3], Qing Wang[1], Mingshu Li[1]
[1]Institute of Software, Chinese Academy of Sciences, China
[2]National ICT Australia, Australia
[3]School of Computer Science and Engineering, University of New South Wales, Australia
{lijuan,wq,mingshu}@itechs.iscas.ac.cn, {He.Zhang, Liming.Zhu, Ross.Jeffery}@nicta.com.au

Abstract—***Background*: Software systems must evolve in order to adapt in a timely fashion to the rapid changes of stakeholder needs, technologies, business environment and society regulations. Numerous studies have shown that cost, schedule or defect density of a software project may escalate as the requirements evolve. Requirements evolution management has become one important topic in requirements engineering research. *Aim*: To depict a holistic state-of-the-art of requirement evolution management. *Method*: We undertook a systematic review on requirements evolution management. *Results*: 125 relevant studies were identified and reviewed. This paper reports the preliminary results from this review: (1) the terminology and definition of requirements evolution; (2) fourteen key activities in requirements evolution management; (3) twenty-eight metrics of requirements evolution for three measurement goals. *Conclusions*: Requirements evolution is a process of continuous change of requirements in a certain direction. Most existing studies focus on how to deal with evolution after it happens. In the future, more research attention on exploring the evolution laws and predicting evolution is encouraged.**

Keywords- requirements evolution, requirements change, management process, measurement, systematic literature review

I. INTRODUCTION

Requirements evolution is inevitable during the whole lifecycle of software projects. Requirements evolution impels systems to be faster, more efficient or more reliable. Lehman's well known Law of Continuing Change observes that a program used in a real-world environment must change or become progressively less useful [1]. Through continuously meeting with user's needs for functional and non-functional features of software systems, well-managed requirements evolution can improve software quality and increase satisfaction.

However, requirements evolution also brings the risk of project failure. An investigation in 4000 European companies shows that how to manage and control requirements evolution is a significant problem in software development [2]. Effective requirement evolution management is one key assurance for the success of software projects [3].

Requirement evolution management involves a series of activities, technologies and stakeholders. Generally, after a evolution request is proposed, a set of activities need to be performed to control the risks, for example, analyzing impact, making decisions, updating requirements documents and tracking problems. However, many challenges remain. How do project managers design or choose appropriate activities for the specific project contexts such as outsourcing? What technologies can be adopted in an activity to decrease risks to the largest extent? In addition, stakeholders (e.g., project managers, requirement engineers, developers, customers and end users) may have conflicting evolution requests due to their different needs. How do we help them achieve a consistent understanding? All of these increase the complexity of requirements evolution management.

Furthermore, the emerging technologies and development paradigms may bring new problems to the traditional requirement evolution management. For instance, in outsourcing development, the outsourcer and the outsourcee need collaborative methods to control requirements evolution proposed by each other. As for COTS-based software, when a component is updated, component re-evaluation and re-selection are the new factors to be considered in evolution impact analysis.

Giving the importance of managing and controlling requirements evolution, we undertook a systematic literature review (SLR, also referred as systematic review). The objective of this review is to depict the landscape of the existing knowledge and experience in managing requirements evolution. Accordingly, we designed a series of research questions, identified and analyzed 125 relevant publications. This paper reports our preliminary findings for the three research questions: (1) What is the definition of requirements evolution? (2) What activities should be performed after the requests for requirements evolution are proposed? (3) How to measure requirements evolution?

In this review, we found "requirements change" and "requirements evolution" were widely misused in the literature. After examining the existing studies and comparing to the relevant concepts such as "evolution" and "software evolution", we propose a definition of "requirements evolution". Requirements evolution is a process of continuous change of requirements in a certain direction. It is composed by discrete events of requirements changes and represents some predictable trends. Most existing studies focus on how to deal with evolution after it happens. For example, many process models were proposed to manage requirements evolution. Recent requirements evolution measurement research focus on predicting evolution using requirements size. Exploring the

evolution laws and predicting evolution to actively evolve requirements need more research attention in the future.

The target readership is three groups who might be interested in the state-of-the-art of requirements evolution management: (1) researchers in software engineering who would like to learn about requirements evolution's relationship with software evolution to address important research gaps; (2) researchers in requirements engineering in general, who would be interested in quickly finding relevant studies on requirements evolution management; (3) practitioners who will be interested in knowing what methods are effective in managing requirements evolution.

The paper is organized as follows: Section II presents a brief overview of requirements evolution and software evolution. Section III describes the research method used for this review. Section IV summarizes the relevant studies. Section V analyzes the results for answering research questions. Section VI discusses the benefits, limitations and strength of evidence. Section VII concludes this study with suggestions for further research on requirements evolution.

II. BACKGROUND

A. *Requirements evolution and software evolution*

Software evolution is a process of continuously updating a system to adapt to changing environment and meet users' requirements. It is an innovative activity focusing on adaptation and migration [4]. Although evolution of the software systems is mostly studied at the level of code and design (with a focus on code reengineering/migration, architectural evolution and software refactoring), requirements evolution has also been regarded as one other important area in software evolution [5].

However, there is so far no common understanding about what requirements evolution is. Software evolution happens after a product is delivered. Similarly, in which exact phase(s) does requirements evolution happen? Is there any difference between requirements changes and requirements evolution? What is the contribution of requirements evolution research to software evolution? In this review, we were motivated to investigate the understanding of requirements evolution in the existing literature. With no doubt, a clear, accurate and explicit definition of requirements evolution can help us identify the research scope and the key problems in this area.

B. *Requirements management*

One objective of requirements management is to manage changing requirements. Changes can be monitored and tracked during the software lifecycle. The primary activities in requirements management include controlling requirements change, controlling versions of requirements and documents, tracking requirements status, tracing requirements to other work product. Fig. 1 shows the details in each of these activities in requirements management.

As shown in Fig. 1, the change control aims to control risks in a software project through impact analysis. After making decisions, related documents and models should be changed correspondingly and kept in consistency. Measuring requirements volatility helps organizations describe the change trend and predict change impact on schedule, cost and software quality. Our research questions in this review focus on these change control activities that are relevant to requirements evolution.

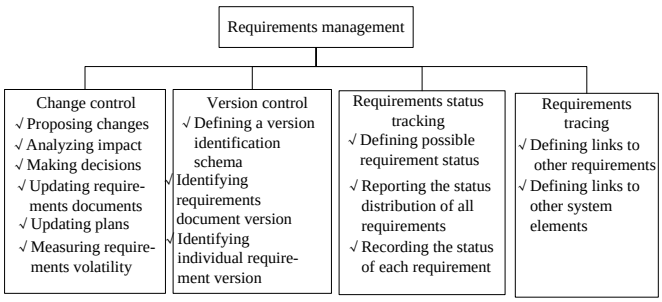


Figure 1. Activities in requirements management [6]

C. *Related work*

Systematic literature review provides a rigorous and repetitive method to aggregate relevant studies on a specific research topic. It has become an important research methodology in the software engineering research community since its introduction. A number of SLRs have been conducted and reported with research topics such as cost estimation [7] and knowledge management [8].

In requirements engineering, the reported SLRs focus on classification of requirements error [9], empirical studies on elicitation techniques [10], generation of textual requirements specifications from software engineering models [11], technology transfer decision support [12] and traceability in requirements engineering [13]. However, to the authors' knowledge, no SLR on requirements evolution research was reported. Requirements evolution, as one essential component in software evolution research and requirements engineering research, is gaining more attention. One objective of this work is to bridge this gap.

III. RESEARCH METHOD

A systematic literature review evaluates and interprets all available research relevant to a particular research question or topic area. It aims to present an evaluation of the literature to relative to research topic using a rigorous and auditable methodology. We have followed the guidelines proposed by Kitchenham, B.A. et al. [14]. In this section, we detail the methods adopted in this SLR.

A. *Protocol development*

At the beginning, the first three authors formed a review panel and performed the major activities in this SLR. We developed a protocol for the systematic review by following the guideline and procedure in [14]. We specified the research questions, search strategy, inclusion/exclusion criteria, data extraction, and methods of synthesis in this protocol. The objective of this review is to answer the research questions shown in Table I.

TABLE I. RESEARCH QUESTIONS AND MOTIVATIONS

Research question	Motivation
1, What is the definition of requirements evolution?	What is the common understanding of requirements evolution in literature? Is there any difference between requirements changes and evolution?
2, What activities are included in the requirements evolution management process?	What activities should be performed after requirements evolution request is proposed?
3, How to measure requirements evolution?	What problems can be solved through measurement and what measures are used in literature?

B. Search strategy

There are many international conferences and journals that have published the relevant work. We met some difficulties during the search query design.

- Which synonyms of "evolution" should be included in search query? "Evolution" has many synonyms, such as change, volatility and uncertainty. Its antonyms e.g., stability may also appear in some papers. It is difficult to exhaustively determine all these words for search query.
- "Requirements evolution management" or "requirements management"? Requirements management covers not only evolution management but also requirements tracing and others. But in terms of our trials, the inclusion of "requirements management" in search query incurs the retrieval of many irrelevant studies, which significantly increase the burden on study selection.
- "Requirements" or "software requirements"? If we choose "requirements" as the key word, many papers on hardware or network requirements will be found. But if we use "software requirements" instead, some relevant publications not including "software" in title or abstract might be missing.

In order to solve above problems and find a majority of relevant literature, we adopted a scientific literature search strategy, which effectively combines manual search and automated search. The details of this scientific search strategy were described in a paper [15].

As the procedure shown in Fig. 2, we initiated our literature search by manually screening the domain-specific publication channels (journals and conferences) of requirements engineering, and established Quasi-Gold Standard (QGS) as illustrative samples [15]. Then the search query was derived based on the common terminology observed in this standard.

(1) Manual search

In September 2009, we manually scanned all papers published through Requirements Engineering Journal (REJ) from 1994 to 2009 and proceedings of International Requirements Engineering Conference (RE) from 1994 to 2008 (Proceedings of RE'09 were not included in IEEE Xplore then). We first examined the "title" to eliminate any papers clearly not related to our research focus; then exclude papers that appeared clearly not relevant in terms of their "abstract" and "keywords". The other irrelevant papers were excluded by carefully reading the full-text. Finally, we selected 8 studies from 14 volumes of REJ and 19 papers from 11 conference proceedings of RE.

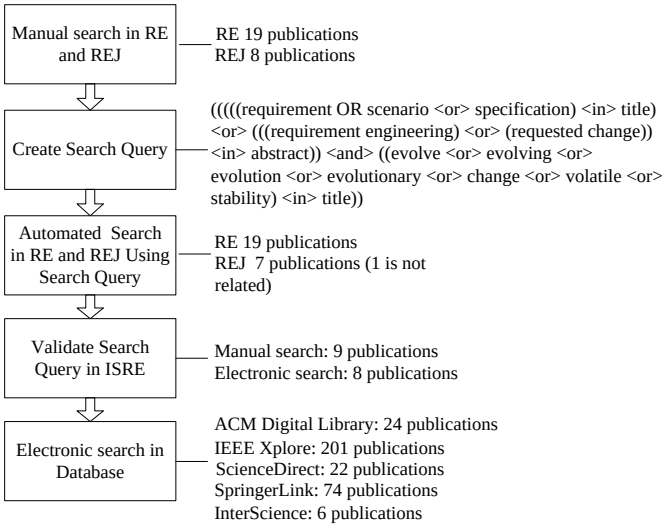


Figure 2. Search strategy

After analyzing the titles and abstractions of the standard (27 study samples), we devised the search string:

(((((requirement OR scenario <or> specification) <in> title) <or> (((requirement engineering) <or> (requested change)) <in> abstract)) <and> ((evolve <or> evolving <or> evolution <or> evolutionary <or> change <or> volatile <or> stability) <in> title)))

This string was validated by automatically searching REJ and RE in terms of its sensitivity and precision.

Sensitivity= number of relevant publications retrieved / total number of relevant studies

Precision=number of relevant publications retrieved / number of articles retrieved

Here, "total number of relevant studies" means relevant studies searched manually. "Number of article retrieved" means number of articles retrieved automatically.

The results are shown in Table II. As suggested in [15], the acceptance threshold is between 75% - 85%. The values of these two performance indicators are acceptable that means the search query is effective enough to find the majority publications in RE and REJ.

Then we further validated it by searching another domain-specific venue - International Symposium on Requirements Engineering (ISRE). The manual search in ISRE proceedings (1994 to 2001, it was merged into RE in 2002) found 9 papers. 8 of them were found in the automated search in ISRE (through IEEE with the above search query). This result further evidenced the effectiveness of the proposed search query.

TABLE II. VALIDATION OF SEARCH STRING

Conference or Journal	Sensitivity	Precision
RE	100%	100%
REJ	75%	86%
ISRE	89%	100%

(2) Automated search

The search query was input to the five publishers' search engines for automated search: ACM Digital Library, IEEE Xplore, SicenceDirect, SpringerLink, Wiley InterScience. The search results are shown in Fig. 3.

C. Study selection

Studies were eligible for inclusion in the review if they presented research on requirements evolution. The source includes research papers and book chapters. A study was excluded

(1) if it is clearly not on software requirements evolution;

(2) if it can not meet the minimum quality assessment in section D;

(3) if the work is a duplicate publication;

(4) if it is a short paper with 3 pages or less.

After the exclusion of the irrelevant studies by screening title-abstract-keywords and full-text (cf. Manual Search), we selected 122 publications. Among these 122 publications, we kept one Ph.D proposal on requirements model evolution [16] published in ICSE'04 due to its representative research question. Furthermore, we included its continuing work published in another conference [17]. In addition, we directly included two papers on requirements measurement in this SLR ([18], [19]) in terms of our previous domain knowledge rather than search results. In total, the number of primary studies is 125.

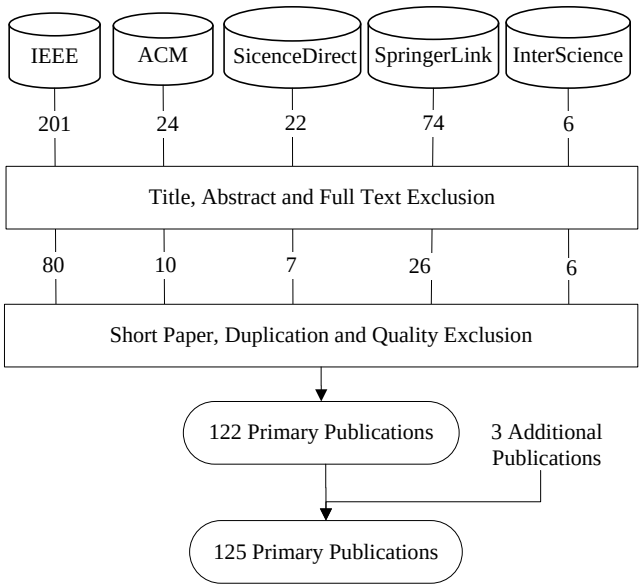


Figure 3. Multi-step filtering of studies

D. Quality assessment

We assessed the quality of the selected publications based on the criteria in Table III. Due to the page limit, the quality results are not included in this paper.

TABLE III. QUALITY ASSESSMENT

Number	Quality assessment criteria
1	Did the study clearly state the aims or research questions?
2	Did the study review the related work for the problem?
3	Does the study validate the method?
4	Did the study recommend the future continuous research?
5	Is there a clear statement of findings?
6	Is the study of value for research or practice?

E. Data extraction and synthesis

The studies were categorized into 5 groups and one study may fall into more than one category.

- Evolution management process: evolution management process, method, framework, modeling, requirements tracing.
- Impact analysis: evolution analysis, evolution risk, effort or cost estimation.
- Model evolution: scenario evolution, requirement specification or model evolution, design model evolution.
- Requirements evolution discovering: requirements stability assessment, requirements volatility discovering, requirements change predicting.
- Requirements evolution measurement: change metrics, quantitative analysis of change.

According to these categories, we designed the data extraction form. This form is composed of 3 parts: citation information, common attributes and category specific attributes. Citation information collects publication data such as authors, title and publication venue. Common attributes are the common data items for all categories of studies, such as requirements evolution definition, cause and types. Category specific attributes gather distinct information from different categories, such as activities in evolution management process.

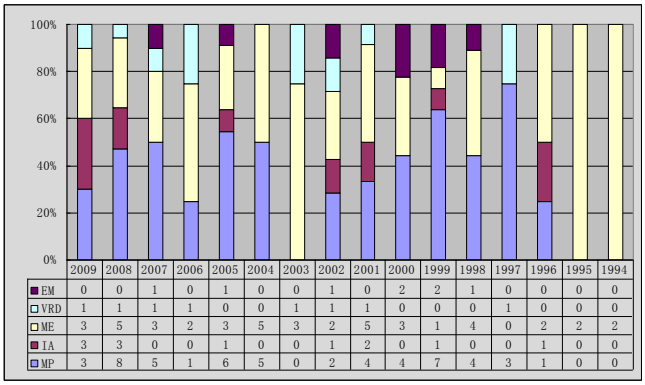
The data extraction process is composed of four phases: (1) the review panel extracted data from 65 publications; (2) authors conducted pilot data analysis to validate the data extraction form; (3) four postgraduate students extracted data from remaining 60 papers; (4) a review panel checked data extracted by students. During the pilot analysis, we found examples should be provided for some questions to help data extractors understand the questions and the form missed some necessary data items. Then we refined the data extraction form based on these findings. In the third phase, every paper has one data extractor and one checker to assure the quality of data. If the students had problems in extraction, they escalate the problems to the review panel to seek a solution. In the last

phase, the review panel checked all the data submitted by students. The extracted data were stored in our online systematic review system (http://systematicreviews.org), which allows sharing, accessing and managing data in a geographically distributed research circumstance.

IV. RESULT

We classified the studies into five categories. The number of publications and their percentage in different category are shown in Fig. 4.

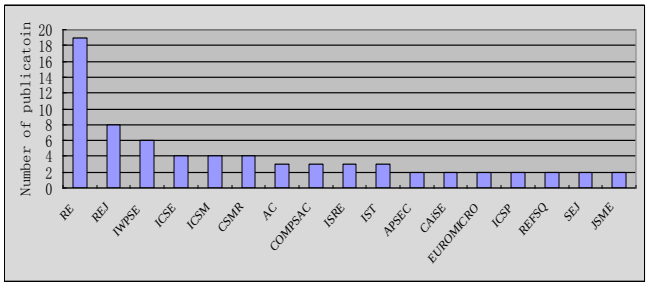
- **Management Process (MP)**: 41% of the studies are related to evolution management process. For example, some studies proposed evolution management process models for outsourcing project, eXtreme Programming (XP) and product line engineering; some focused on specific activities such as requirements tracing and cause analysis.
- **Model Evolution (ME)**: 36% of the studies fall into this category. The research topics include evolution of formal requirements specification, scenario evolution, inconsistency management of requirements model, architecture evolution driven by requirements change.
- **Impact Analysis (IA)**: 10% of the studies are in this category. Existing studies on impact analysis focus on identifying what artifact need to be modified to accomplish a change and proposing methods to estimate the potential consequences of a change.
- **Evolution Measurement (EM)**: This category includes 6% of the studies. Existing researches shed light on evolution impact on project performances, evolution characteristics (e.g., cause and type), and evolution prediction based on requirements size.
- **Volatile Requirements Discovering (VRD)**: 7% of the studies are in this category. Discovering volatile requirements at the beginning of a project can decrease the probability of requirements evolution. Some studies use scenarios to assess requirements stability, or crosscutting concerns to discover volatile and aspectual requirements.



	2009	2008	2007	2006	2005	2004	2003	2002	2001	2000	1999	1998	1997	1996	1995	1994
EM	0	0	1	0	1	0	0	1	0	2	2	1	0	0	0	0
VRD	1	1	1	1	0	0	1	1	1	0	0	0	1	0	0	0
ME	3	5	3	2	3	5	3	2	5	3	1	4	0	2	2	2
IA	3	3	0	0	1	0	0	1	2	0	1	0	0	1	0	0
MP	3	8	5	1	6	5	0	2	4	4	7	4	3	1	0	0

Figure 4. Distribution of publication in different category every year

Among the 125 publications, 79% of the studies were published in conferences, 18% of the studies in journals and others as workshop papers. Regarding the publication venues, 51 conferences, 4 workshops and 11 journals published papers on requirements evolution management. Fig. 5 shows those venues with two or more relevant studies published.



Note: Requirements Engineering Journal (REJ), International Requirements Engineering Conference (RE), International Workshop on Principles of Software Evolution (IWPSE), International Conference on Software Engineering (ICSE), International Conference on Software Maintenance (ICSM), European Conference on Software Maintenance and Reengineering (CSMR), Aerospace Conference (AC), Computer Software and Applications Conference (COMPSAC), International Symposium on Requirements Engineering (ISRE), Information Science and Technology (IST), Asia-Pacific Software Engineering Conference (APSEC), International Conference on Advanced Information Systems (CAiSE), EUROMICRO Conference, International Conference on Software Process (ICSP), Requirements Engineering: Foundation for Software Quality (REFSQ), Software Engineering Journal (SEJ), Journal of Software Maintenance and Evolution: Research and Practice (JSME)

Figure 5. Major conferences and journals

V. DISCUSSION

A. *Definition of requirements evolution (Q1)*

There are many synonyms of "evolution" used in the existing relevant literature, such as change, creep and stability. These words express different characteristics of the evolution" phenomenon and can easily cause confusion in the understanding of requirements evolution.

We counted such key words used in the titles of studies to find the most frequent words in the primary studies. Note that we avoided counting the synonyms used in the full-text of paper because authors usually try to adopt different synonyms in text to improve the paper's readability. There are more than 6 synonyms of "evolution" adopted in titles. They are change, evolution, creep, stability, volatility and uncertainty. Among them, "evolution" and "change" are most used: 48% of the publications used "evolution" (including evolve, evolving and etc.) in the titles and 46% used "change" instead (including changing). As shown in Fig. 6, "evolution" was used more frequently in the recent decade (over 50% in most years).

Requirements change means the addition, deletion and modification of requirements [20-23]. But surprisingly, we found no explicit definition of requirements evolution. Consulting the Webster Dictionary, evolution is "a process of change in a certain direction, a process of continuous change from a lower, simpler, or worse to a higher, more complex, or better state" [24]. In software engineering, "evolution" is frequently used to describe continuous change of one artifact or

product. For example, Lehman pointed out continuing change of software is one important software evolution law. According to this understanding, requirements evolution is a process of continuous change of requirements in a certain direction.

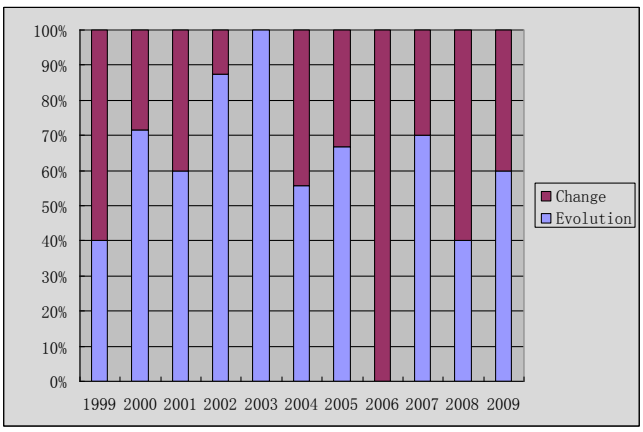


Figure 6. Evolution and change as keywords used in titles

Although the existing studies used different words, the main research topic addressed concentrates on how to deal with frequent and continuous requirements changes. For example, to measure the impact of a large number of requirements changes on project schedule [25], to analyze the impact of frequent changes on specification [26]. Accordingly, we suggest "requirements evolution" as the most appropriate term for denoting this research topic. Note that, in this SLR, a paper with "evolution" in its title is not a hard criterion for study selection.

Among the studies, there is a different understanding of the phase in which requirements evolution happens. In most cases, requirements evolution happens not only after the system has been built, but also during the process of implementing the system [27]. However, Ernst, N.A. et al. state requirements evolution occurs after a system is in operation [28] because requirements evolution is a sub-area of software evolution and the latter happens in the post-release phase. In our opinion, the phase in which the evolution happens depends on whether the artifact to evolve, in its completed form, existed before a phase. For example, software evolution happens after software, as an artifact, is released. In this sense, requirements evolution happens after requirements definition is completed. Thus, requirements evolution can happen during the entire lifecycle excluding the definition stage. Furthermore, requirements evolution not only determines changes in the requirements themselves (both representations in requirements specifications and their semantics), but also includes the change propagation process from requirements to software design, construction, verification and maintenance [29].

Based on the above observations, statements and analysis, we propose the definition of requirements evolution: Requirements evolution is a process of continuous change of requirements in a certain direction. It can happen during the entire software lifecycle excluding the definition phase. The propagation of requirements evolution spans from requirements to maintenance of a software system.

Summary: The existing relevant studies have no explicit definition of requirements evolution. Requirements changes and requirements evolution were misused in the literature. Based on the analysis of the research emphasis of the relevant studies and the understanding of "evolution" and its application in software engineering, a definition of requirements evolution is proposed.

B. Requirements evolution management activities (Q2)

When requests for requirements evolution are proposed, a series of activities need to be performed to deal with the evolution. We extracted fourteen activities of requirements evolution management from the existing studies and classified them into four phases as shown in Table IV. Note that, due to the page limit only 1 or 2 example references were listed for each activity in this paper.

- **Prepare for requirements evolution**: Requirements evolution is unpredictable. It is important to establish an evolution control strategy in the early stages of a software project to secure the quality of requirements. In this phase, the key activities are verifying requirements, establishing the requirements baseline, setting up a channel to control change, tracing requirements and planning for change.
- **Impact analysis and decision making**: Once requirements evolution requests are proposed, the decision has to be made on whether or not to accept the evolution request and further how to implement the evolution. Impact analysis is the input to decision making. It includes cause analysis, impact on project performance analysis, artifact impacted identification, and so on. Moreover, achieving consistent understanding about the evolution and identifying change priorities are also important to decision making.
- **Evolution implementation and verification**: Requirements evolution may propagate from requirements level to code level. Its implementation means modifying all the artifacts impacted such as requirements specification, architecture, code and test case, etc. In addition, the evolution correctness needs to be verified through software technologies, e.g., testing and checking.
- **Evolution tracking**: Evolution tracking records evolution knowledge and experiences such as the defects in artifacts caused by evolution and the issues occurring in management process.

For a certain software project, not all the activities have to be executed to manage its requirements evolution. The project may focus on several activities according to its specific context. In Table IV, we noted those activities addressed in different project types. For example, in outsourcing development, the requirements of a new product are provided by an external client. In order to minimize the risks of budget overruns, the client may insist on a fixed price agreement. The requirements document often acts as the contract. When requirements evolution happens, it is important but also challenging to evolve the agreement between project team and client through

negotiation. In addition, a requirements baseline and plan for evolution are effective approaches to managing evolution [27]. As for eXtreme Programming (XP), the strongest undercurrent of XP is embracing change. XP emphasizes requirements verification and validation through onsite-customer, testing and continuous integration, plan for change through simple design and refactoring, correct change implementation through collective ownership [30][31].

Summary: fourteen key activities in requirements evolution management were gathered from the existing relevant studies. The example relevant studies are attached to each activity, which is addressed in some specific project contexts such as outsourcing and XP. Most existing evolution management activities address how to deal with requirements evolution after it happens.

TABLE IV. EVOLUTION MANAGEMENT ACTIVITIES

Phase	Activity	Description	Reference
Prepare for requirements evolution	Requirements verification *^	Assure the requirements quality such as accuracy, correctness, consistency etc.	[27],[32]
	Baseline requirements ^	Establish the basis for the next project phase	[27],[33]
	Establish channel to control evolution ^	Establish the evolution management strategy and process include people, technology and tool	[27], [34]
	Requirements tracing +	Establish traceability between requirements and related artifacts	[35], [36]
	Plan for evolution *^#	Identify adaptable requirements, increase adaptability of design, etc.	[30], [37]
Impact analysis and decision making	Evolution type identification	Identify evolution type and assess the importance of evolution	[32],[38]
	Cause analysis #	Identify the importance and impact extent through analyzing causes	[39],[22]
	Evolution risk analysis +	Change impact on functionality, quality (e.g. performance, safety, reliability, usability),cost, customer and other external stakeholders	[31],[40]
	Negotiation ^#	Conflicts between stakeholders identification and trade-off analysis	[27],[22]
	Evolution prioritization #	Prioritize evolution to make decision which should be implemented first under the constraint of resources	[22],[41]
	Decision making	Decide which artifact should be modified and how to implement the evolution	[41],[42]
Evolution implementation	Modify affected artifact and verification *^	Modify requirements model and other affected artifacts. Test the artifacts after modification	[43]
Evolution tracking	Issue tracking ^	Record issues and defects to assess the evolution process	[27]
	Evolution measurement	Analyze the problems in evolution management process	[18]

Legend: XP(*),Outsourcing project(^), Product line(#), Component based development(+)

C. Rrequirements evolution measuremnt (Q3)

Requirements evolution happens frequently and is difficult to predict in the lifecycle of software. Volatile requirements can cause cost and schedule overrun, further failure in delivering project objectives. Requirements evolution measurement can provide guidance to the requirements management activities by quantifying and predicting changes to requirements. By measuring evolution, project managers will be aware of the scale of evolution and its possible impacts, and take appropriate and instant actions to mitigate project risks.

Most existing researches focus on analyzing evolution impact on project performance and evolution characteristics (e.g., cause and type). Recent requirements evolution measurement research strived to predict evolution from requirements size. We list the major metrics used in the existing studies in Table V. Stark et al [44] collected data from 20 software releases to understand the source, magnitude, and effects of changing requirements on software maintenance. They found requirements volatility is the major reason for the schedule and cost overrun. Loconsole A. et al. collected all use cases from a medium-sized software project and found size of use case model is a good indicator of requirements volatility [18].

Most studies are industrial empirical studies and performed in a single organization. They reported the conclusion but did not introduce the data collection and clean process, measurement instruments, validity evaluation analysis as shown in Table VI. This makes these empirical studies hard to be replicated or referred, and limits the validity of the conclusions.

Summary: Most existing studies focus on analyzing evolution characteristics or impact on project which can act as the basis for evolution prediction. However, most studies neither explicitly introduced the metrics design method and data elicitation method, nor discussed the threat to validity. This weakens the validity of the conclusions. Recent studies focus on evolution prediction from requirements size.

TABLE V. REQUIREMENTS EVOLUTION MEASURES

Goal	Metrics	Reference
Understand the sources, frequency, and types of evolution	Requirements volatility, Requirements change by type, Requirements changes by month, Requirements change by source, Planned and actual effort-days for each requirement, Number of requirements changes, Requirements maturity index, Requirements stability index, Historical requirements maturity, Type of requirement and detailed need	[44],[45],[25],[21], [46]
Analyze the effort, cost, schedule, and quality impact of requirements evolution	Schedule, cost and quality impact of changes, Complexity of the class definition, Interaction of classes, Planned and actual effort-days for each requirement, Planned and actual number of calendar days for a version, Requirements changes made to the version after plan approval, Quality variance, Change cost, Budget reduction, Requirements dependency; Change density, Requirements addition, modification and removal, Error rate and fix cost, Acceptance rate, Timescale variance and budget variance	[44], [21], [47]
Investigate the correlation between the use case size and requirements evolution	Size of use case model, size of change to use case model	[18], [19]

TABLE VI. CASE STUDY

Publication Year	Reference	Data source	Empirical study quality
1998	[44]	20 software releases	No data collection process, no validity evaluation
1999	[46]	44 software releases	No data collection process, no validity evaluation
1999	[21]	No data	
2000	[25]	22 software releases	No data collection process, no validity evaluation
2002	[45]	22 software releases	No data collection process, no validity evaluation
2005	[18]	39 use cases	Introduce the data collection process and analyze validity
2007	[19]	21 use cases	Introduce the data collection process and analyze validity

VI. THREATS TO VALIDITY

Limitation of query: we experienced two major difficulties when designing the search query. First, there exist too many synonyms of "evolution", more than 6 words describing the changing phenomenon of requirements, but we could not use all of them to construct the query due to the limited search capability of the digital libraries. The second is a more appropriate subject matter to search, "requirements evolution management" or "requirements management", "requirements" or "software requirements". Requirements management covers evolution management and some papers on requirements management also resolve the requirements tracing problems which is out of the scope of our review (cf. Section III). To solve these problems, we designed the query based on the publications from the domain-specific journal and conferences (i.e. the QGS) in this field. Using this query we found all the majority of publications in the domain-specific journal and conferences. This strategy assured, in a scientific manner, our review results reflect the majority of the relevant studies in this field.

Validity of evidence: The studies collected from the literature was identified through a search of literature databases covering most relevant journals, proceedings and other literature in this area. Data extraction forms were utilized to consistently extract the desired information from each of the selected publications. The information extracted was validated through comparison of independent analysis results within the review panel. The inclusion and exclusion criteria were well defined prior to the literature search and followed to include the relevant studies only. Four research students extracted evidences from 60 publications. They possess the basic knowledge of requirements engineering. Before starting extraction, they received the training on SLR methodology and guidelines, particularly on data extraction. Every publication has one extractor and one checker. To assure the quality of data extracted by students, the review panel read the 60 papers and examined all the data submitted by students.

VII. CONCLUSIONS AND FUTURE WORK

This paper reports the state-of–the-art of requirements evolution management and measurement derived from a systematic review. We identified and gathered publications concerning requirements evolution management from 1994 up to 2009. We proposed a terminology and definition of requirements evolution, enumerated fourteen often-adopted key activities in requirements evolution management with their specific project context, as well as twenty-eight metrics of requirements evolution for three measurement goals.

Requirements evolution is a process of continuous change of requirements in a certain direction. However, most existing researches focus on how to deal with evolution after it happens. Requirements evolution still remains a phenomenon little understood from both evolution laws and prediction perspectives. Exploring the evolution laws and predicting evolution to actively evolve requirements need more attention in the future.

In addition to the preliminary results reported in this paper, other findings on more specific research questions, such as the evolution cause and risk, evolution impact analysis, evolution of models, will be reflected in our future report. It will contribute to the development of a body of knowledge and evidence of requirements evolution management.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China under grant No. 60803023, No. 61073044, National Science and Technology Major Project 2010ZX01036-001-002, 2010ZX01037-001-002. This work is also supported by National ICT Australia which is funded through the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council and is supported by the Commonwealth of Australia under the Australia-China Science and Research Fund. We thank Lin Shi, Peng Wan, Qiang Cui, Junjie Wang for participating in data extraction.

REFERENCES

[1] M. M. Lehman, Ramil, J. F., Wernick, P. D., Perry, D. E. and Turski, W.M., , "Metrics and laws of software evolution-The nineties view," in In Proc. of the Fourth Intl. Software Metrics Symposium, Albuquerque, NM, November 1997.

[2] I. Sommerville and P. Sawyer, Requirements Engineering a Good Practice Guide: Wiley, 2000.

[3] C. J. Damian D., "An Empirical Study of the Complex Relationships between Requirements Engineering Processes and Other Processes that Lead to Payoffs in Productivity, Quality, and Risk Management," IEEE Transactions on Software Engineering, vol. 32(7), pp. 433-453, 2006.

[4] M. W. Godfrey and D. M. German, "The past, present, and future of software evolution," in Frontiers of Software Maintenance, FoSM 2008. , 2008.

[5] C. Rolland, C. Salinesi, and A. Etien, "Eliciting gaps in requirements change," Requir. Eng., vol. 9, pp. 1-15, 2004.

[6] K. E. Weigers, Software Requirements: Microsoft Press, 2003.

[7] M. Jorgensen and M. Shepperd, " A Systematic Review of Software Development Cost Estimation Studies," IEEE Transactions on Software Engineering, vol. 33, pp. 33 - 53, 2007.

[8] F. Bjornson and T. Dingsoyr., "Knowledge management in software engineering: A systematic review of studied concepts, findings and research methods used," Information and Software Technology, vol. Vol. 50, No. 11, pp. 1055-1068., 2008.

[9] G. S. Walia and J. C. Carver, "A systematic literature review to identify and classify software requirement errors," Information and Software Technology, 2009.

[10] O. Dieste and N. Juristo, "Systematic Review and Aggregation of Empirical Studies on Elicitation Techniques," IEEE Transactions on Software Engineering, 2009.

[11] J. Nicolás and A. Toval, " On the generation of requirements specifications from software engineering models: A systematic literature review," Information and Software Technology, 2009.

[12] M. Ivarsson and T. Gorschek, "Technology transfer decision support in requirements engineering research: a systematic review of REj," Requirements Engineering vol. 14, pp. 155-175, 2009.

[13] S. Winkler and J. v. Pilgrim, "A survey of traceability in requirements engineering and model-driven development " Software and Systems Modeling, pp. 1619-1366, 2009.

[14] B. A. Kitchenham, "Guidelines for performing Systematic Literature Reviews in Software Engineering Version 2.3," 2007.

[15] H. Zhang and M. A. Babar, "On searching relevant studies in software engineering," in 14th International Conference on Evaluation and Assessment in Software Engineering (EASE'10), 2010.

[16] A. B. Barragans Martinez and J. J. Pazos Arias, "X-SCTL/MUS: a formal methodology to evolve multi-perspective software requirements specifications," in Software Engineering, 2004. ICSE 2004. Proceedings. 26th International Conference on, 2004, pp. 72-74.

[17] A. B. B. Martinez, J. J. P. Arias, and A. F. Vilas, "Merging requirements views with incompleteness and inconsistency," in Software Engineering Conference, 2005. Proceedings. 2005 Australian, 2005, pp. 58-67.

[18] A. Loconsole, Borstler J. , "An Industrial Case Study on Requirements Volatility Measures," in Proceeding of APSEC - 12th IEEE Asia Pacific Software Engineering Conference, Taipei, Taiwan, 2005.

[19] A. a. B. J. Loconsole, "Are Size Measures Better than Expert Opinion? An Industrial Case Study on Requirements Volatility," in Proceedings of the 14th IEEE Asia Pacific Conference on Software Engineering, Nagoya, Japan, 2007.

[20] C. Mao, Y. Lu, and X. Wang, "A Study on the Distribution and Cost Prediction of Requirements Changes in the Software Life-Cycle," in Unifying the Software Process Spectrum, 2006, pp. 136-150.

[21] W. Lam, M. Loomes, and V. Shankararaman, "Managing requirements change using metrics and action planning," in Software Maintenance and Reengineering, 1999. Proceedings of the Third European Conference on, 1999, pp. 122-128.

[22] W. Lam and V. Shankararaman, "Requirements change: a dissection of management issues," in EUROMICRO Conference, 1999. Proceedings. 25th, 1999, pp. 244-251 vol.2.

[23] A. M. Davis, N. Nurmuliani, P. Sooyong, and D. Zowghi, "Requirements Change: What's the Alternative?," in Computer Software and Applications, 2008. COMPSAC '08. 32nd Annual IEEE International, 2008, pp. 635-638.

[24] in Online Webster, http://www.merriam-webster.com/.

[25] S. Anderson and M. Felici, "Controlling Requirements Evolution: An Avionics Case Study," in Computer Safety, Reliability and Security, 2000, pp. 361-370.

[26] L. Lin, S. J. Prowell, and J. H. Poore, "The impact of requirements changes on specifications and state machines," Software—Practice & Experience, vol. 39, pp. 573-610, 2009.

[27] M. Lormans, H. van Dijk, A. van Deursen, E. Nocker, and A. de Zeeuw, "Managing evolving requirements in an outsourcing context: an industrial experience report," in Software Evolution, 2004. Proceedings. 7th International Workshop on Principles of, 2004, pp. 149-158.

[28] M. Lormans, "Monitoring Requirements Evolution using Views," in Software Maintenance and Reengineering, 2007. CSMR '07. 11th European Conference on, 2007, pp. 349-352.

[29] C. Chang and L. Cai, "Agent based requirements evolution over the Internet," in Applications and the Internet Workshops, 2001. Proceedings. 2001 Symposium on, 2001, pp. 83-88.

[30] B. Xu, X. Yang, Z. He, and S. R. Maddineni, "Extreme programming in reducing the rework of requirement change," in Electrical and Computer Engineering, 2004. Canadian Conference on, 2004, pp. 1567-1570 Vol.3.

[31] J. Tomayko, "Managing Evolving Requirements Using eXtreme Programming," in Soft-Ware 2002: Computing in an Imperfect World, 2002, pp. 13-18.

[32] E. Anne and S. Camille, "Managing Requirements in a Co-evolution Context," in Proceedings of the 13th IEEE International Conference on Requirements Engineering: IEEE Computer Society, 2005.

[33] G. H. Galal and R. J. Paul, "A Qualitative Scenario Approach to Managing Evolving Requirements," Requirements Engineering, vol. 4, pp. 92-102, 1999.

[34] M. Messaadia, M. H. El-Jamal, and A. E. K. Sahraoui, "On systems engineering deployment and requirements evolution," in Systems Engineering, 2005. ICSEng 2005. 18th International Conference on, 2005, pp. 427-432.

[35] A. von Knethen, "Change-oriented requirements traceability. Support for evolution of embedded systems," in Software Maintenance, 2002. Proceedings. International Conference on, 2002, pp. 482-485.

[36] M. Kassab, O. Ormandjieva, and M. Daneva, "A Traceability Metamodel for Change Management of Non-functional Requirements," in Software Engineering Research, Management and Applications, 2008. SERA '08. Sixth International Conference on, 2008, pp. 245-254.

[37] A. K. Thurimella and B. Bruegge, "Evolution in Product Line Requirements Engineering: A Rationale Management Approach," in Requirements Engineering Conference, 2007. RE '07. 15th IEEE International, 2007, pp. 254-257.

[38] S.-N. Arash and H. Volker, "Towards a framework for requirement change management in healthcare software applications," in Companion to the 22nd ACM SIGPLAN conference on Object-oriented

programming systems and applications companion Montreal, Quebec, Canada: ACM, 2007.

[39] C. J. Burgess, I. Dattani, G. Hughes, J. H. R. May, and K. Rees, "Using Influence Diagrams to Aid the Management of Software Change," Requirements Engineering, vol. 6, pp. 173-182, 2001.

[40] L. Aversano and M. Tortorella, "An assessment strategy for identifying legacy system evolution requirements in eBusiness context," Journal of Software Maintenance and Evolution: Research and Practice, vol. 16, pp. 255-276, 2004.

[41] S. Ramzan and N. Ikram, "Making Decision in Requirement Change Management," in Information and Communication Technologies, 2005. ICICT 2005. First International Conference on, 2005, pp. 309-312.

[42] V. Nanda and N. H. Madhavji, "The impact of environmental evolution on requirements changes," in Software Maintenance, 2002. Proceedings. International Conference on, 2002, pp. 452-461.

[43] A. Kobayashi and M. Maekawa, "Need-based requirements change management," in Engineering of Computer Based Systems, 2001. ECBS 2001. Proceedings. Eighth Annual IEEE International Conference and Workshop on the, 2001, pp. 171-178.

[44] G. Stark, A. Skillicorn, and R. Smeele, "A micro and macro based examination of the effects of requirements changes on aerospace software maintenance," in Aerospace Conference, 1998. Proceedings., IEEE, 1998, pp. 165-172 vol.4.

[45] S. Anderson and M. Felici, "Quantitative aspects of requirements evolution," in Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International, 2002, pp. 27-32.

[46] G. E. Stark, P. Oman, A. Skillicorn, and A. Ameele, "An examination of the effects of requirements changes on software maintenance releases," Journal of Software Maintenance: Research and Practice, vol. 11, pp. 293-309, 1999.

[47] S. Uehara, O. Mizuno, Y. Itou, and T. Kikuno, "An MVC-based analysis of object-oriented system prototyping for banking related GUI applications-correlationship between OO metrics and efforts for requirement change," in Object-Oriented Real-Time Dependable Systems, 1999. Proceedings. Fourth International Workshop on, 1999, pp. 65-73.