View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by UPCommons. Portal del coneixement obert de la UPC

Developing a Survey to Collect Expertise in Agile Product Line Requirements Engineering Processes

Technical Report co-published as
UTDCS-18-07 The University of Texas at Dallas
and
LSI-07-15-R Universitat Politècnica de Catalunya

Version 1.0
July, 2007

Kunwu Feng
The University of Texas at Dallas
kunwu.feng@students.utdallas.edu

Meli Lempert
Universitat Politècnica de Catalunya
Meli.lempert@gmail.com

Yan Tang
The University of Texas at Dallas
tangyan1981@students.utdallas.edu

Kun Tian
The University of Texas at Dallas
kun.tian@students.utdallas.edu

Kendra M.L. Cooper
The University of Texas at Dallas
kcooper@utdallas.edu

Xavier Franch
Universitat Politècnica de Catalunya
franch@lsi.upc.edu

Table of Contents

List of Figures

List of Tables

Change history

Version 1.0	July 24, 2007	Version 1.0 released

1. Introduction

Agile development practices [20] respect “the quality of being agile; readiness for motion; nimbleness, activity, dexterity in motion”. Agile methods offer solutions that provide lighter weight, faster, and nimbler software development processes that allow developers to quickly create high quality software in rapidly changing business environments. However, the current agile methods, like XP or Scrum, are primarily focused on developing smaller, more people-oriented (less plan-oriented) projects. In many application domains, agile methods for larger, product line projects are needed. Software product line engineering techniques support creating a portfolio of similar software systems from a shared set of software assets. Many of the proposed product line engineering techniques have been plan-oriented approaches and tend to be heavy weight processes. Our overall research project focusses on integrating agile and product line engineering techniques, with an emphasis on the requirements engineering activity.

One aim of the overall project approach is to develop an expert system that can assist a requirements engineer in selecting a requirements engineering process that is well suited for their project, in particular with respect to the use of agile and product line engineering methods. To the best of our knowledge, this is the first expert system to do this.

The development of an expert system generally has two main steps (refer to Figure 1). The first is to acquire knowledge from experts in the domain. The second is to embody, or represent, the knowledge; this can be accomplished using a variety of approaches including rule based, Bayesian Belief Networks, etc. Once represented, the knowledge can be reasoned about.

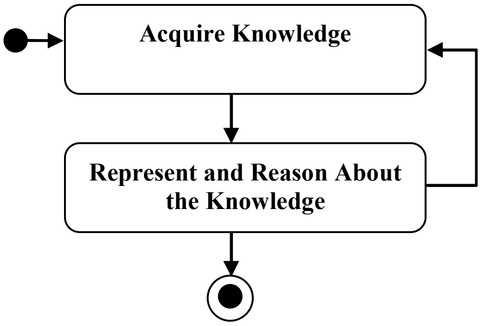


Figure 1 Overview of Developing the Expert System

Knowledge acquisition is achieved in our research by developing a questionnaire and obtaining the expertise of researchers and practitioners actively involved in software development using agile, product line engineering techniques. Questionnaires are frequently used in quantitative marketing research and social research in general. They are a valuable method of collecting a wide range of information from a large number of respondents. In addition, our questionnaire is web-based, as our (international) respondents are geographically distributed.

The focus of this report is on the specification of the questionnaire. We anticipate the specification of the questionnaire to evolve and plan to release revised versions of this technical report to reflect the changes. The questionnaire is systematically developed (refer to Figure 2); the main steps are described below.

Prepare Background/Foundation. Several activities are performed to establish a solid foundation for this research. The first activity is to carry out a rigorous literature survey in three key areas: Requirements Engineering (RE) approaches in Agile Methods, RE approaches in Product Line Engineering, and established work in the area of questionnaire design. For the literature survey for RE in agile and product line engineering approaches, 66 articles and books are considered. Two main sources [11] [16] are utilized to identify the characteristics of a "good" questionnaire. These are summarized in Appendix A. The second activity involves analyzing the 12 principles of agile methods [20] and ranking them with respect to their impact and relationship to established RE activities (elicitation, specification, analysis, validation, and management). The relationship of each principle to the RE activities are ranked using three values: high, medium, and low. This analysis is necessary to focus the questionnaire on the appropriate area and elicit good questions to ask the experts. The detailed results of the analysis are available in Appendix B.

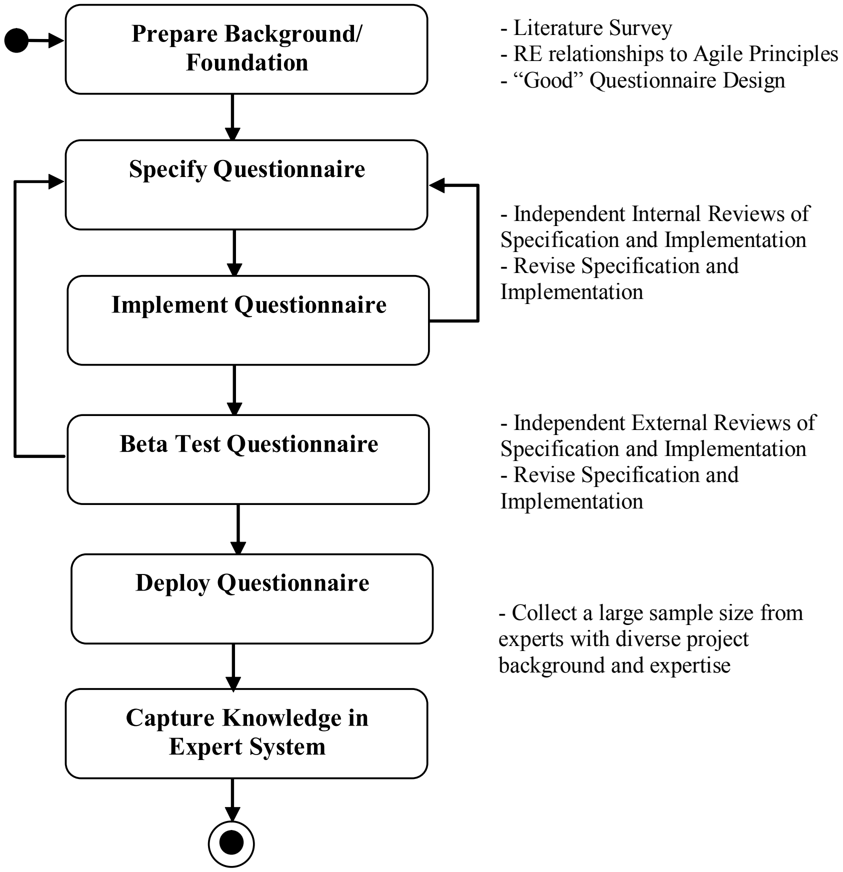

Prepare Background/ Foundation
- Literature Survey
- RE relationships to Agile Principles
- "Good" Questionnaire Design
Specify Questionnaire
- Independent Internal Reviews of Specification and Implementation
- Revise Specification and Implementation
Implement Questionnaire
Beta Test Questionnaire
- Independent External Reviews of Specification and Implementation
- Revise Specification and Implementation
Deploy Questionnaire
- Collect a large sample size from experts with diverse project background and expertise
Capture Knowledge in Expert System

Figure 2 Developing the Questionnaire

Specify the Questionnaire. With the background in place, the next step was to specify the questionnaire. The questions and structure of the questionnaire were iteratively defined, reviewed and corrected.

Brainstorm/identify possible questions. A collection of possible questions is generated; an initial structure for the questionnaire is proposed. This activity is driven using the analysis of the RE activities with respect to the agile principles and the knowledge from the literature survey in agile methods and product line engineering techniques. The results of the analysis of the RE activities indicate that principles 2, 3, 6 and 10 are the most highly related; in addition 1 and 4 also rate careful attention in our questionnaire development. For example, Principle 6 is “The most efficient and effective method of conveying information to and within a development team is face-to-face conversation”; it was ranked as highly related to all RE activities. Based on this, questions about the level of stakeholder interaction, project size, and geographic distribution of the stakeholders were proposed. The analysis results are available in Appendix B.

Propose Questions and Structure. Using “good” questionnaire design principles, questions/possible responses and the overall structure of the questionnaire are defined. Examples of “good” questionnaire design principles include a) begin with a few easy (non-threatening) and interesting questions that introduce the respondents to the questionnaire; b) group the questions into logically coherent sections; and c) organize the questions into a meaningful order and format. Additional principles of designing a “good” questionnaire have been summarized in Appendix A.

The overall structure of the questionnaire into two main parts converged rapidly, by the third iteration. For each main part, however, defining the finer grain structure and the specific questions required numerous iterations. The purpose of the first part is to collect information about the expert’s specific area of expertise. For example, some experts may work in embedded software development, web based software development, information systems, and so on. The purpose of the second part of the questionnaire is to present a small set of project scenarios to the expert, which are as closely related to their area of expertise as possible, and obtain the expert’s opinion about what kind of RE process to use on these projects. In turn, based on the data collected from the second part of the questionnaire, a decision network will be developed to provide various options regarding to specific software product line and agile method techniques for each phase of requirement engineering for a specific project.

Implement the Questionnaire. The questionnaire has been developed as a web based application, which allows for convenient, international access. It is implemented using C#; data from the responses are stored in a MySQL database. The questionnaire has been implemented in two major iterations. In each iteration a section of the questionnaire was implemented, reviewed (independently reviewed by non-developers on the team), and corrected. The first iteration was for Part I and the second iteration was for Part II. The questionnaire is available at [19].

Beta Test the Questionnaire. The questionnaire is beta tested; a set of 6 experts in the community has been asked to fill in the questionnaire. They are provided with a review form to help identify issues, concerns and problems with the questionnaire. The beta testers have been encouraged to provide frank comments and critiques about the questionnaire. Based on the feedback from the beta testers, the questionnaire is updated.

Deploy the Questionnaire. Once the questionnaire is updated, it is disseminated to the experts in the community. Researchers and practitioners involved with program committees and/or participants at related workshops (e.g., APLE 2006, RWASE 2007, SPLC 2007) are invited to respond; in addition general announcements are to be made on related newsgroups. A minimum sample size of 40 respondents is our goal; we anticipate the respondents to be a mix of researchers and practitioners.

Capture Knowledge in an Expert System. The data from the community are collected, analyzed, and embodied in an expert tool. The expert system is architected as a two layered system. The GUI layer provides forms to collect user input and provides the ability to visualize the recommended RE processes. The engine layer is designed using a Bayesian Belief Network. The tool is to be made freely available via a website.

The structure of this report is as follows. Section 2 presents the specification for Part I of the questionnaire; Section 3 presents the specification of Part II. Conclusions are in Section 4.

2. Questionnaire Part I Specification

The specification for Part I of the questionnaire is presented in this section of the report.

Obtain expert's background on their use of agility, re-use (in particular product line engineering techniques on a specific project (mostly completed or completed)

Please answer the questions in Part I with a specific project in mind, either substantially or fully completed. At the end of the questionnaire, you will have the opportunity to repeat the questionnaire for another project.

1.1 Backgrounds/Profile

1.1.1 Your experience in requirements engineering

1) Select the option that best characterizes your experience as a requirements engineer, or business analyst, in agile product line development. *(Select one option)*

    1. <2 years
    2. 2-3 years
    3. 4-6 years
    4. >7 years

1.1.2 Project Type (new, on-going, re-engineering)

2) Select the option that best characterizes your project. *(Select one option)*

    1. The project was a new project (Greenfield engineering)

    2. The project was an on-going project (adding new capabilities or maintenance)

    3. The project was a re-engineering project of a legacy system

1.1.3 Project Domain

3) Before starting the project, did the business or organization already have substantial expertise in the domain for the proposed project (e.g., organization has strong group of experts in the domain, already developed software products in the domain, etc.)? *(Select one option)*

   1. No experts available for the project with extensive experience in the domain or zero percent of expert in develop team.

   2. Few experts available for the project with extensive experience in the domain (2+ experts, developed similar product before) or 5% of experts in develop team.

   3. Some experts available for the project with extensive experience in the domain (5+ experts, developed similar product before) or 10% of experts in develop team.

   4. Many experts available for the project with extensive experience in the domain (10+ experts, developed similar product before) or 30% of experts in develop team.

   5. Numerous experts available for the project with extensive experience in the domain (20+ experts, developed similar product before). Business is the technical leader in the domain or 60% of experts in develop team.

4) What are the main characteristics of the application? *(Select one or more options)*

   1. Information system (data intensive)
   2. Real time/Embedded system
   3. Web based, service oriented distributed system (Web service / P2P)

   4. Intelligent, knowledge based system
   5. Dynamically adaptable system (Agent)
   6. COTS development
   7. Others? please specify

5) What is your domain of expertise? *(Select one or more options)*

1. Telecommunications
2. Aerospace
3. Finance (e.g., banking, insurance)

4. Healthcare
5. Medical diagnostics
6. Military/Defense

7. Entertainment (e.g., games)
8. Transportation
9. Retail (e.g., point of sale)

10. Warehousing and Distribution of goods, services
11. Consumer electronics (e.g., mobile systems)
12. Automotive systems

13. Other? Please specify.

1.1.4 Project Size

6) How many people are allocated to work on the project? (e.g., management, engineering staff) *(Select one option)*

1. <10
2. 10-25
3. 26-50

4. 51- 80
5. 81-100
6. 100-150

7. >150

7) What is the duration of the project (from inception to delivery)? *(Select one option)*

1. <6 months
2. 6 - < 12 months
3. 12 months - <24 months

4. 24 months - <36 months
5. 36 months - <48 months
6. 48 months - <60 months

7. more than 60 months

8) What is the expected SLOC (source line of code)? *(Select one option)*

1. < 10 KSLOCS
2. 10-100 KSLOCS
3. 100-1000 KSLOCS
4. > 1000 KSLOCS

1.1.5 Project Interaction with Stakeholders

9) For each of the following, how many geographic locations (different facilities, different cities, different countries, etc.) are involved for the project? *(Select one option)*

Total Number of locations (customers, developers):
1. 1 location
2. 2-4 locations
3. >5 locations

Number of customer locations (end users, decision makers)
4. 1 location
5. 2-4 locations
6. >5 locations

Number of developer locations (requirements engineers, programmers, etc)
7. 1 location
8. 2-4 locations
9. >5 locations

10) What is the frequency of face-to-face meetings? *(Select one option)*

1. Very low (meet each business quarter)
2. Low (meet each month)

3. Moderate involvement (meet each two weeks)

4. Highly involvement (meet each week)
5. Very highly involved (meet everyday)

11) What is the frequency of meetings of other kinds (e.g., phone, video-conferencing, etc.)? *(Select one option)*

1. Very low (meet each business quarter)
2. Low (meet each month)
3. Moderate involvement (meet each two weeks)

4. Highly involvement (meet each week)
5. Very highly involved (meet everyday)

12) How are the requirements validated? *(Select one option)*

1. Informal discussion providing verbal feedback, no formal action items or problem reports

2. Organized, semi-formal review providing informal written feedback (e.g. e-mail), no formal action items or problem reports

3. Organized, semi-formal review providing formal written feedback, action items or problem reports

4. Formal inspection, action items or problems reports providing formal documents

13) What is the frequency for customer stakeholders' deliverables or demonstrations? *(Select one option)*

1. Weekly
2. Monthly
3. At least every business quarter (3 months)
4. Bi-annual (6 months)
5. Annual (12 months)
6. on delivery

1.1.6 Project Requirements

14) To what degree are the requirements managed (as well as configuration management)? (i.e., in a repository, under change management control, etc.) *(Select one option)*

1. Requirements are managed verbally, ad-hoc change process is used, changes are handled verbally
2. Requirements are managed using general purpose tool (e.g., word, excel, etc.) to document the requirements and changes, moderate change process is used
3. Requirements are managed using specialized requirements management tool (e.g., DOORS, etc.), rigorous change process is used

15) To what degree are the requirements documented? *(Select one option)*

1. Requirements are not documented in writing at all.
2. Requirements are documented in e-mail and meeting minutes reports; ad-hoc documentation is used.
3. Requirements are semi-formally documented using general purpose tool (e.g., word, etc.), moderate documentation is used

4. Requirements are formally documented using specialized documentation tool (e.g., rational rose, etc.), rigorous documentation is used

16) To what degree are the requirements traced? (i.e., traced to design or test artifacts) *(Select one option)*

1. Requirements are not traced
2. Requirements are traced using general purpose tool (e.g., word, excel, etc.), few reports are moderately generated and used
3. Requirements are traced using specialized requirements management tool (e.g., DOORS, etc.), reports are extensively generated and used

17) Which of the following non-functional requirements have high priority in the project? *(Select one option)*

1. Safety
2. Security
3. Reliability

4. Availability
5. Scalability
6. Response time performance

7. Compliance with standards (international, de-facto)
8. System resources requirements (i.e., CPU, memory footprint)
9. Others? please specify

18) To what degree does the customer understand the requirements (functional and non-functional) for the project? *(Select one option)*

1. Very clear vision
2. Moderately clear vision, some uncertainty
3. Vision is not clear, significant uncertainty

19) What is the churn rate (i.e., change rate) on the requirements? *(Select one option)*

1. Very low (few, infrequent changes)
2. Low
3. Medium
4. High
5. Very high (many, frequent changes)

20) What elicitation techniques are used to capture the requirements? *(Select one or more options)*

1. Brainstorming
2. On-site interactive sessions
3. Electronic interactive sessions

4. Focus Groups
5. Review of existing documentation

6. Scenarios
7. Goal modeling
8. Paper prototypes

9. Executable Prototypes
10. Story boards
11. Observation and social analysis

12. Others? please specify

21) What tools are used to capture the requirements? *(Select one or more options)*

1. None (verbal communication)
2. Paper notes or cards
3. Spreadsheet or word processor (e.g., excel, word)

4. Drawing tool (e.g., Visio)
5. CASE tool (e.g. Rational Rose, other)
6. Others? please specify

1.2 Agile Requirements Methods Used

22) Was an agile requirement engineering method used from the beginning or was it introduced later? *(Select one option)*

1. Yes. From the beginning
2. No. Later

If introduced later, why? *(Select one or more options)*

1. Need to streamline the RE process to meet the time constraint.
2. Determined that the rigorous RE process was unnecessary.
3. Other reasons? Please specify.

23) Was an agile requirement engineering method modified to be less agile or removed during the project? *(Select one option)*

1. Yes
2. No

If so, why? *(Select one or more options)*

1. Need to modify the Agile RE process to make it more suitable for your project, which requires more rigorous RE activities for details and formality.
2. Applying Agile RE to your project was unsuccessful from the developer's perspective
3. Applying Agile RE to your project was unsuccessful from the customer's perspective.

4. Others, please specify

24) What Agile RE methods did you use in your project, either as defined in books available or a tailored version? *(Select one or more options)*

1. None
2. Extreme Programming (XP)
3. Scrum

4. Crystal Clear and Other Crystal Methodologies (Crystal)
5. Dynamic Systems Development Method (DSDM)
6. Adaptive Software Development (ASD)

7. Agile Modeling Driven Development (AMDD)
8. Lean Software Development
9. Agile Documentation

10. Feature Driven Development (FDD)
11. Others? Please specify.

25) How suitable was the use of Agile Requirements for your project? *(Select one option)*

1. Not very suitable, numerous problems related to the use of agile methods
2. Suitable, worked well on the project, few problems related to the use of agile methods
3. Very suitable, minor problems related to the use of agile methods

1.3 Reuse and Product Line Methods Used

26) How much of existing requirement specifications are re-used? *(Select one option)*

1. Very low (few requirements are reused)
2. Low
3. Moderate

4. High
5. Very high (a large portion of requirements are used)

27) Did your RE process provide systematic support for the re-use of requirements? *(Select one option)*

1. No (Only optimistically reused the requirements)
2. Yes, seldom
3. Yes, only moderately

4. Yes, very often
5. Yes, frequently

28) Is your project part of a product line? *(Select one option)*

1. No (single product)
2. Yes, this is the first product in the product line
3. Yes, this is the second product in the product line

4. Yes, this is at least the third product in the product line

29) Did you perform domain analysis for this project? *(Select one option)*

1. No (Go to question 32)
2. Yes

If yes, what tools did you use? *(Select one or more options)*

1. Knowledge extraction tools

2. Entity-relationship diagramming tools
3. Object-oriented development tools

4. Semantic clustering and automatic classification tools
5. Computer-aided software engineering tools
6. Others? Please specify

30) If domain analysis was performed, then how many variation points were identified? *(Select one option)*

1. <10
2. 10- <20
3. 20-<50
4. >50

31) If domain analysis was performed, then how were the variation points distributed in the requirements? *(Select one option)*

1. Mostly in the functional requirements
2. Mostly in the non-functional requirements
3. Approximately the same in the functional and non-functional requirements

32) If you reuse requirements, how do you maintain the reusable items? *(Select one option)*

1. Reusable items are communicated verbally, and maintained by tacit knowledge
2. Reusable items are managed using general purpose tool and are stored in a centralized depository with semi-formal change management process
3. Reusable items are managed using specialized requirements management tool, and rigorous (formal) change process is used

33) What product line methods did you use in your project? *(Select one or more options)*

1. Family Oriented Abstraction, Specification and Translation (FAST)
2. Synthesis
3. Feature Oriented Domain Analysis (FODA)

4. Product Line Software Engineering (Pulse)
5. Sherlock
6. Odyssey-DE

7. Pohl's SPLE Approach (or a tailored version of it)
8. Clements' SPLE Approach (or a tailored version of it)
9. Others? Please specify.

34) Was product line engineering method suitable for your projects? *(Select one option)*

1. Not very suitable, numerous problems related to the use of product line methods
2. Suitable, worked well on the project, few problems related to the use of product line methods
3. Very suitable, minor problems related to the use of product line methods

3. Questionnaire Part II Specification

The purpose of this part is to obtain expert's opinion on degree of agility and product line requirements engineering techniques that are best suited for specific project scenarios.

In Part II of the questionnaire, a set of project scenarios that are highly related to the experts' area of expertise are presented to the user. The selection is based on answers provided in questions No. 2, 3, 5, 6, 7, 8, 9, 13, and 28 in Part I. The selection algorithm is presented below. For each project scenario that is selected, the expert is asked a set of questions about what RE process they believe is the best suited for the project. The development of the project scenarios is presented in [9].

3.1. Select Project Scenarios to Present to the Expert

A comprehensive set of project scenarios has been defined, which includes 9 sets of project scenarios (162 scenarios). An algorithm to select the project scenarios that are the closest match to the experts' background is needed. The "Narrow-Down Algorithm" is proposed in this work to accomplish this. The responses to questions in Part I of the questionnaire are used in this algorithm.

When the experts submit the answers of Questionnaire Part I, the Narrow Down Algorithm is running to select several most relevant scenarios presented to experts. The Algorithm is executed based on 9 properties, which are the expert answers provided in questions No. 2, 3, 5, 6, 7, 8, 9, 13, and 28 in Part I. The properties are Project characteristic, Project domain, New/enhanced project, Number of developers, Project duration, Lines of code, Project interaction, Demo frequency and Agile/Product Line project.

For instance, one of the expert answers could be as follows:

- Project characteristic: Information System (data intensive)
- Project domain: Finance System

- New/enhanced project: new project
- Number of developers: 25-50 developers
- Project duration: 6-12 months
- Lines of code: 10-100 KSLOCS
- Project interaction: 2-4 locations
- Demo frequency: Monthly
- Agile/Product Line project: Agile project

The Algorithm need to find a small set, which are most relevant scenarios based on those answers.

The algorithm to select a subset of the scenarios has five main activities (refer to Figure 3).

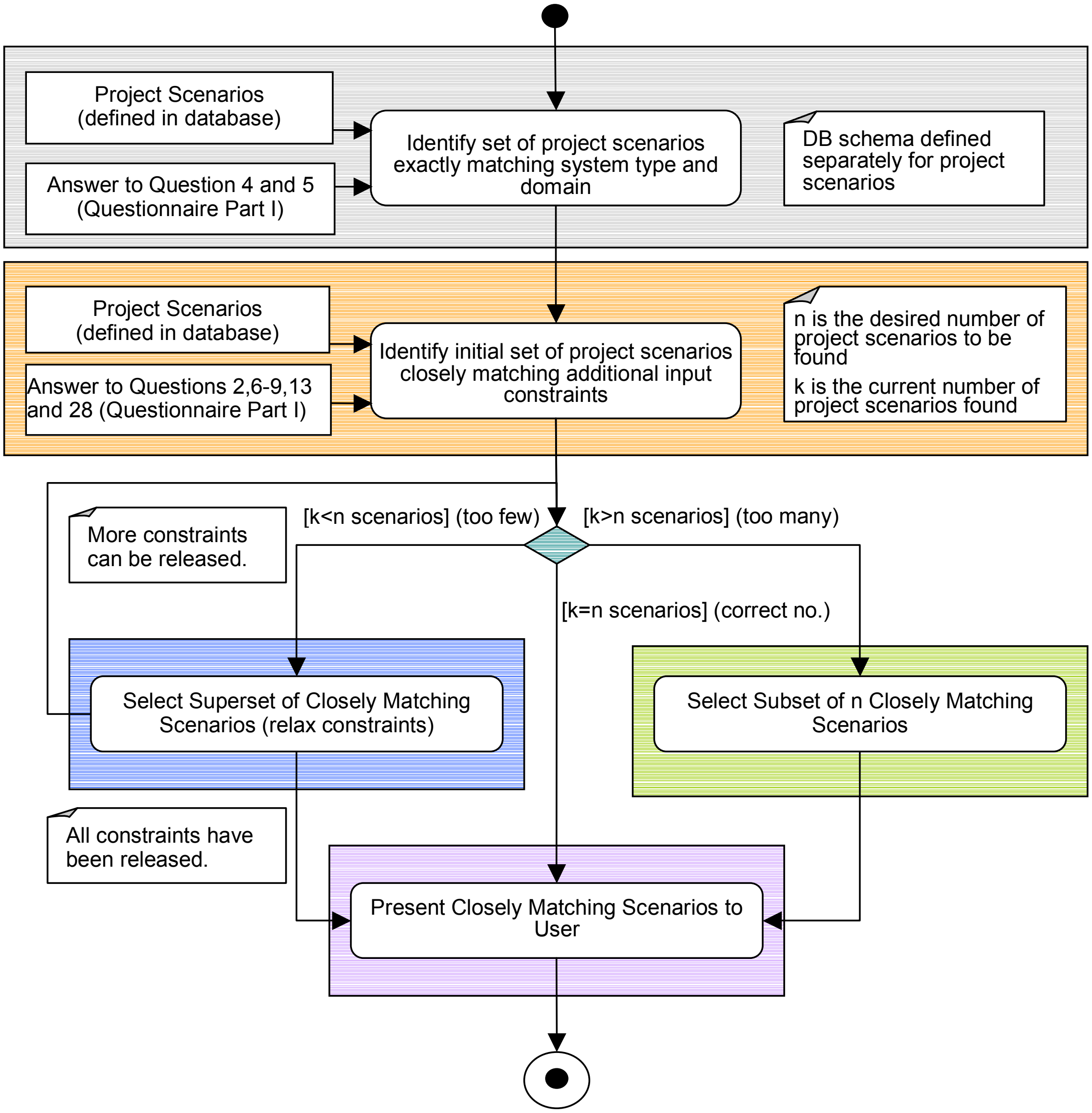

Project Scenarios (defined in database)
Answer to Question 4 and 5 (Questionnaire Part I)
Identify set of project scenarios exactly matching system type and domain
DB schema defined separately for project scenarios
Project Scenarios (defined in database)
Answer to Questions 2,6-9,13 and 28 (Questionnaire Part I)
Identify initial set of project scenarios closely matching additional input constraints
n is the desired number of project scenarios to be found
k is the current number of project scenarios found
More constraints can be released.
[k<n scenarios] (too few)
[k>n scenarios] (too many)
[k=n scenarios] (correct no.)
Select Superset of Closely Matching Scenarios (relax constraints)
Select Subset of n Closely Matching Scenarios
All constraints have been released.
Present Closely Matching Scenarios to User

Figure 3 Overview Activity Diagram for Narrow-Down Algorithm

The first step is to **Identify set of project scenarios exactly matching system type and domain.** The inputs are the project scenarios and the expert answers, and the outputs are available scenario sets that exactly match the project type and domain. If no scenario returns, then the algorithm is terminated. Exact matches are required here, as expertise is often defined within the context of a domain and project type.

The next step is to **Identify initial set of project scenarios closely matching additional input constraints**, including new/enhanced project, number of developers, project duration, lines of code, project interaction, demo frequency, and Agile/PL project. The output of the step is a set of available scenarios that closely match the expert's responses.

As described above, the system will present a subset of scenarios to the experts. Here, we use n to represent the desired number of scenarios. If the number of the available scenarios is less than n, then the activity, **Select Superset of Closely Matching Scenarios (relax constraints)**, is used. If the number of the available scenarios is larger than n, then the activity to **Select Subset of n closely Matching Scenarios is used.** If the number of the available scenarios is exactly n, then the activity to **Present Closely Matching Scenarios to User** is used.

Figure 4, 5, and 6 are the decomposed (lower level) activity diagrams for **Identify initial set of project scenarios closely matching additional input constraints**, **Select Superset of Closely Matching Scenarios (relax constraints)**, and **Select Subset of n closely Matching Scenarios**.

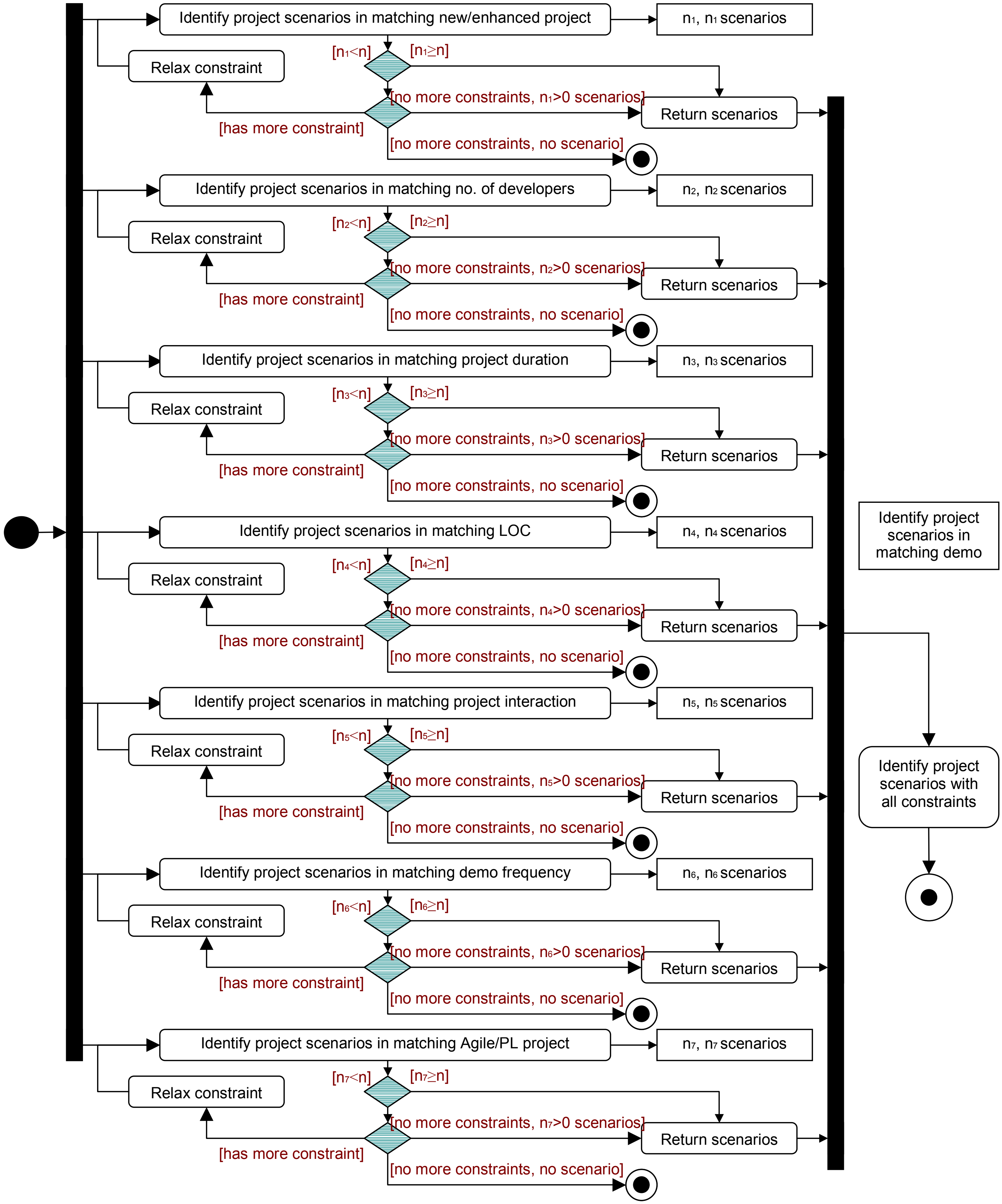

Identify project scenarios in matching new/enhanced project
n_1, n_1 scenarios
Relax constraint
[n_1<n]
[n_1≥n]
[no more constraints, n_1>0 scenarios]
Return scenarios
[has more constraint]
[no more constraints, no scenario]
Identify project scenarios in matching no. of developers
n_2, n_2 scenarios
Relax constraint
[n_2<n]
[n_2≥n]
[no more constraints, n_2>0 scenarios]
Return scenarios
[has more constraint]
[no more constraints, no scenario]
Identify project scenarios in matching project duration
n_3, n_3 scenarios
Relax constraint
[n_3<n]
[n_3≥n]
[no more constraints, n_3>0 scenarios]
Return scenarios
[has more constraint]
[no more constraints, no scenario]
Identify project scenarios in matching LOC
n_4, n_4 scenarios
Relax constraint
[n_4<n]
[n_4≥n]
[no more constraints, n_4>0 scenarios]
Return scenarios
[has more constraint]
[no more constraints, no scenario]
Identify project scenarios in matching project interaction
n_5, n_5 scenarios
Relax constraint
[n_5<n]
[n_5≥n]
[no more constraints, n_5>0 scenarios]
Return scenarios
[has more constraint]
[no more constraints, no scenario]
Identify project scenarios in matching demo frequency
n_6, n_6 scenarios
Relax constraint
[n_6<n]
[n_6≥n]
[no more constraints, n_6>0 scenarios]
Return scenarios
[has more constraint]
[no more constraints, no scenario]
Identify project scenarios in matching Agile/PL project
n_7, n_7 scenarios
Relax constraint
[n_7<n]
[n_7≥n]
[no more constraints, n_7>0 scenarios]
Return scenarios
[has more constraint]
[no more constraints, no scenario]
Identify project scenarios in matching demo
Identify project scenarios with all constraints

Figure 4 Identify Initial Set of Project Scenarios Closely Matching Additional Input Constraints

Figure 4 is the activity diagram to identify initial set of project scenarios closely matching additional input constraints. These additional input constraints are new/enhanced project, number of developers, project duration, lines of code, project interaction, demo frequency, and Agile/PL project.

The input of this step is a scenario set mapping to project type and domain. Then several activities can be executed in parallel to find relevant scenarios by matching the additional input constraints. The inputs of these activities are a set of scenarios, which are defined in [9], and experts' answers for Questions 2, 4-9, 13 and 28 (Questionnaire Part I).

Identify project scenarios in matching new/enhanced uses the response to Part I, Question 2 in the questionnaire. Several project types are defined in question 1, which are new project, on-going project and re-engineering project of a legacy system. These options are assumed as constraints. On-going projects and re-engineering projects of a legacy system are regarded as enhancement projects. Initially, using the expert's answer to find related scenarios, the number of scenarios marked as n_1. If less than n scenarios return, then relax the constraint, adding above and below options (if possible) as new condition, re-search from the input until find n or more scenarios, then record these scenarios. If when all the constraints are released, that is, all the options are using in finding related scenarios and the number of returned scenarios is still less than n, then forward these scenarios to the next activity, **Present Closely Matching Scenarios to User**. If all the constraints have been relaxed and no scenarios are found, then the algorithm is terminated.

Identify project scenarios in matching no. of developers uses the response to Part I, Question 6 in the questionnaire. The options range in size from less than 25 to greater than 150. Initially, using the expert's answer to find related scenarios, the number of scenarios is n_2. It considers options above and below exact match, in the cases when less than n scenarios return. If when all the constraints are released, and the number of returned scenarios is still less than n, then forward these scenarios to the next activity, **Present Closely Matching Scenarios to User** If all the constraints have been relaxed and no scenarios are found, then the algorithm is terminated.

Identify project scenarios in matching project duration uses the response to Part I, Question 7 in the questionnaire. Several options of project duration are presented in the question, from less than 6 months to more than 5 years. Initially, using expert answers to find related scenarios, the number of scenarios is n_3. It considers options above and below exact match, in the cases when less than n scenarios return. If when all the constraints are released, and the number of returned scenarios is still less than n, then forward these scenarios to the next activity, **Present Closely Matching Scenarios to User**. If all the constraints have been relaxed and no scenarios are found, then the algorithm is terminated.

Identify project scenarios in matching LOC uses the response to Part I, Question 8 in the questionnaire. Lines of code (LOC) is another important factor for project size. It varies from the agile respect, which are some small projects with LOC less than 10K, to product line respect, which are larger one with LOC more then 1,000K. Initially, using the expert's answer to find related scenarios, the number of scenarios is n_4. It considers options above and below exact match, in the cases when less than n scenarios return. If when all the constraints are released, and the number of returned scenarios is still less than n, then forward these scenarios to the next activity. If all the constraints have been relaxed and no scenarios are found, then the algorithm is terminated.

Identify project scenarios in matching project interaction uses the response to Part I, Question 9 in the questionnaire. Three sub-questions are relative to project locations, including total number of locations, number of customer locations and number of developer locations. Only the total number of locations is considered. Initially, using the expert's answer to find related scenarios, the number of scenarios is n_5. It considers options above and below exact match, in the cases when less than n scenarios return. If when all the constraints are released, and the number of returned scenarios is still less than n, then forward these scenarios to the next activity, **Present Closely Matching Scenarios to User**. If all the constraints have been relaxed and no scenarios are found, then the algorithm is terminated.

Identify project scenarios in matching demo frequency uses the response to Part I, Question 13 in the questionnaire. Initially, using expert answers to find related scenarios, the number of scenarios marked as n_6. It considers options above and below exact match, in the cases when less than n scenarios return. If when all the constraints are released, and the number of returned scenarios is still less than n, then forward these scenarios to the next activity, **Present Closely Matching Scenarios to User**. If after relax all the constraints, still no scenario is found, then the algorithm is terminated.

Identify project scenarios in matching Agile/PL project uses the response to Part I, Question 28 in the questionnaire. In the questionnaire, several project types are defined, which are single product, first product in the product line, second product in the product line and at least the third in the product line. Initially, using the expert's answer to find related scenarios, the number of scenarios is n_7. It considers options above and below exact match, in the cases when less than n scenarios return. If when all the constraints are released, and the number of returned scenarios is still less than n, then forward these scenarios to the next activity, **Present Closely Matching Scenarios to User**. If all the constraints have been relaxed and no scenarios are found, then the algorithm is terminated.

When all the scenarios are gathered from these activities, it is reasonable to use intersection relationship to get the final result because the system needs to select the most relative scenarios from all the scenarios. **Identify project scenarios with all constraints** is proposed in this step to accomplish this and records the scenarios and the number of related scenarios k. By comparing k and the desired number of scenarios n, three different conditions are presented in 2 activity diagrams.

Figure 5 shows the condition: if k is less than n, then too few scenarios have been found. Therefore, a relaxing constraints algorithm is proposed.

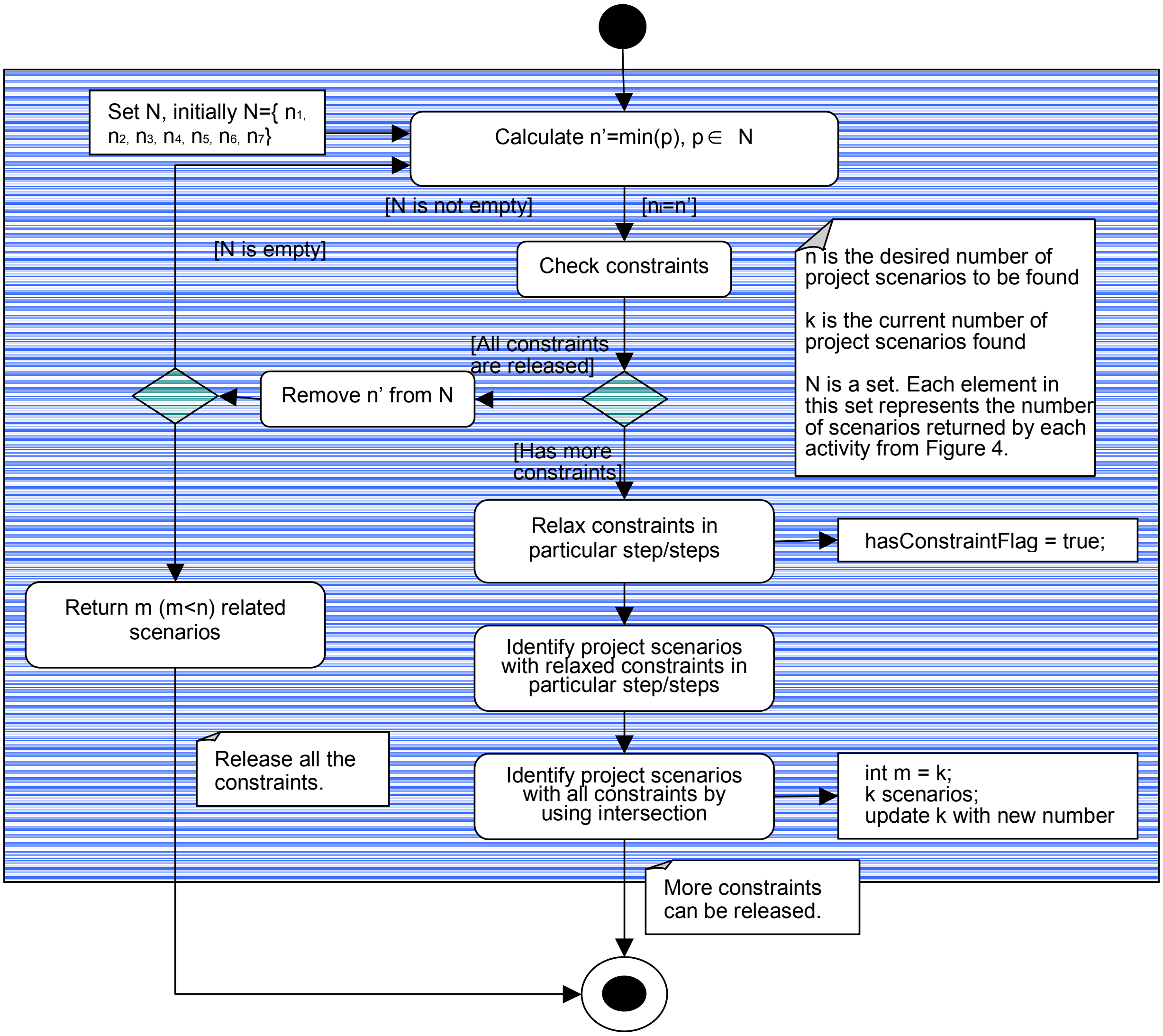

Set N, initially N={ n1, n2, n3, n4, n5, n6, n7}
Calculate n'=min(p), p∈ N
[N is not empty]
[ni=n']
[N is empty]
Check constraints
n is the desired number of project scenarios to be found
k is the current number of project scenarios found
N is a set. Each element in this set represents the number of scenarios returned by each activity from Figure 4.
[All constraints are released]
Remove n' from N
[Has more constraints]
Relax constraints in particular step/steps
hasConstraintFlag = true;
Return m (m<n) related scenarios
Identify project scenarios with relaxed constraints in particular step/steps
Release all the constraints.
Identify project scenarios with all constraints by using intersection
int m = k;
k scenarios;
update k with new number
More constraints can be released.

Figure 5 Select Superset of Closely Matching Scenarios (relax constraints)

The key point of relaxing constraints is to find which activity/activities need to re-execute. The method is described as following:

1. Store k to a new variable m.
2. Prepare a set, N. The elements in this set represent the number of scenarios returned by each activity from Figure 4. In addition, those activities do not relax all the constraints. Initially, n_1, n_2, n_3, n_4, n_5, n_6, and n_7 are the numbers of the set N.
3. Find out the minimum number/numbers from set N, marked as n'.
4. Check the constraints of the activity/activities, which correspond to n'.

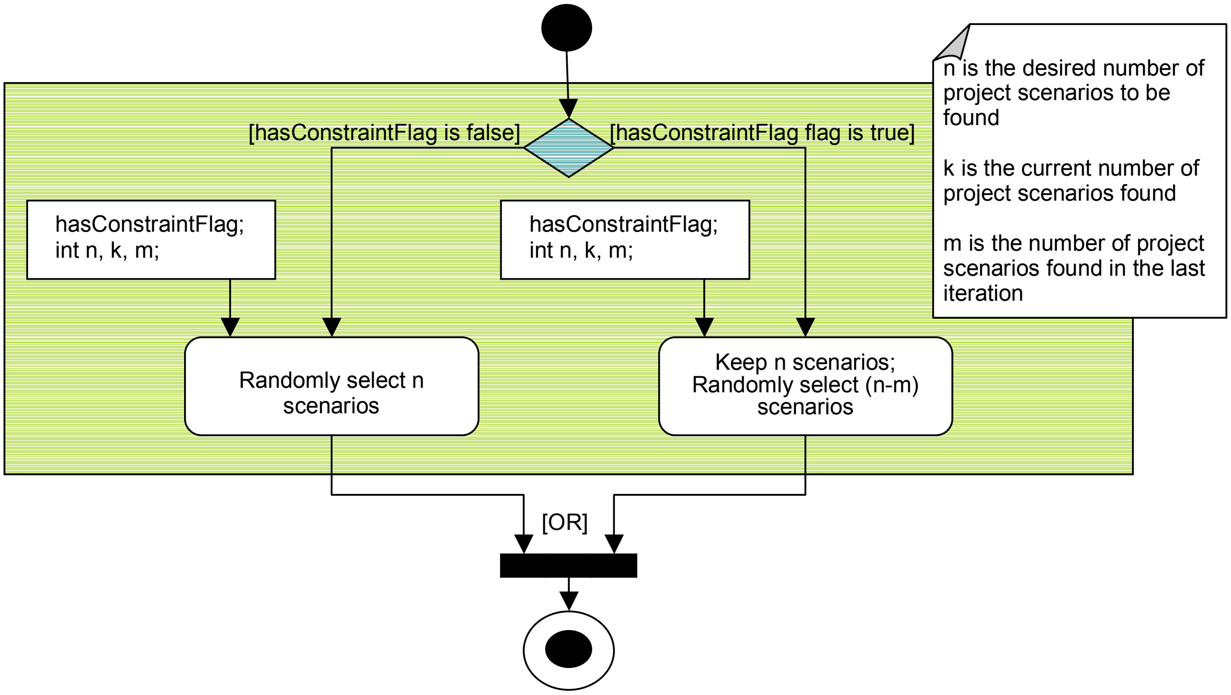
   If the activity/activities can relax more constraints,

   a) Consider options above and below current match, re-search from the input until find more scenarios, replace n_i with new value. Set the hasConstraintFlag to true.
   b) Identify project scenarios with all constraints by using intersection, get a new set of scenarios, store the old value k to integer m, and update k with the new number.
   c) Compare k with n, if k is less than n, repeat step 3.

   If the activity/activities has/have no more constraint to relax, then, all the options are selected, but still less than k scenarios are returned.

   a) Remove n' from the set N. (Because need to relax another activity's constraint and find more scenarios). If the set N is empty, then go to step 5.
   b) Repeat step 3.
5. If the set N is empty, that means, after relax all the constraints in every activity, still less than n scenarios are returned, and it is not possible to obtain n scenarios. Therefore present the matching scenarios k to the user.

Figure 6 shows the condition: if k is larger than n, then too many scenarios are found.



[hasConstraintFlag is false]
[hasConstraintFlag flag is true]
hasConstraintFlag;
int n, k, m;
hasConstraintFlag;
int n, k, m;
Randomly select n scenarios
Keep n scenarios; Randomly select (n-m) scenarios
[OR]
n is the desired number of project scenarios to be found
k is the current number of project scenarios found
m is the number of project scenarios found in the last iteration

Figure 6 Select Subset of Closely Matching Scenarios

If hasConstraintFlag is false, that means, with the original constraints which defined in **Identify initial set of project scenarios closely matching additional input constraints**, more then n scenarios are returned. They are all very close to users' answers. Therefore, randomly select n scenarios then present to user.

If hasConstraintFlag is true, that means, by using original constraints in **Identify initial set of project scenarios closely matching additional input constraints**, only m scenarios are returned, where m is less than n. Therefore, relax some constraints and select superset of closely matching scenarios, and return k scenarios, where k is larger than n. However, m scenarios are closer to users' answers. To present n scenarios to user, keep those closer scenarios and only need to randomly select (n-m) scenarios from these (k-m) scenarios.

Another situation is if k is equal to n, then present those scenarios to user directly, because these scenarios are very close to user's input.

1.4 Present Questions on the Selected Project Scenarios

The first four questions in Part II are the same for each expert. The remaining questions are dynamically generated, based on their responses to the first four questions. The generation of questions is illustrated in Figures 7-11.

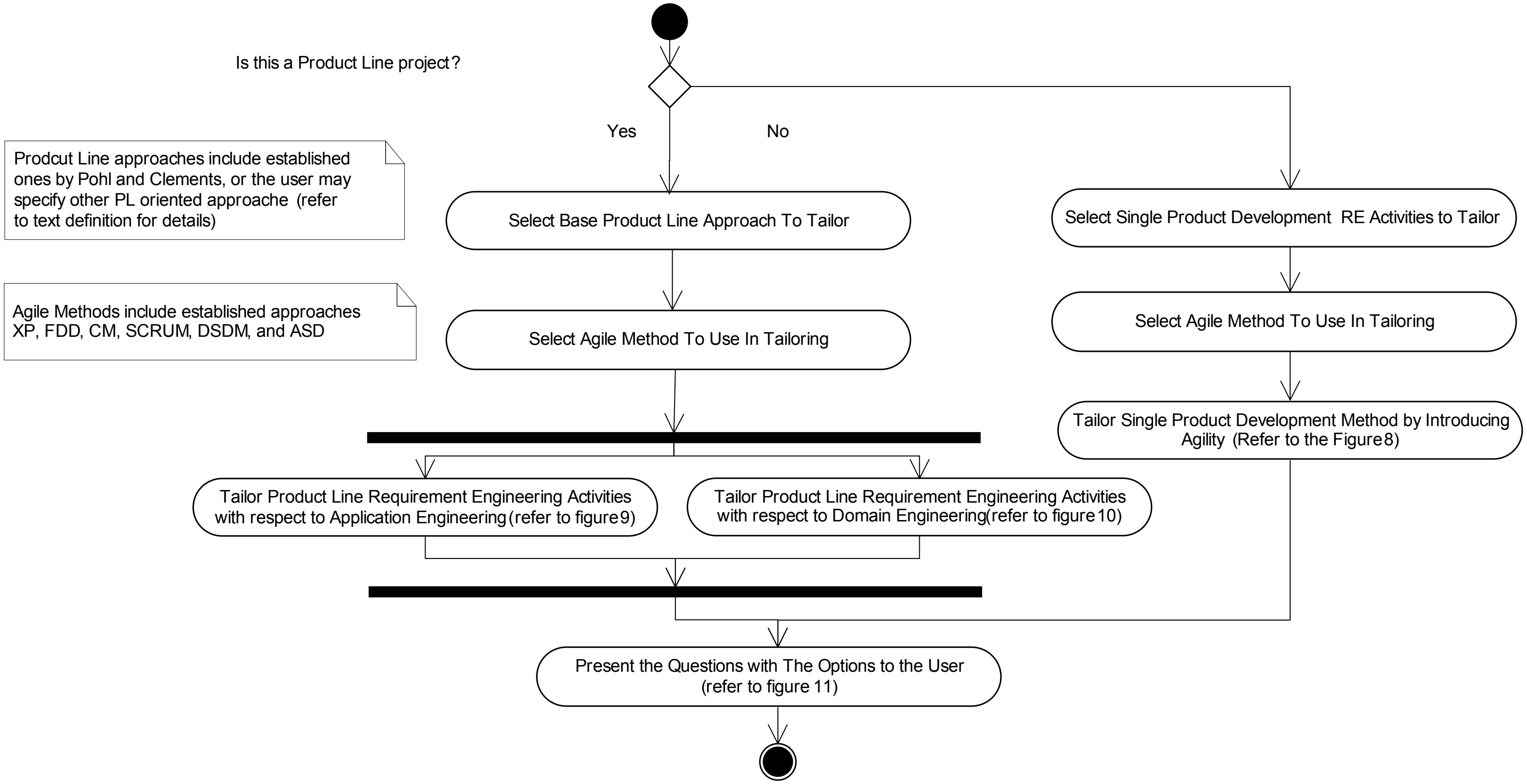

Is this a Product Line project?
Yes
No
Prodcut Line approaches include established ones by Pohl and Clements, or the user may specify other PL oriented approache (refer to text definition for details)
Agile Methods include established approaches XP, FDD, CM, SCRUM, DSDM, and ASD
Select Base Product Line Approach To Tailor
Select Agile Method To Use In Tailoring
Select Single Product Development RE Activities to Tailor
Select Agile Method To Use In Tailoring
Tailor Single Product Development Method by Introducing Agility (Refer to the Figure 8)
Tailor Product Line Requirement Engineering Activities with respect to Application Engineering (refer to figure 9)
Tailor Product Line Requirement Engineering Activities with respect to Domain Engineering(refer to figure 10)
Present the Questions with The Options to the User (refer to figure 11)

Figure 7 Activity Diagram for the Part II Questions

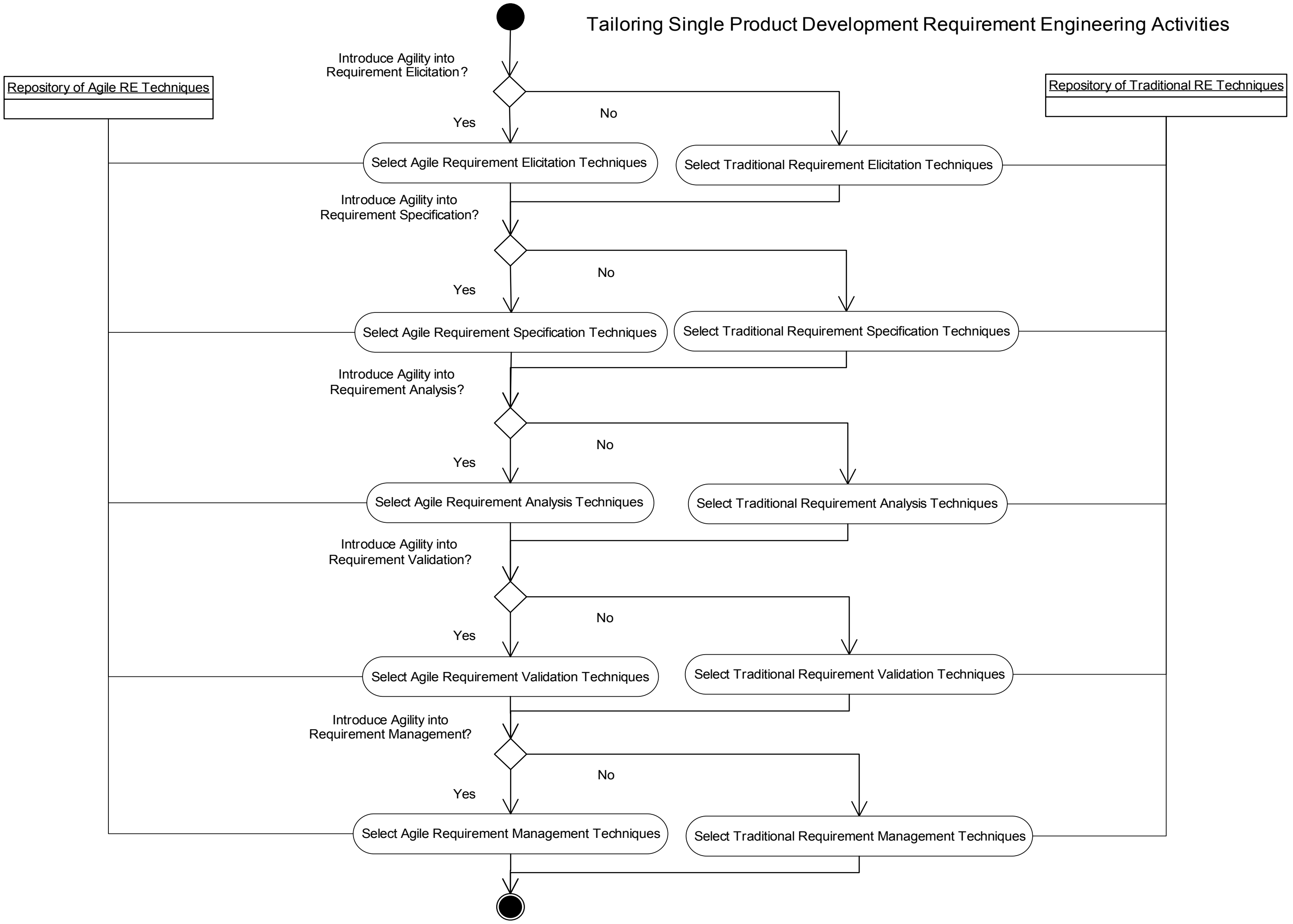
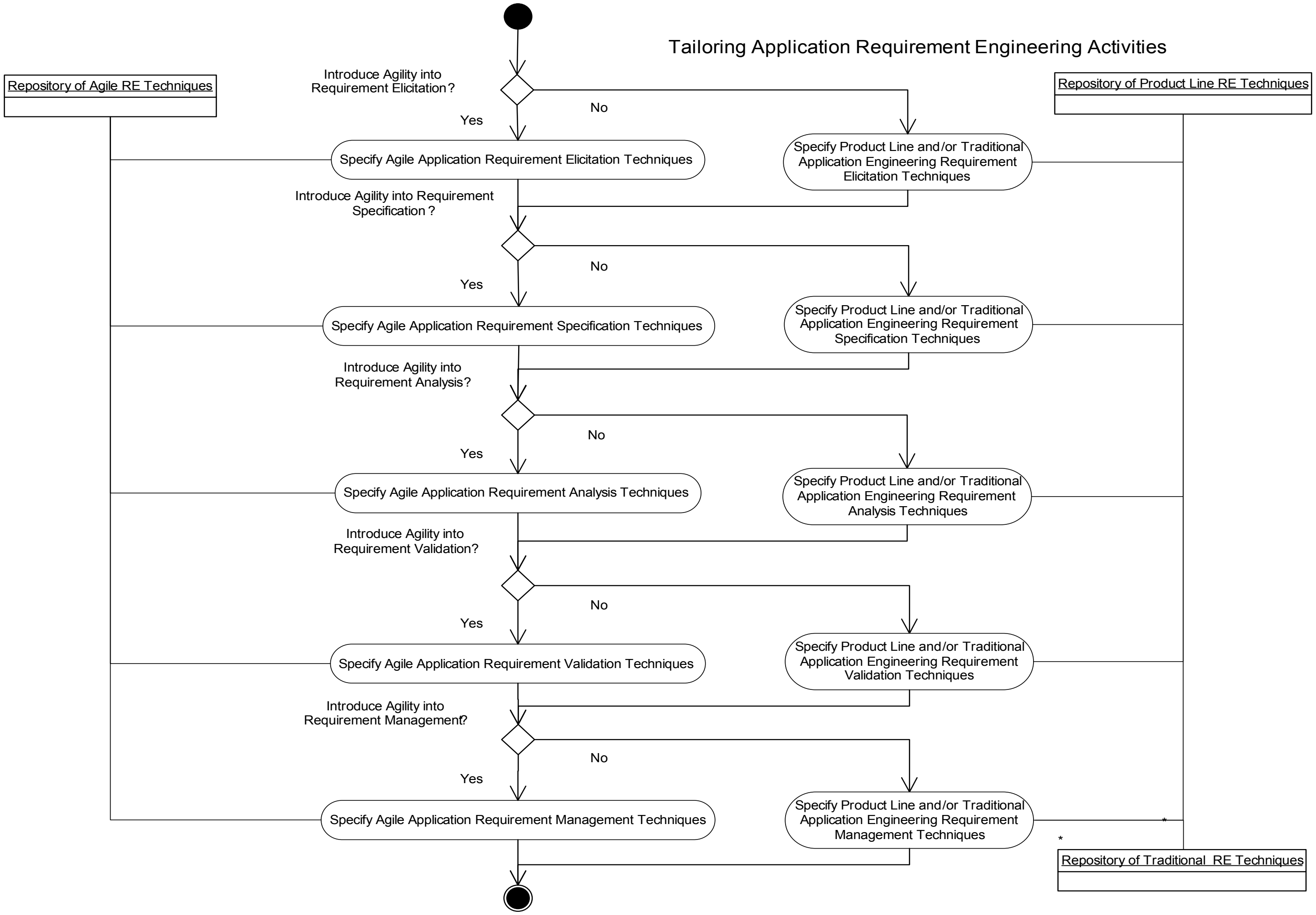

Tailoring Single Product Development Requirement Engineering Activities
Repository of Agile RE Techniques
Repository of Traditional RE Techniques
Introduce Agility into Requirement Elicitation?
Yes
No
Select Agile Requirement Elicitation Techniques
Select Traditional Requirement Elicitation Techniques
Introduce Agility into Requirement Specification?
Yes
No
Select Agile Requirement Specification Techniques
Select Traditional Requirement Specification Techniques
Introduce Agility into Requirement Analysis?
Yes
No
Select Agile Requirement Analysis Techniques
Select Traditional Requirement Analysis Techniques
Introduce Agility into Requirement Validation?
Yes
No
Select Agile Requirement Validation Techniques
Select Traditional Requirement Validation Techniques
Introduce Agility into Requirement Management?
Yes
No
Select Agile Requirement Management Techniques
Select Traditional Requirement Management Techniques

Figure 8 Tailoring Single Product Development RE Activities



Tailoring Application Requirement Engineering Activities
Repository of Agile RE Techniques
Repository of Product Line RE Techniques
Repository of Traditional RE Techniques
Introduce Agility into Requirement Elicitation?
Yes
No
Specify Agile Application Requirement Elicitation Techniques
Specify Product Line and/or Traditional Application Engineering Requirement Elicitation Techniques
Introduce Agility into Requirement Specification ?
Yes
No
Specify Agile Application Requirement Specification Techniques
Specify Product Line and/or Traditional Application Engineering Requirement Specification Techniques
Introduce Agility into Requirement Analysis?
Yes
No
Specify Agile Application Requirement Analysis Techniques
Specify Product Line and/or Traditional Application Engineering Requirement Analysis Techniques
Introduce Agility into Requirement Validation?
Yes
No
Specify Agile Application Requirement Validation Techniques
Specify Product Line and/or Traditional Application Engineering Requirement Validation Techniques
Introduce Agility into Requirement Management?
Yes
No
Specify Agile Application Requirement Management Techniques
Specify Product Line and/or Traditional Application Engineering Requirement Management Techniques
*
*

Figure 9 Tailoring Product Line RE Activities With Respect To Domain Engineering

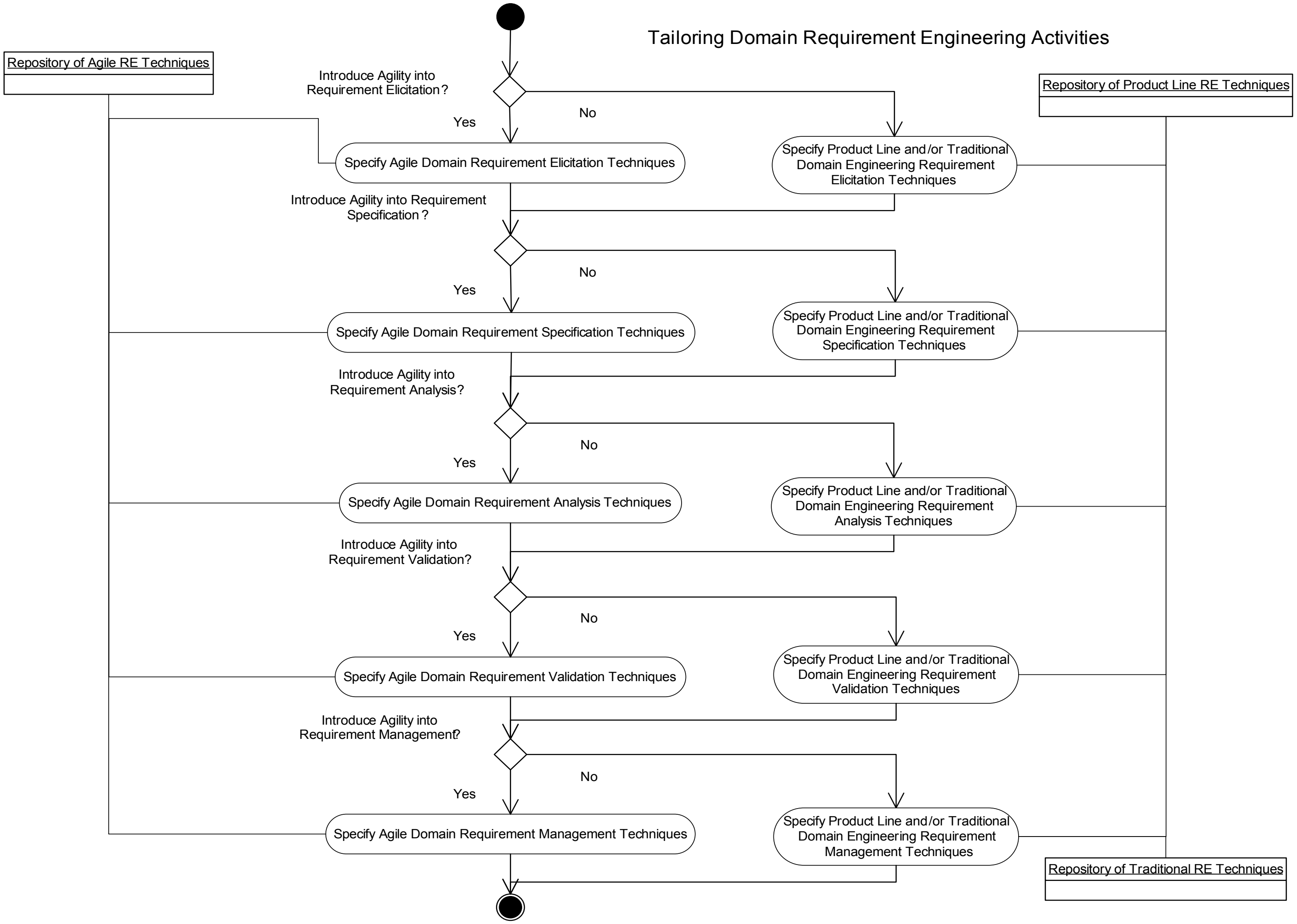

Tailoring Domain Requirement Engineering Activities
Repository of Agile RE Techniques
Repository of Product Line RE Techniques
Repository of Traditional RE Techniques
Introduce Agility into Requirement Elicitation?
Yes
No
Specify Agile Domain Requirement Elicitation Techniques
Specify Product Line and/or Traditional Domain Engineering Requirement Elicitation Techniques
Introduce Agility into Requirement Specification?
Yes
No
Specify Agile Domain Requirement Specification Techniques
Specify Product Line and/or Traditional Domain Engineering Requirement Specification Techniques
Introduce Agility into Requirement Analysis?
Yes
No
Specify Agile Domain Requirement Analysis Techniques
Specify Product Line and/or Traditional Domain Engineering Requirement Analysis Techniques
Introduce Agility into Requirement Validation?
Yes
No
Specify Agile Domain Requirement Validation Techniques
Specify Product Line and/or Traditional Domain Engineering Requirement Validation Techniques
Introduce Agility into Requirement Management?
Yes
No
Specify Agile Domain Requirement Management Techniques
Specify Product Line and/or Traditional Domain Engineering Requirement Management Techniques

Figure 10 Tailoring Product Line RE Activities With Respect To Application Engineering

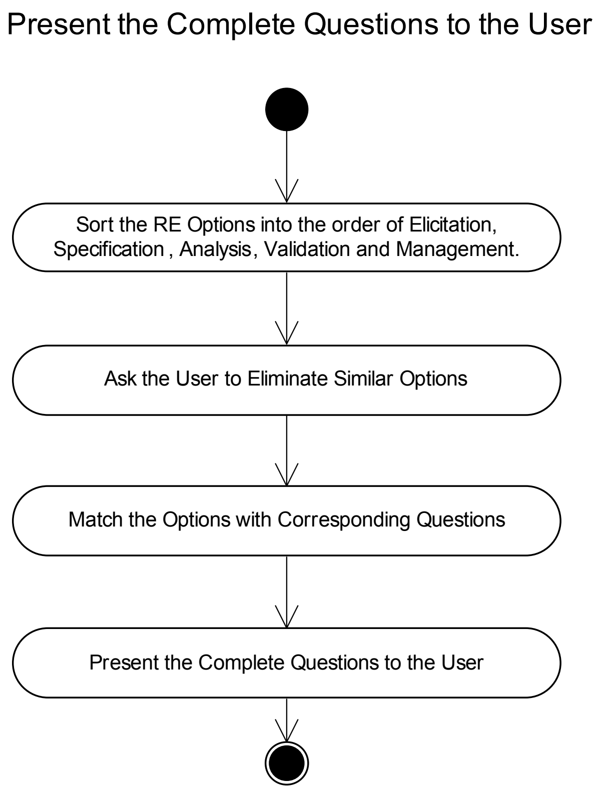

Present the Complete Questions to the User
Sort the RE Options into the order of Elicitation, Specification , Analysis, Validation and Management.
Ask the User to Eliminate Similar Options
Match the Options with Corresponding Questions
Present the Complete Questions to the User

Figure 11 Present Complete Questions to the User

The first four questions are the same for each expert.

1. Is your project a Product Line project? *(Select one option)*
    1. Yes, this is a Product Line Project.
    2. No, this is a Single Product Development Project.

    If your answer is yes, then which one of the following PL approaches would you use (and tailor if necessary) as a base process?

    1. Clements' Approach [5]
    2. Pohl's Approach [15]
    3. Others, please specify.

2. Of the following established Agile Methods, which one would you choose to tailor the development approach (either PL or single product development approach) with? *(Select one or more options)*

    Please use Figure 12 revised from [4] for the relative degree of agility in the approaches.

    1. Extreme Programming (XP) [3]
    2. Agile Documentation [21, 22, 23]
    3. Lean Software Development [22, 24]

    4. Agile Modeling Driven Development (AMDD) [25]
    5. Feature Driven Development (FDD) [14]
    6. Crystal Methods (Crystal Methods) [6]

    7. Scrum (Scrum) [17]
    8. Dynamic System Development Method (DSDM) [12][18]
    9. Adaptive Software Development (ASD) [8]

    10. Others, please specify.

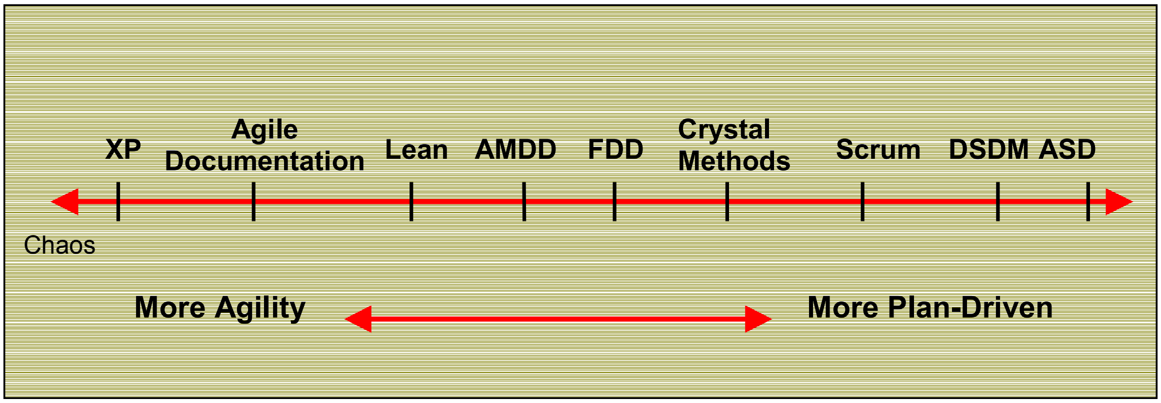


Figure 12 Relative Degree of Agility for Existing Agile Methods (revised from[4])

3. If your answer to Question one was a Single Product Development project, then what RE activity/activities would you ***mainly*** focus on introducing the agility in the requirements engineering process? *(Select one or more options)*
    1. Requirements elicitation
    2. Requirements specification
    3. Requirements analysis

    4. Requirements validation
    5. Requirements management

4. If your answer for Question one is a PL project, then what PL activity would you ***mainly*** focus on introducing the agility in the requirements engineering process? *(Select one or more options)*
    1. For Domain Requirement Engineering
        a) Requirement elicitation
        b) Requirement specification

        c) Requirement analysis
        d) Requirement validation
        e) Requirement management

    2. For Application Requirement Engineering
        a) Requirement elicitation
        b) Requirement specification
        c) Requirement analysis

        d) Requirement validation
        e) Requirement management

The remaining questions are dynamically generated using the expert's responses from Questions 1-4.

Input
The expert's responses for Questions 1-4
Let
W represent the answer to Question 1
X represent the answer to Question 2
Y represent the set of answers to Question 3
Z represent the set of answers to Question 4

Output

Dynamically generated questions.

Three categories of RE techniques were utilized in developing this part of the questionnaire: traditional RE techniques, agile RE techniques (based on established agile methods), and software product line techniques.

The options presented have been extracted from the literature. However, the users have the option to specify any RE techniques. If a technique has proposed a specific approach, then it is presented to the user as a recommended option; however, the user can select any option.

Procedure

1.4.1 Determine Single Product Development Options

Determine Traditional RE options to present for Single Product Development

For each choice (y1, y2, … y5) in Y, if agility is not applied (i.,e., the option is not selected), then display the traditional techniques for the user to choose from:

Entry Condition: w=2 (This is a Single Product Development project.)

If y1 is not selected, then select Traditional Requirement Elicitation Traditional Techniques and let user choose: (Select one or more options)

1) Brainstorming
2) On-site interactive sessions
3) Electronic interactive sessions

4) Focus Groups
5) Review of existing documentation
6) Scenarios

7) Goal modeling
8) Paper prototypes
9) Executable Prototypes

10) Story boards
11) Observation and social analysis
12) Others? please specify

If y2 is not selected, then select Traditional Requirement Specification Traditional Techniques: (Select one or more options)

1) English text (paragraph format, shall statements)
2) RUP Based Approach (Vision Document, Use Case Model, Supplementary (non-functional) requirements)
3) Formal Methods
4) Others, please specify.

If y3 is not selected, then select Traditional Requirement Analysis Traditional Techniques: (Select one or more options)

1) Formal Inspections
2) Peer Reviews
3) Walkthroughs
4) Checklists
5) Formal techniques (type-checking, model checking, theorem proving)
6) Ad-hoc Reviews
7) Others, please specify.

If y4 is not selected, then select Traditional Requirement Validation Traditional Techniques: (Select one or more options)

1) Formal inspections
2) Customer reviews
3) Prototype demonstration
4) Others, please specify.

If y5 is not selected, then select Traditional Requirement Management Traditional Techniques: (Select one or more options)

1) Configuration management
2) Traceability to design artifacts, test cases
3) Others, please specify.

Determine Agile RE options to present for Single Product Development

For each choice (y1, y2, … y5) in Y, if agility is applied (i.e., the option is selected), then display agile techniques based on response x

Entry Condition: p=2 (This is a Single Product Development project.)
If y1 is selected

Case x=1, then select Requirement Elicitation Agile XP techniques: (Select one or more options)

1) On-Site interactive sessions (XP) (Recommended)
2) White Boards (Lean)
3) Model Storming Sessions (AMDD)

4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD)
6) Others, please specify.

Case x=2, then select Requirement Elicitation Agile Documentation techniques: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean)

3) Model Storming Sessions (AMDD)

4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD)
6) Others, please specify.

Case x=3, then select Requirement Elicitation Lean Agile techniques: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean) (Recommended)
3) Model Storming Sessions (AMDD)

4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD)
6) Others, please specify.

Case x=4, then select Requirement Elicitation AMDD Agile techniques: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean)
3) Model Storming Sessions (AMDD) (Recommended)

4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD)
6) Others, please specify.

Case x=5, then select Requirement Elicitation Agile FDD techniques: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean)
3) Model Storming Sessions (AMDD)
4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD)
6) Others, please specify.

Case x=6, then select Requirement Elicitation Agile Crystal techniques: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean)
3) Model Storming Sessions (AMDD)

4) Interactive sessions (Crystal) (Recommended)

5) Focus Group (Scrum, DSDM, ASD)
6) Others, please specify.

Case x=7, then select Requirement Elicitation Agile SCRUM techniques: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean)
3) Model Storming Sessions (AMDD)

4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD) (Recommended)
6) Others, please specify.

Case x=8, then select Requirement Elicitation Agile DSDM techniques: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean)
3) Model Storming Sessions (AMDD)

4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD) (Recommended)
6) Others, please specify.

Case x=9, then select Requirement Elicitation ASD Agile techniques: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean)
3) Model Storming Sessions (AMDD)

4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD) (Recommended)
6) Others, please specify.

Case x=10, then select all and let the user choose: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean)
3) Model Storming Sessions (AMDD)

4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD)
6) Others, please specify.

If y2 is selected

Case x=1, then select Requirement Specification Agile XP techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum) (Recommended)
2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

Case x=2, then Select Requirement Specification Agile Documentation techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum)
2) Feedback-driven documentation (*Agile Documentation*) (Recommended)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

Case x=3, then Select Requirement Specification Lean techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum)
2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean) (Recommended)

4) Story cards, Iteration Feature List (Lean) (Recommended)

5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

Case x=4, then Select Requirement Specification AMDD techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum)
2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD) (Recommended)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

Case x=5, then select Requirement Specification Agile FDD techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum)
2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods) (Recommended)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

Case x=6, then Select Requirement Specification Agile Crystal techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum)

2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods) (Recommended)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

Case x=7, then Select Requirement Specification Agile SCRUM techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum) (Recommended)
2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

Case x=8, then Select Requirement Specification Agile DSDM techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum)
2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods) (Recommended)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

Case x=9, then Select Requirement Specification Agile ASD techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum)
2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD) (Recommended)
8) Others, please specify.

Case x=10, then Select all and let the user choose: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum)
2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

If y3 is selected

Case x=1, then select Requirement Analysis Agile XP technique: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP) (Recommended)
2) Feedback-driven documentation (Agile Documentation)
3) Scrum Meeting (Lean)

4) Inclusive Model (AMDD)
5) Expert Review (FDD, Crystal)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

Case x=2, then select Requirement Analysis Agile Documentation Agile techniques: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation) (Recommended)
3) Inclusive Model (AMDD)

4) Scrum Meeting (Lean)
5) Expert Review (FDD, Crystal)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

Case x=3, then select Requirement Analysis Lean Agile techniques: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation)
3) Scrum Meeting (Lean) (Recommended)

4) Inclusive Model (AMDD)
5) Expert Review (FDD, Crystal)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

Case x=4, then select Requirement Analysis AMDD Agile techniques: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation)
3) Scrum Meeting (Lean)

4) Inclusive Model (AMDD) (Recommended)

5) Expert Review (FDD, Crystal)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

Case x=5, then select Requirement Analysis Agile FDD techniques: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation)
3) Scrum Meeting (Lean)

4) Inclusive Model (AMDD)
5) Expert Review (FDD, Crystal) (Recommended)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

Case x=6, then select Requirement Analysis Agile Crystal techniques: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation)
3) Scrum Meeting (Lean)

4) Inclusive Model (AMDD)
5) Expert Review (FDD, Crystal) (Recommended)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

Case x=7, then v Requirement Analysis Agile SCRUM techniques: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation)
3) Scrum Meeting (Lean)

4) Inclusive Model (AMDD)
5) Expert Review (FDD, Crystal)
6) Sprint Review Meeting (Scrum) (Recommended)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

Case x=8, then select Requirement Analysis Agile DSDM techniques: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation)
3) Scrum Meeting (Lean)

4) Inclusive Model (AMDD)
5) Expert Review (FDD, Crystal)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM) (Recommended)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

Case x=9, then select Requirement Analysis ASD Agile techniques: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation)
3) Scrum Meeting (Lean)

4) Inclusive Model (AMDD)
5) Expert Review (FDD, Crystal)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD) (Recommended)
9) Others, please specify.

Case x=10, then select all and let the user choose: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation)

3) Scrum Meeting (Lean)

4) Inclusive Model (AMDD)
5) Expert Review (FDD, Crystal)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

If y4 is selected

Case x=1, then select Requirement Validation Agile XP techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP) (Recommended)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD)

7) Expert Test (FDD)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD)
9) Others, please specify.

Case x=2, then select Requirement Validation Agile Documentation Agile techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*) (Recommended)
3) Unit testing stories (Agile *Documentation*) (Recommended)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD)

7) Expert Test (FDD)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD)
9) Others, please specify.

Case x=3, then select Requirement Validation Lean Agile techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean) (Recommended)
5) Daily Meeting (Lean) (Recommended)
6) Model Review (AMDD)

7) Expert Test (FDD)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD)
9) Others, please specify.

Case x=4, then select Requirement Validation AMDD Agile techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD) (Recommended)

7) Expert Test (FDD)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD)
9) Others, please specify.

Case x=5, then select Requirement Validation Agile FDD techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD)

7) Expert Test (FDD) (Recommended)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD)
9) Others, please specify.

Case x=6, then select Requirement Validation Agile Crystal techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD)

7) Expert Test (FDD)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD) (Recommended)
9) Others, please specify.

Case x=7, then select Requirement Validation Agile SCRUM techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD)

7) Expert Test (FDD)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD) (Recommended)
9) Others, please specify.

Case x=8, then select Requirement Validation Agile DSDM techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD)

7) Expert Test (FDD)

8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD) (Recommended)
9) Others, please specify.

Case x=9, then select Requirement Validation ASD Agile techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD)

7) Expert Test (FDD)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD) (Recommended)
9) Others, please specify.

Case x=10, then select all and let the user choose: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD)

7) Expert Test (FDD)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD)
9) Others, please specify.

If y5 is selected, then…

Case x=1, then select Requirement Management Agile XP techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP) (Recommended)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

Case x=2, then select Requirement Management Agile Documentation Agile techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

Case x=3, then select Requirement Management Lean Agile techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum) (Recommended)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

Case x=4, then select Requirement Management AMDD Agile techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

Case x=5, then select Requirement Management Agile FDD techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

Case x=6, then select Requirement Management Agile Crystal techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods) (Recommended)
3) Daily Scrum Meeting (Lean, Scrum)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

Case x=7, then select Requirement Management Agile SCRUM techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum) (Recommended)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

Case x=8, then select Requirement Management Agile DSDM techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum)

4) High Level Requirements Baselines (DSDM) (Recommended)
5) Others, please specify.

Case x=9, then select Requirement Management ASD Agile techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

Case x=10, then select all and let the user choose: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

1.4.2 Determine Product Line Development Options

Determine RE options to present for introducing Agility into Product Line Domain Engineering,

Entry Condition: w=1 (This is a Product Line Project.)

Select PL Domain Requirement Elicitation Techniques (Multiple Choices)

If the user chooses to introduce agility into Domain Requirement Elicitation, then let the user select Agile Requirement Elicitation Techniques: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean)
3) Model Storming Sessions (AMDD)

4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD)
6) Others, please specify.

Else let the user select traditional PL Requirement Elicitation Techniques: (Select one or more options)

1) Structured Group Discussion between Stakeholders, Domain Experts and RE Engineers
2) Expert reviews of existing products & competitors’ products
3) Others, please specify.

Select PL Domain Requirement Specification Techniques (Multiple Choices)

If the user chooses to introduce agility into Domain Requirement Specification, then let the user select Agile Requirement Specification Techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum)
2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

Else let the user select traditional PL Requirement Specification Techniques: (Select one or more options)

1) RUP Based Approach (Vision Document, Use Case Model, Supplementary (non-functional) requirements) (UML, etc)
2) Domain Specific Language
3) Informal documentation

4) Semi-formal documentation
5) Formal documentation
6) Others, please specify.

Select PL Domain Requirement Analysis Techniques (Multiple Choices)

If the user chooses to introduce agility into Domain Requirement Elicitation, then let the user select Agile Requirement Analysis Techniques: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation)
3) Scrum Meeting (Lean)

4) Inclusive Model (AMDD)
5) Expert Review (FDD, Crystal)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

Else let the user select PL Requirement Analysis Techniques: (Select one or more options)

1) Application-Requirements Matrix,
2) Checklist-Based Analysis
3) Others, please specify.

Select PL Domain Requirement Validation Techniques (Multiple Choice)

If the user chooses to introduce agility into Domain Requirement Validation, then let the user select Agile Requirement Validation Techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD)

7) Expert Test (FDD)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD)
9) Others, please specify.

Else let the user select traditional PL Requirement Validation Techniques: (Select one or more options)

1) Formal Requirements Review (group session to verify the requirements)
2) Informal Requirement Review (Discussion between customers and RE engineers)
3) Prototyping

4) Requirements Testing
Others, please specify.

Select PL Domain Requirement Management Techniques (Multiple Choices)

If the user chooses to introduce agility into Domain Requirement Management, then let the user select Agile Requirement Management Techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

Else let the user select PL Requirement Management Techniques: (Select one or more options)

1) Change Control Process
2) Traceability Management
3) Others, please specify.

Determine RE options to present for introducing Agility into Product Line Application Engineering,

Entry Condition: w=1 (This is a Product Line Project.)

Select PL Application Requirement Elicitation Techniques (Multiple Choices)

If the user chooses to introduce agility into Domain Requirement Elicitation, then let the user select Agile Requirement Elicitation Techniques: (Select one or more options)

1) On-Site interactive sessions (XP)
2) White Boards (Lean)
3) Model Storming Sessions (AMDD)

4) Interactive sessions (Crystal)
5) Focus Group (Scrum, DSDM, ASD)
6) Others, please specify.

Else let the user select traditional PL Requirement Elicitation Techniques: (Select one or more options)

1) Structured Group Discussion between Stakeholders, Domain Experts and RE Engineers
2) Expert reviews of existing products & competitors’ products
3) Others, please specify.

Select PL Application Requirement Specification Techniques (Multiple Choices)

If the user chooses to introduce agility into Domain Requirement Specification, then let the user select ask the user what Agile Requirement Specification Techniques: (Select one or more options)

1) Story Cards, Task Lists (XP, Scrum)
2) Feedback-driven documentation (*Agile Documentation*)
3) Minimum Marketable Features (Lean)

4) Story cards, Iteration Feature List (Lean)
5) Usage Model (Essential use cases, features, user stories) (AMDD)
6) Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods)

7) High Level Requirement Specification (for accommodating change tolerance) (ASD)
8) Others, please specify.

Else let the user select traditional PL Requirement Specification Techniques: (Select one or more options)

1) RUP Based Approach (Vision Document, Use Case Model, Supplementary (non-functional) requirements) (UML, etc)
2) Domain Specific Language
3) Informal documentation (Optional)

4) Semi-formal documentation (Optional)
5) Formal documentation (Optional)
6) Others, please specify.

Select PL Application Requirement Analysis Techniques (Multiple Choices)

If the user chooses to introduce agility into Domain Requirement Analysis, then let the user select Agile Requirement Analysis Techniques: (Select one or more options)

1) Lightweight and lazy requirements analysis based on User Stories and incremental requirements (XP)
2) Feedback-driven documentation (Agile Documentation)
3) Scrum Meeting (Lean)

4) Inclusive Model (AMDD)
5) Expert Review (FDD, Crystal)
6) Sprint Review Meeting (Scrum)

7) System Prototyping (DSDM)
8) Domain Expert Reviews and Constant Prototyping (ASD)
9) Others, please specify.

Else let the user select traditional PL Requirement Analysis Techniques: (Select one or more options)

1) Application-Requirements Matrix,
2) Checklist-Based Analysis
3) Others, please specify.

Select PL Application Requirement Validation Techniques (Multiple Choices)

If the user chooses to introduce agility into Domain Requirement Validation, then let the user select Agile Requirement Validation Techniques: (Select one or more options)

1) On-site Customer (User Acceptance Tests) (XP)
2) FitNesse Stories (Acceptance tests as stories) (Agile *Documentation*)
3) Unit testing stories (Agile *Documentation*)

4) Feedback (Lean)
5) Daily Meeting (Lean)
6) Model Review (AMDD)

7) Expert Test (FDD)
8) User Reviewing (Crystal Methods, Scrum, DSDM, ASD)
9) Others, please specify.

Else let the user select traditional PL Requirement Validation Techniques: (Select one or more options)

1) Formal Requirements Review (group session to verify the requirements)
2) Informal Requirement Review (Discussion between customers and RE engineers)
3) Prototyping

4) Requirements Testing
5) Others, please specify.

Select PL Application Requirement Management Techniques (Multiple Choices)

If the user chooses to introduce agility into Domain Requirement Management, then let the user select Agile Requirement Management Techniques: (Select one or more options)

1) Maintaining minimal and extensible requirements on User Story Cards (XP)
2) Requirement File (Crystal Methods)
3) Daily Scrum Meeting (Lean, Scrum)

4) High Level Requirements Baselines (DSDM)
5) Others, please specify.

Else let the user select traditional PL Requirement Management Techniques: (Select one or more options)

1) Change Control Process
2) Traceability Management
3) Others, please specify.

4. Conclusions

The on-line questionnaire on Agile Product Line Requirement Engineering is used to elicit expert options on the appropriateness about a specific RE activity satisfying an agile product line project configuration. The questionnaire contains two parts. The first is used to elicit project configuration as well as expert background and the second part is used to elicit the expert opinions on Agile Product Line Requirement Engineering activities with respect to their project configurations, i.e., scenarios. It is now available online at [1]. The questionnaire will help to elicit raw data that will later on be manually reviewed for the input into (or configuration for) our expert engine. The expert engine in turn will produce suggested RE activities according to various projects configurations based on the conditional probabilities calculated on the analyzed data.

References

[1] Hurtado, J. A., Bastarrica, M. C., *Implementing CMMI using a Combination of Agile Methods,* Clei Electronic Journal, Vol. 9, June 2006

[2] Pekka Abrahamsson, Outi Salo, Jussi Ronkainen, Juhani Warsta, *Agile software development methods. Review and analysis,* VTT Publications 478, pp. 107 , Espoo, 2002

[3] Kent Beck, *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 1999.

[4] Barry Boehm, *Get Ready for Agile Methods with Care,* Software Development, January 2002, pp. 64-69.

[5] Paul Clements, Linda Northrop, *Software Product Lines: Practices and Patterns.* Addison-Wesley, 2005.

[6] Alistair Cockburn, *Agile Software Development*. Adisson-Wesley, 2002.

[7] Neeraj K. Gupta, Lalita Jategaonkar Jagadeesan, Eleftherios E. Koutsofios, David M. Weiss, *Auditdraw: Generating Audits the FAST Way"* In Proceedings of the IEEE International Symposium on Requirements Engineering, January 1997.

[8] James A. Highsmith, *Adaptive Software Development*. Dorset House Publishing, 2000

[9] Feng, K., Lempert M., Tang, Y., Tian, K., Cooper, K., Franch, X., *Defining Project Scenarios for the Agile Requirements Engineering Product-line Development*

Questionnaire, Technical report, co-published as UTDCS-21-07 (UTD) and LSI-07-14-R (UPC), 2007.

[10] Linda Levine, *Reflections on Software Agility and Agile Methods: Challenges, Dilemmas, and the Way Ahead,* Key Notes, Software Engineering Institute, Carnegie Mellon University, May, 2005.

[11] Oppenheim, O.N., *Questionnaire Design And Attitude Measurement.* NewYork : Basic Books Inc, 1966.

[12] David Norfolk, *Understanding DSDM*, *PC Network Advisor*, issue 67, pp. 15 - 18

[13] Frauke Paetsch, Armin Eberlein, Frank Maurer, *Requirements Engineering and Agile Software Development*, in Proceedings of the Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003

[14] Stephen R. Palmer and John M. Felsing, *A Practical Guide to Feature-Driven Development*. Prentice Hall, 2002.

[15] Klaus Pohl, Gunter Bockle, Frank van der Linden, *Software Product Lines Engineering: Foundations, Principles, and Techniques.* Springer-Verlag Berlin Heidelberg, 2005.

[16] Berdie, Doug R., Anderson, John F., Niebuhr, Marsha A, *Questionnaires: Design and Use*, 2nd edition. The Scarecrow Press Inc, 1986.

[17] Ken Schwaber, *The Scrum development process,* in *Business Object Design and Implementation:* OOPSLA'95 Workshop Proceedings. Springer-Verlag, 1997.

[18] Jennifer Stapleton, *Dynamic Systems Development Method - The method in practice*. Addison-Wesley, 1997.

[19] Agile Product Line Engineering Questionnaire, available http://129.110.92.41/Default.aspx

[20] Manifesto for Agile Software Development, available at: http://www.agilemanifesto.org.

[21] Andreas Rueping, *Agile Documentation: A Pattern Guide to Producing Lightweight Documents for Software Project.* John Wiley & Sons, 2003

[22] Agile/Lean Documentation: Strategies for Agile Software Development. http://www.agilemodeling.com/essays/agileDocumentation.htm

[23] Grig Gheorghiu, Avamar, *Agile Documentation (using tests as documentation),* PyCon 2006, Feb. 26, Addison, TX

[24] Mary and Tom Poppendieck, *Lean Software Development - An Agile Toolkit for Software Development Managers,* Addison Wesley, 2003

[25] Agile Model Driven Development (AMDD): The Key to Scaling Agile Software Development. http://www.agilemodeling.com/essays/agileDocumentation.htm

Appendix A. Designing a “Good” Questionnaire

This Appendix provides a summary of points presented in [11], [16] on how to design a “good” questionnaire. In particular, the structure and organization of a questionnaire are the focus of this Appendix.

Questionnaire Structure and Organization

The following points are key elements in the design of a “good” questionnaire:

1. The title of the questionnaire should be highlighted on the first page.

2. The respondent should be given instructions on how to complete the questionnaire.

3. The questionnaire should be grouped into logically coherent sections. Therefore, the questions dealing with a specific topic should go together and the questions that make use of the same response options should stay together.

4. If possible, then there should be smooth transitions between sections, avoiding the appearance of unrelated questions in the beginning and the end of each individual section.

5. The questionnaire should begin with a few easy (non-threatening) and interesting questions that introduce the respondents to our questionnaire, because difficult or “dull” questions may make the questions seem overwhelming to the respondents and hence reduce the likelihood of their completing it.

6. Important questions should be presented in the middle of our questionnaire.

7. The end of the questionnaire should be clearly marked so that respondents will not become confused while they are completing it.

8. The respondent should be able to answer the questionnaire in 30 - 45 minutes. Some studies have shown that the length of a questionnaire does not necessarily

affect response. More important than length is question content. A subject is more likely to respond if they are involved and interested in the research topic.

9. The number of answers for a question should be more than 2 and less than 10. The labels of such answers could be numeric (1, 2.) or alphabetic (a, b).

10. For questions that have a long checklist of responses, a line should be skipped after every third or fourth item to help the respondent focus on the appropriate answers.

11. The respondents should not be asked to order (i.e., rank) a series of more than five items.

12. The questionnaire should not put too many questions on one page.

Appendix B. An Analysis of Requirements Engineering Activities

The 12 principles of agile methods[1] are ranked in terms of how related the principles are to typical activities in a RE process, including elicitation, specification, analysis, validation, and change management. A simple three choice ranking scheme is used:

- highly related to an RE activity
- moderately related to an RE activity
- less related to an RE activity

The rationale for each ranking is provided.

Principle 1. "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software."

Ranking.
This is highly related to the elicitation, specification, analysis, and validation activities
This is less related to the change management activities

Rationale.
The RE activities focus on understanding what business goals a system needs to accomplish (early requirements), what the stakeholders' need a system to do (functional requirements) and how the system needs to behave (non-functional requirements). This is accomplished through the elicitation, specification, analysis, and validation of requirements; consequently we rank the principle as highly related to these. If the requirements are met in the delivered code, then the customer is expected to be more satisfied.

The principle is ranked as less related to the change management activity. It is an important software quality assurance aspect that keeps the requirements up to date with respect to on-going changes. However, the principle is much less related to this

[1] Principles behind the Agile Manifesto, available at: www.agilemanifesto.org

activity than to the elicitation, specification, and validation activities, as the focus is not on delivering change management reports, but on delivering the software (i.e., the code).

Notes. The specific scheduling and frequency of delivering the software is more related to the project management aspects of the development, not the engineering activities.

Principle 2. "Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage."

Ranking.
This is highly related to the elicitation, specification, analysis, validation, and change management activities.

Rationale.
Supporting changes in the requirements relies on eliciting and understanding the change, specifying the change, analyzing the change, and validating the change is correct with the stakeholders. The management of changes is also highly related to this principle.

Principle 3. "Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale."

Principle 3 Ranking.
This is highly related to the elicitation, specification, analysis, validation, and change management activities.

Rationale.
The ranking reflects the need to rapidly elicit, specify, analyze, validate, and manage changes in requirements for each deliverable of working software. If the requirements for each deliverable are not understood, then the customer is not likely to receive

software that does what they need. The requirement engineering process needs to occur iteratively.

Notes. The specific scheduling and frequency of delivering the software is more related to the project management aspects of the development, not the engineering activities.

Principle 4. "Business people and developers must work together daily throughout the project."

Principle 4 Ranking.
This is highly related to the elicitation and validation activities.
This is moderately related to the specification, analysis, and change management activities.

Rationale.
The ranking reflects the need to interactively elicit and validate the requirements with the stakeholders for each deliverable of working software, perhaps on a daily basis. This is related to the need to accomplish this work rapidly and support the delivery of software in a short amount of time (Principle 3).

The principle is ranked as moderately related to the specification activity, as the requirements need to be specified (in some form) to allow the stakeholders to analyze and validate them. The activity is a supporting, not primary, activity with respect to this principle.

The principle is ranked as moderately related to the analysis activity, as the requirements need to be analyzed for internal consistency, correctness, clarify, etc. of the requirements specification, but is primarily the responsibility of the developers (e.g., requirements engineers). The activity is a supporting, not primary, activity with respect to this principle.

The principle is ranked moderately related to the change management activity. It is an important software quality assurance aspect that keeps the requirements up to date with respect to on-going changes. If the requirements engineers are working together daily, then the changes to the requirements can be changing rapidly and need to be managed (in some way).

Principle 5. “Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.”

Principle 5 Ranking.
This is less related to the elicitation, specification, analysis, validation, and change management activities.

Rationale.
The principle is ranked as less related to the RE activities because it focuses on management issues.

Principle 6. “The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.“

Principle 6 Ranking.
This is highly related to the elicitation, specification, analysis, validation, and change management activities.

Rationale.
The principle is ranked as highly related to the RE activities because it focuses on using verbal, rather than written, communication in a project. Applied to the RE process, the elicitation, specification, analysis, validation, and change management of requirements would occur in face-to-face meetings (not via websites, paper documents, e-mail, etc.).

Principle 7. "Working software is the primary measure of progress. "

Principle 7 Ranking.

This is less related to the elicitation, specification, analysis, validation, and change management activities.

Rationale.

The principle is ranked as being less related to the RE activities because it focuses on management issues such as monitoring and controlling the progress of a project.

Principle 8. "Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely. "

Principle 8 Ranking.

This is less related to the elicitation, specification, analysis, validation, and change management activities.

Rationale.

The principle is ranked as being less related to the RE activities because it focuses on management issues such as workload assignments.

Principle 9. "Continuous attention to technical excellence and good design enhances agility."

Principle 9 Ranking.

This is less related to the elicitation, specification, analysis, validation, and change management activities.

Rationale.

The principle is ranked as less related as it focuses on creating a culture of excellence for a team. This principle is more related to management and leadership issues.

Principle 10. “Simplicity--the art of maximizing the amount of work not done--is essential.”

Principle 10 Ranking.
This is highly related to the elicitation, specification, analysis, validation, and change management activities.

Rationale.
The principle is ranked as highly related to the RE activities because it focuses on streamlining the elicitation, specification, analysis, validation, and change management activities. For example, in the specification activity, the requirements could be specified using brief notes taken during the elicitation activity, rather than a collection of fully defined use cases.

Principle 11. The best architectures, requirements, and designs emerge from self-organizing teams.

Principle 11 Ranking.
This is less related to the elicitation, specification, analysis, validation, and change management activities.

Rationale.
The principle is ranked as being less related to the RE activities because it focuses on team organization and management issues.

Principle 12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Principle 12 Ranking.
This is less related to the elicitation, specification, analysis, validation, and change management activities.

Rationale.

The principle is ranked as being less related to the RE activities because it focuses on process improvement and management issues.

Discussion

The ranking is summarized in Table 1.

[26]	**Agile Principle**											
RE Activity	1	2	3	4	5	6	7	8	9	10	11	12
Elicitation	H	H	H	H	L	H	L	L	L	H	L	L
Specification	H	H	H	M	L	H	L	L	L	H	L	L
Analysis	H	H	H	M	L	H	L	L	L	H	L	L
Validation	H	H	H	H	L	H	L	L	L	H	L	L
Change Mgmt.	L	H	H	M	L	H	L	L	L	H	L	L

Table 1. Evaluation of the 12 Agile Principles With Respect to RE Process Activities

Given the ranking proposed, the following principles of agile methods are the most closely related to the RE process:

> Principle 1. "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software."
>
> Principle 2. "Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage."
>
> Principle 3. "Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale."
>
> Principle 4. "Business people and developers must work together daily throughout the project."
>
> Principle 6. "The most efficient and effective method of conveying information to and within a development team is face-to-face conversation."
>
> Principle 10. "Simplicity--the art of maximizing the amount of work not done--is essential."

These principles can be used to frame the problem of how to choose a RE process with a degree of agility that is suitable for a project. For example, using Principle 6 some questions to ask about a project may include:

1. To what level do you anticipate the customer stakeholders needing or wanting to be involved in the requirements engineering process in face-to-face meetings to discuss their business goals, elicit software requirements, validate the requirements, perhaps via software deliverables, demonstrations, etc.)?
   - ▪ Low (meet every six months)
   - ▪ Limited involvement (meet each business quarter)
   - ▪ Moderate involvement (meet each month)
   - ▪ Highly involved (meet each week or more)

2 For each of the following, how many geographic locations (different facilities, different cities, different countries, etc.) are likely to be involved for the proposed project?

Customer stakeholders, such as:

- o end users
- o decision makers (business needs, budget, etc.)

- ▪ 1 location
- ▪ 2-4 locations
- ▪ >5 locations

Development stakeholders, such as:

- o marketing group
- o requirements engineers or business analysts
- o architects
- o designers
- o implementation
- o testing

- ▪ 1 location
- ▪ 2-4 locations
- ▪ >5 locations

The higher the number of stakeholders' locations and the less often the customers are needing or wanting to be involved in face-to-face meetings, the less agile the RE process needs to be.

The next step in this research is to collect expertise from the community using a questionnaire. The expertise collected will be captured in an intelligent tool, that can be used to guide choosing a RE process with respect to the degree of agility and the use of product line engineering techniques.

Appendix C. Summary of Traditional, Agile and Product Line RE Techniques

Traditional RE Techniques

Requirement Elicitation Techniques	***Requirement Specification Techniques***	***Requirement Analysis Techniques***	***Requirement Validation Techniques***	***Requirement Management Techniques***
Requirements Workshops *Interviewing* *Brainstorming and Idea Reduction* *Surveys, Questionnaires and Data Mining* *Market Analysis* *Throw-away prototyping* *Electronic Interviews* *Observation*	*English text (paragraph format, shall statements)* *RUP Based Approach (Vision Document, Use Case Model, Supplementary (non-functional) requirements)* *Formal Methods*	*Formal Inspections* *Peer Reviews* *Walkthroughs* *Checklists* *Formal techniques (type-checking, model checking, theorem proving)* *ad-hoc review*	*Formal inspections* *Customer reviews* *Prototype demonstration*	*Configuration management* *Requirement Traceability to design artifacts, test cases*

Agile Methods RE Techniques

Agile Requirement Elicitation Techniques	***Agile Requirement Specification Techniques***	***Agile Requirement Analysis Techniques***	***Agile Requirement Validation Techniques***	***Agile Requirement Management Techniques***
On-Site interactive sessions (XP) Interactive sessions (Crystal) Focus Group (Scrum, DSDM, ASD) Model Storming Sessions (AMDD) White Boards (Lean)	*Story Cards, Task Lists (XP,SCRUM)* *Functional Requirements Specifications (Use Cases, mainly) (FDD, DSDM, Crystal Methods)* *High Level Requirement Specification (ASD)* *Usage Model (Essential use cases, features, user stories) (AMDD)*	*Lightweight and lazy requirements analysis based on User Stories and incremental requirements(XP)* *Expert Review (FDD, Crystal, Scrum)* *System Prototyping (DSDM)* *Sprint Review Meeting (Scrum)* *Domain Expert Reviews and Constant*	*On-site Customer (User Acceptance Tests) (XP)* *Expert Test (FDD)* *User Reviewing (Crystal Methods, DSDM, Scrum, ASD)* *Model Review (AMDD)* *Feedback (Lean)* *Daily Meeting (Lean)* *FitNesse Stories*	*Maintaining minimal and extensible requirements on User Story Cards(XP)* *Requirement File (Crystal Methods)* *Daily Scrum Meeting (Scrum)* *High Level Requirements Baselines (DSDM)*

	Minimum Marketable Features (Lean) *Story cards, Iteration Feature List (Lean)* *Feedback-driven documentation (Agile Documentation)*	*Prototyping (ASD)* *Inclusive Model (AMDD)* *Scrum Meeting (Lean)* *Feedback-driven documentation (Agile Documentation)*	*(Acceptance tests as stories) (Agile Documentation)* *Unit testing stories (Agile Documentation)*	

PL Approaches RE Techniques

PL Requirement Elicitation Techniques	***PL Requirement Specification Techniques***	***PL Requirement Analysis Techniques***	***PL Requirement Validation Techniques***	***PL Requirement Management Techniques***
Structured Group Discussion between Stakeholders, Domain Experts and RE Engineers , *Expert reviews of existing products & competitors' products*	*RUP Based Approach (Vision Document, Use Case Model, Supplementary (non-functional) requirements) (UML, etc)* *Domain Specific Language* *Informal documentation (Optional)* *Semi-formal documentation (Optional)* *Formal documentation (Optional)*	*Application-Requirements Matrix,* *Checklist-Based Analysis*	*Formal Requirements Review (group session to verify the requirements)* *Informal Requirement Review (Discussion between customers and RE engineers)* *Prototyping* *Requirements Testing*	*Change Control Process* *Traceability Management*