CHAPTER IV

ELICITING SECURITY REQUIREMENTS USING THE PROPOSED SECURE APPRECIATIVE INQUIRY TECHNIQUE

4.1 Introduction

Two security requirements elicitation methods and the best practices have been selected and filtered. Security Quality Requirements Engineering (SQUARE) and Comprehensive, Lightweight Application Security Process (CLASP) as discussed in Table 8, Chapter II, were compared with other most common approaches and best practices. According to the factors related to requirement elicitation and security requirements; this enhancement of appreciative inquiry method was proven to be successful based on the case study that elicits functional requirements. The method finds extra and unique requirements, and finally uses the AI method (See Appendix A).

Now, it is essential to decide, by mapping the steps of these two methods that they are suitable to be employed with AI phases (Discovery, Dream, Design and Destiny), thus they produce the proposed Secure Appreciative Inquiry Technique (SAIT).Through the case study conducted adopting the new proposed technique integrating the two approaches (CLASP and SQUARE) with the AI method evaluate the performance of the proposed SAIT that was carried out.

4.2 Benefits of Integrating Security Requirements into Software Requirements

As discussed in the literature review, SQUARE, CLASP and AI carry some steps, such as, the dimensions and the main points, where each one of them contains one or more sub-steps, which contribute towards eliciting security requirements for

(SQUARE and CLASP), and general software requirements using AI. This elicitation steps are necessary to address the security state of an SDLC artifact before (at an early stage especially in the requirement phase) and after the software development process. Moreover, this could help software developers in allocating further resources to increase the security and decrease the vulnerabilities in any software. The elicited information about vulnerabilities and errors is important because if an error leads to vulnerability was not corrected in an early phase, the cost of correcting it might increase tenfold with every additional development phase (Khan, 2008).

There are many benefits of integrating the elicitation method for software requirements (AI) and security requirements (SQUARE and CLASP): Firstly, simultaneous, use of methods to elicit software requirements and security requirements. Secondly, the efficiency of AI in eliciting security requirements is evaluated. AI has been proven to be successful in several case studies, in eliciting functional/business requirements and finding extra and unique requirements. Finally, to integrate, the best practices from the SQUARE method and CLASP are identified to be embedded with the AI method, ultimately for eliciting software requirements and security requirements.

4.3 Secure Appreciative Inquiry Technique: Embedding Square, Clasp With AI

The proposed technique has been integrated according to the type of process or step in AI, SQUARE and CLASP. For example, the steps which focus on identifying software requirements in the AI and security requirements in SQUARE were merged in the same phase in Secure Appreciative Inquiry Technique (SAIT) besides integrating the steps involved in system/software requirements from CLASP and AI with all justifications. The integration process must also apply to all the remaining steps. The following section will explain the steps by development of the proposed SAIT according to the AI phases.

4.3.1. Discovery Phase of proposed SAIT

This section presents the integrating steps in CLASP, SQUARE with AI in the discovery phase of the proposed Secure Appreciative Inquiry Technique as shown in table 10.

TABLE 10: Proposed discovery phase for SAIT

Existing Techniques			**New Technique**	
AI (Discovery Phase)	**SQUARE**	**CLASP**	**SAIT (Discovery Phase)**	**Who Involved**
'what is': Cooperrider questions • identifies the existing situation of the old system (**Software Requirements**), System description • explain experience and skills • current system feedback	• agree on definitions (**Security Requirements**) • security goals, which are achieved by the current system	• identify global security policy • identify resources and trust boundaries • specify operational environment	1. identifies the existing situation of the old system (**Software Requirements**); system description (**AI**) 2. agree on definitions (**Security Requirements**) (**SQUARE**) 3. current system feedback **(AI)** 4. identify security goals, which are achieved by the current system **(SQUARE)** 5. identify global security policy **(CLASP)** 6. identify resources and trust boundaries **(CLASP)** 7. explain experience and skills **(AI)** 8. specify operational environment **(CLASP)**	Stakeholders, Users and Developers

a) Discussion and justifications of SQUARE

Discovery phase focuses on the system's boundaries, system identifiers and it entails understanding the current system (old system). Therefore, in the first step of the AI method, the developers should gain system descriptions, and understand the existing system situation, in terms of the system requirements, the next integrated SQUARE step, which is "***Agree on Definitions***" with the initial step in the first phase of the AI method, because, the definitions step had focused on identifying the boundaries of the system in terms of the security requirements.

The third step of the AI method is "***Current System Feedback***" which focuses on studying the current system in terms of software requirements such as: what are the software requirements in the current system, and which one of them is needed in the future system. Then the SQUARE step is integrated, in a process termed "***Identify Security Goals***", because this step is consistent with the AI step, in terms of the software requirements to the current system; i.e. in this step, AI finds the software requirements in the current system and SQUARE finds security requirements in the current system. Therefore, it is appropriate to integrate it with this step.

b) Discussion and justifications of CLASP

The discovery phase focuses on the system boundaries, system identifiers and understanding the current system (old system). In this phase, a new step has been added, which is step 5, named as "***Identify Global Security Policy***" CLASP activity, because, the mission of this phase is to identify the size, boundaries and situation of the system in terms of the software requirements in the old system. The mission of the aforementioned CLASP activity is to identify all the general security policies related to the current system i.e. discover the current system in terms of the security policy. It is thus, suitable to integrate this activity with this phase, to discover the security policy in conjunction with the discovery of software requirements.

Furthermore, a new step (step 6) has been added, called "***Identify Resources and Trust Boundaries***" CLASP activity, because, this activity, as well as step 4 in this

phase, is part of the best original practices of CLASP related to "***Capture Security Requirements***" (see table 8, chapter II). In the investigation phase; this activity investigates current system resources (data accessed or provided by the software), and trusted areas.

"***Specify Operational Environment***" activity of CLASP, as well as steps 5 and 6 are integrated with the first phase of AI, at the investigation stage; this activity analyzes and specifies the level of protection that suits the future system, restrictions to which it is subjected, and the prospective effect on its reputation that should be exploited by the application.

4.3.2. Dream Phase of proposed SAIT

This section presents the integrating of steps in CLASP, SQUARE with AI in the dream phase of the proposed Secure Appreciative Inquiry Technique as shown in table 11.

TABLE 11: Proposed dream phase for SAIT

Existing Techniques			**New Technique**	
AI (Dream Phase)	**SQUARE**	**CLASP**	**SAIT (Dream Phase)**	**Who Involved**
'might be': • What are the new business requirements they dream to achieve	• Identify security needs; what are the efforts to protect the system assets (Future System)	Nil	1. What are the new business requirements they dream to achieve **(AI)** 2. Identify security needs which are related to the future system **(SQUARE)**	Stakeholders, Developers and Security Experts

a) Discussion and justifications of SQUARE

Dream phase: focuses on what the team (stakeholders, developers and security experts) thinks for the future system, but, before that, they should identify the needs, and in this step, AI identifies the system needs (software requirements) to see what might be in their dreams. On the other hand, in the dream phase, SQUARE has a step called "***Identify Security Needs***"; the benefit of this step in the SQUARE method is that, it highlights all the security needs, to protect the system at the end of the day. This point leads us to think deeply that, AI identifies all software needs (requirements) to visualize the future system. All previous contexts lead us to use the SQUARE step, called "***Identify Security Needs***", whereby to determine all security needs, need to dream about the requirements. Ultimately, integrating the approaches enable us to gain software requirements and security requirements, this is called (SAIT).

b) Discussion and justifications of CLASP

Any activity of CLASP best practices has not been integrated in the "Dream Phase" because during the analytical phase of this study the following is established:

i. This method covers the whole SDLC and not only focusing on the requirement phase.
ii. All CLASP best practices activities are practical (do not have the stop point, to think about what might be proposed in the software, during the requirement phase).
iii. The dream phase is available in the CLASP method after the implementation phase, which is not covered in the AI method (all AI phases during the requirement phase).

4.3.3. Design Phase of proposed SAIT

This section presents the integrating steps in CLASP, SQUARE with AI in the design phase of the proposed Secure Appreciative Inquiry Technique, which are shown in table 12.

TABLE 12: Proposed design phase for SAIT

Existing Techniques			New Technique	
AI (Design Phase)	**SQUARE**	**CLASP**	**SAIT (Design Phase)**	**Who Involved**
'what should be': • Examine the feasible size of the system. Based on the recognized strengths highlighted by Discovery and Dream Phases • External point of view	• Develop artifacts to support security requirements' definition • Draw Use Case, Misuse Case Diagrams • **Elicit security requirements** • Categorize security requirements • **Perform risk assessment to the elicited security requirements by using external tools (Table 9, p.55) to quantify, categorize and prioritize security requirements to the whole system**	• Identify user roles and resource capabilities • Apply security principles to design • Detail misuse cases • Identify attack surface and annotate class designs with security properties • Perform security analysis of system requirements and design (threat modeling) • Verify security attributes of resources, Perform source-level security review • Identify, implement, and perform security tests • Manage the security issue disclosure process • Address reported security issues	1. Examine the feasible size of the system. Based on the recognized strengths highlighted by Discovery and Dream Phases **(AI)** 2. Develop artifacts to support security requirements' definition **(SQUARE)** 3. Identify user roles and resource capabilities **(CLASP)** 4. Apply security principles to design **(CLASP)** 5. Perform the security analysis of system requirements and design (threat modeling) **(CLASP)** 6. Draw Use Case, Misuse Case Diagrams (Identify attack surface and annotate class designs with security properties) **(SQUARE) & (CLASP)** 7. External point of view **(AI)** 8. Verify security attributes of resources	Users, Developers and Security Experts

		• Document security-relevant requirements • Integrate the security analysis into source management process • **Monitor security metrics**	**(CLASP)** 9. **Elicit security requirement (SQUARE)** 10. Categorize security requirements (Perform source-level security review) **(SQUARE) & (CLASP)** 11. **Perform risk assessment to the elicited security requirements by using external tools (Table 9, p.55) to quantify, categorize and prioritize security requirements to the whole system**; (Identify, implement, and perform security tests) **(SQUARE) & (CLASP)** 12. Manage security issue disclosure process **(CLASP)** 13. Address reported security issues **(CLASP)** 14. Document security-relevant requirements **(CLASP)** 15. Integrate the security analysis into the source management process **(CLASP)** 16. **Monitor security metrics (CLASP)**	

a) Discussion and justifications of SQUARE

Design phase: it focuses on how the system should look in the future. In the first step of this phase, the users and developers examine the feasible size of the system, that is identified in the second phase (Dream phase). Furthermore, in the dream phase the security experts have identified the security needs in the future's system, so they should examine the security needs, which are identified in the dream phase. In the SQUARE approach the step is called, "***Develop Artifacts to Support Security Requirements Definition***", it is used to examine what have been identified before; based on the needed artifacts: scenarios, misuse cases, models, templates, forms.

In the second step of the design phase, AI elicits software requirements using the diagrams, which are performed in the first step of this phase; in the same step elicitation is crucial for the security requirements using the diagrams, which are performed by the step of the SQUARE method. Furthermore, extra steps also need to be conducted in the SQUARE method called as, "***Categorize Security Requirements as Different Levels (system, software, etc.), and Classify Whether They are Requirements or Other Kinds of Constraints***", this step is required to categorize the security requirements, and each group concerned as asset i.e. every asset has some specific requirements to be achieved; every software requirement has its own security requirements as a form of protection, thus, there is a need to categorize security requirements according to the related assets.

"***Perform Risk Assessment***", step in the SQUARE approach is needed to find the risk for the whole software, using misuse cases, scenarios and security goals, which are performed earlier; this process will be done in conjunction with the risk assessment method. This method has been discussed in details in chapter five, but in the normal method (SQUARE method), the external method is used to make risk assessment.

"***Prioritize Security Requirements***", step in SQUARE depends on the results of the 'perform risk assessment' step, because the security requirements, cannot be prioritized unless there is a value for each single security requirement alone; this step will also be discussed in chapter five.

b) Discussion and justifications of CLASP

The second step of this design phase, focuses on what should be in future system, where, the users and the developers use the AI to elicit software requirements using the diagrams, which have been drawn in the same phase; on the other hand elicitation of the security requirements is necessary in the same step by using the diagrams, which are performed by the activity of CLASP best practices "***Identify User Roles and Resource Capabilities***", to help security experts elicit security requirements, in conjunction with the process of eliciting software requirements. An extra step of CLASP best practices is also carried out "***Apply Security Principles to Design***", where this activity is needed to illustrate the misuse cases diagrams, and to identify attack surface. This activity will help the security experts in accurately eliciting security requirements, thus, there is a need to conduct this activity this time.

"***Perform Security Analysis of System Requirements and Design (threat modeling)***", as a CLASP activity is needed, especially after completing the previous activity, to find the risk of the whole software with other steps in this phase, using different diagrams. In this phase, all software requirements and security requirements must appear, and this gives us another reason to integrate this activity.

The next step of the AI phase (Design Phase) focuses on checking and assessing of the existing software requirements, by using the external point of view, to verify that the process is proper. CLASP best practices focus on this assessment, but from standpoint of security, by using "***Verify Security Attributes of Resources***" activity. From here, there is a strong need to integrate this activity with this step.

"***Perform Source-Level Security Review***", and "***Identify, Implement, and Perform Security Tests***", these activities are part of the main CLASP best practice "***Perform Application Assessments***" in the original CLASP. The step focuses on reviewing security for the last time, before proceeding to the next phase; this review is very important because, security issue is a critical issue in any software industry, as it may lead the proposed software to fail, if it is not checked and tested many times. The above reasons lead us to integrate the security review step and the security test step

with this phase of proposed technique, which is equivalent to this SQUARE step named "***Categorize Security Requirements and Perform Risk Assessment***".

Even after completing the 4th and 5th steps "***Security Review and Security Test***", in case if there is a security problem (some source-level not covered by security), there is a plan to address it by integrating "***Manage Security Issue Disclosure Process***" activity, to manage and recover the security issues which have not been taken into account, and "***Address Reported Security Issues***" to address what has been found, in terms of security issues.

In the next step of SAIT, the "***Document Security-Relevant Requirements***" activity of CLASP best practices is integrated, because for sure, after gaining, reviewing and testing the security requirements, they have to be documented. This documented security requirements, as to quantify security requirements will be revisited in chapter Five.

"***Integrate Security Analysis Into Source Management Process***" activity of CLASP best practices should be integrated with SAIT because, the software requirements have already been gathered, and now the security requirements is available to cover the software requirements. Thus, it is necessary to integrate security requirements with related functional requirements or business requirements, to know the extent to which this software is secure; this integration is one of the inputs to other processes explained in chapter five, for the purpose of quantifying security requirements in the software industry.

In the last step of CLASP the "***Monitor Security Metrics***" activity is integrated as well, because the metrics are important factors for a general software security attempt, particularly in the requirement phase. They are essential in determining the present security stance of software, and focusing on the most crucial weaknesses through the use of some tools. This step has a strong and direct relationship with the quantification of the security requirements carried out in chapter five.

4.3.4. Destiny Phase of proposed SAIT

This section presents the integrating of steps in CLASP, SQUARE with AI in the destiny phase of the proposed Secure Appreciative Inquiry Technique as shown in table 13.

TABLE 13: Proposed destiny phase for SAIT

Existing Techniques			New Technique	
AI (Destiny Phase)	**SQUARE**	**CLASP**	**SAIT (Destiny Phase)**	**Who Involved**
'what can be': • Focus on the expected software, which will be developed and employed • Defining the activities that have to be considered	• Requirements inspection (the requirements are analyzed for vagueness, this is the ultimate security requirements	Nil	1. Software Requirements and Security Requirements (AI) & (SQUARE) 2. Expected Software (AI) & (SQUARE)	Stakeholders, Developers and Security Experts

a) Discussion and justifications of SQUARE

Destiny phase: focuses on what features will be in the future's system; AI will define all software requirements and activities that have to be considered in future system, to be more vital and tangible.

The final step of the SQUARE method is known as "***Requirements Inspection***", where the requirements are analyzed for vagueness, variances, and inappropriate sense. The outcome of this phase is the ultimate security requirements information for the stakeholders. So, integrating this step of the SQUARE method with the Destiny phase of SAIT is sensible; because all the steps of the SQUARE and AI approaches will show the result (Software Requirements and security requirements) in this final phase.

b) Discussion and justifications of CLASP

Any activity of CLASP best practices in "Destiny phase" has not been integrated, due to the following reasons:

i. This method covers the whole SDLC, and it does not focus on the requirement phase.
ii. CLASP best practices encompass the destiny phase, but this destiny is for the whole system (testing phase), which means that, the testing phase for the whole system is the one, which selects the destiny of the whole system.

All the steps and processes discussed in this section are shown in table 14.

TABLE 14: Complete phases for proposed secure appreciative inquiry technique (SAIT)

Existing Techniques			New Technique	
AI Phases	**SQUARE**	**CLASP**	**SAIT**	**Who Involved**
Discovery Phase (Table 10)	• Agree on definitions (**Security Requirements**) • Identify security goals, which are achieved by the current system	• Identify global security policy • Identify resources and trust boundaries • Specify operational environment	1. identifies the existing situation of the old system (**Software Requirements**); system description **(AI)** 2. agree on definitions (**Security Requirements**) **(SQUARE)** 3. current system feedback **(AI)** 4. identify security goals, which are achieved by the current system **(SQUARE)** 5. identify global security policy **(CLASP)** 6. identify resources and trust boundaries **(CLASP)** 7. explain experience and skills **(AI)** 8. specify operational environment **(CLASP)**	Stakeholders, Users and Developers
Dream Phase (Table 11)	• Identify security needs; what are the dreams to protect the system assets (Future System)	Nil	1. What are the new business requirements they dream to achieve **(AI)** 2. Identify security needs which are related to the future system **(SQUARE)**	Stakeholders, Developers and Security Experts
Design Phase (Table 12)	• Develop artifacts to support security requirements' definition • Draw Use Case, Misuse Case	• Identify user roles and resource capabilities • Apply security principles to design	1. Examine the feasible size of the system. Based on the recognized strengths highlighted by Discovery and Dream Phases **(AI)** 2. Develop artifacts to support security requirements'	Users, Developers and Security Experts

	Diagrams • **Elicit security requirements** • Categorize security requirements • **Perform risk assessment to the elicited security requirements by using external tools (Table 9, p.55) to quantify, categorize and prioritize security requirements to the whole system**	• Detail misuse cases • Identify attack surface and annotate class designs with security properties • Perform security analysis of system requirements and design (threat modeling) • Verify security attributes of resources, Perform source-level security review • Identify, implement, and perform security tests • Manage the security issue disclosure process • Address reported security issues • Document security-relevant requirements • Integrate security analysis into source management process • **Monitor security metrics**	definition **(SQUARE)** 3. Identify user roles and resource capabilities **(CLASP)** 4. Apply security principles to design **(CLASP)** 5. Perform the security analysis of system requirements and design (threat modeling) **(CLASP)** 6. Draw Use Case, Misuse Case Diagrams (Identify attack surface and annotate class designs with security properties) **(SQUARE) & (CLASP)** 7. External point of view **(AI)** 8. Verify security attributes of resources **(CLASP)** 9. **Elicit security requirement (SQUARE)** 10. Categorize security requirements (Perform source-level security review) **(SQUARE) & (CLASP)** 11. **Perform risk assessment to the elicited security requirements by using external tools (Table 9, p.55) to quantify, categorize and prioritize security requirements to the whole system**: (Identify, implement, and perform security tests) **(SQUARE) & (CLASP)** 12. Manage security issue disclosure process **(CLASP)** 13. Address reported security issues **(CLASP)** 14. Document security-relevant requirements **(CLASP)** 15. Integrate the security analysis into the source management process **(CLASP)**	

			16. **Monitor security metrics (CLASP)**	
Destiny Phase (Table 13)	• Requirements inspection (the requirements are analyzed for vagueness, this is the ultimate security requirements	Nil	1. Software Requirements and Security Requirements **(AI) & (SQUARE)** 2. Expected Software **(AI) & (SQUARE)**	Stakeholders, Developers and Security Experts

The steps in the proposed SAIT in the table 14 where there are four phases to complete the eliciting user and security requirements process through SAIT; the steps are explained in the following subsections:

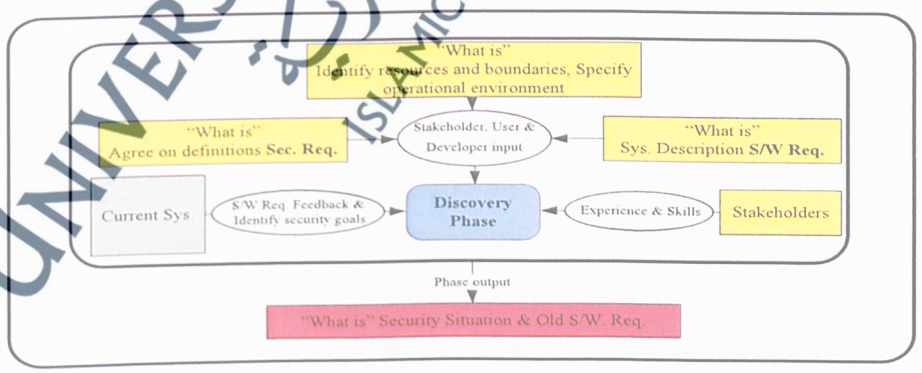
4.3.5. Phase 1: Discovery Phase of SAIT

In this phase, the developer finds the current situation of the organization. The stakeholders will provide their experience, skills and how they have been running the organization up to the present. They will disclose their success stories in the organization and how they have accomplished them as shown in figure 13. Three basic questions devised by Cooperrider (2008) will be used to for these assessments:

a) Describe the greatest experience of your lifetime.
b) What are the things that you value most in you, your work and organization?
c) What are the core factors that give life to your organization?

The above questions motivate the developers and security experts and give a big picture to the whole system; in addition, they also give them the magnitude of the suffering, by the tasks accomplished by users every single day.

FIGURE 13: Discovery Phase of SAIT.



- Identify the existing situation of the old system (Software Requirements); System description (Stakeholders, Users and Developers). What are the weaknesses of the

current system? What are the things that are not achieved by the current system and specific operational environment?

- Agree on definitions step. Enables apparent the connections between security requirements experts and stakeholders by structured interviews or focus group, to agree on definitions using a standards like IEEE. This facilitates identifying security goals, and the assets that must be protected. Thus, it contributes to facilitate elicit security requirements in future.

- Provide the current system feedback and identifying security goals, which are achieved by the current system. In the beginning, the stakeholders will declare several security objectives, and those objectives are structured in this step. Any issues can also be fixed in this phase.

- Guarantee that, security requirements has the identical degree of ownership, as all other prerequisites. Nevertheless, application architects and project managers can effortlessly focus on functionality, when determining requirements, as they help materialize the more significant objective of the software, to provide more worth to the organization. Security concerns can easily be tracked, so it is vital that, security requirements be a specific part of any application development effort.

- Identify the global security policy. The level of protection is suitable to address the risks of any organization, restrictions to which it is subjected, and the prospective effect on its reputation should be exploited.

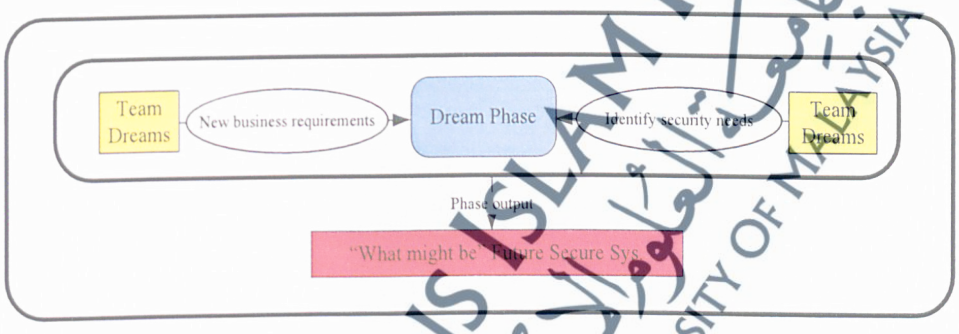
Output of Phase 1:

The outputs of this phase are: what are the security situations of the old system?; what are the old software/business requirements that are needed in future system, what are the definitions agreed, what are resources and trust boundaries, what are the global security policy and what are the existing situation of the old system (Software Requirements and security requirements).

Phase 2: Dream Phase of SAIT

This phase looks at, what the user sees as the future of the system, specifically and of the organization as a whole as shown in figure 14. Here, the imagination and creativity of the stakeholder will be encouraged. The stakeholder will be able to contribute more effectively, since they have already completed the discovery phase. There should be no constraints in terms of the dreams of the stakeholder.

FIGURE 14: Dream Phase of SAIT.



- Stakeholders, developers and security experts' (team) dream (future system, i.e. what do they dream about the software requirements to be in the new system (what are the new business requirements they dream to achieve)).

- Identify security needs/goals; what are the dreams to protect the system assets (future system). Definitions, 'candidate' goals, business drivers, policies and procedures, all these should be considered to make a concrete draft about 'what might be', to finish the daily tasks, using the proposed system, in secured means by protecting all the organization's assets and resources.

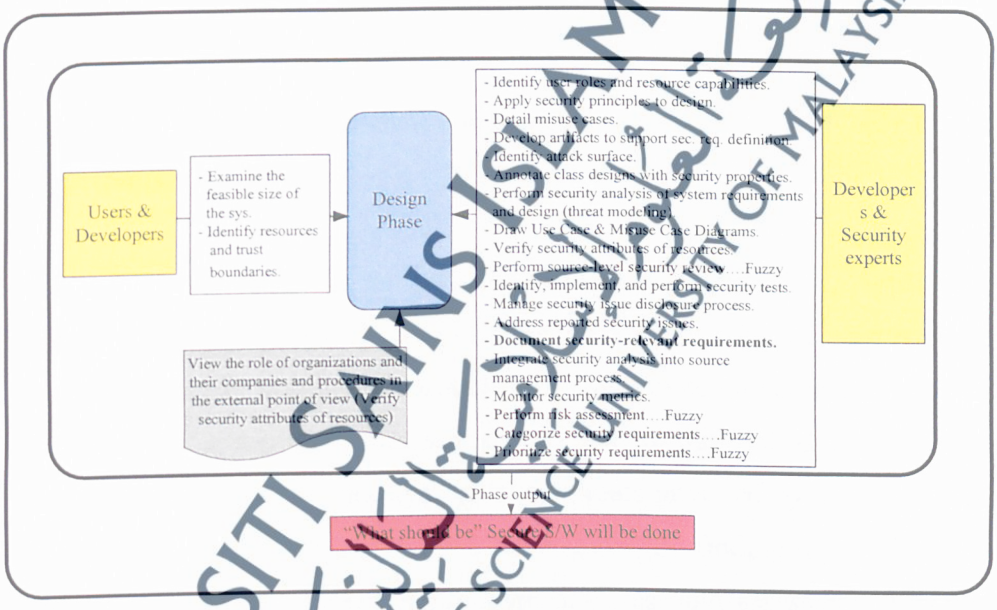
Output of Phase 2:

The outputs of this phase are: 'what might be' the system boundaries, assets, resources besides the agreed security definitions, which are related to business/software requirements.

4.3.6. Phase 3: Design Phase of SAIT

This phase looks at how the system could be, where the user and the developer look at the possible area for the size at the system. Using the identified strength that the stakeholder has highlighted, the size and the strength of the system can be formulated now. The role of the organization, relationship, policies and processes within the organization is looked at from an outsider's viewpoint.

FIGURE 15: Design Phase of SAIT.



- Users and the developers examine the feasible size of the system. It is based on the recognized strengths highlighted by the stakeholder.

- Develop artifacts to support security requirements' definition; the following relics are to be gathered: system architecture diagram, use case scenarios/diagrams, misuse case scenarios/diagrams, attack trees, and standardized templates and forms. Consequently, the dimension and the strength of the system can now be derived.

- Consequently the dimension and the strength of the system can be derived now (Draw Use Case Diagrams (Identify user roles and resource capabilities) and Misuse Case Diagrams; apply security principles to design (detail misuse cases, identify attack surface, and annotate class designs with security properties)); (Perform security analysis of system requirements and design (threat modeling)).

- Elicit software and security requirements. An essential factor in this phase is to make sure that the software/business requirements are proven, and that, they do not have any implementation or design restrictions, rather than requirements; thus, all security requirements that are related to software requirements should be elicited.

  a) The role of organizations, their companies and procedures are viewed in the external point of view.

  b) As a matter of fact, it is not possible to analyze security in an application, therefore application testing and evaluations should still be a core element of the entire security technique. Particularly so, automated assessments tests can discover the security problems that are not recognized during code or implementation reviews, discover security threats unveiled by the operational environment, and act as a protection mechanism, by finding downfalls in the design, requirements, or execution. Typically, test and assessment functions are held by a test analyst, or by the quality assurance organization, but it can be extended to the complete life cycle. The factors and elements that should be considered are:

     i. Verify security attributes of resources.
     ii. Perform source-level security review.
     iii. Identify, implement, and perform security tests.
     iv. Document security-relevant requirements.
     v. Integrate security analysis into source management process: A vital objective of software security is to generate and sustain multiple-use source code, which fortifies the fundamental security services in

software and over an organization's applications. This objective is most effectively accomplished, by applying secure development practices into an organization's general development process as soon as possible in the SDLC.

vi. Manage the security issue disclosure process, and address reported security issues: It is specifically significant in the perspective of program up-dates and improvements, to determine which actions will be employed to recognize, evaluate, focus on, and resolve weaknesses. Establishing resolving techniques will accelerate response, and reduce risks, by interpreting tasks, obligations, and procedures to adhere, after recognizing the weaknesses. Removal techniques are often fed by application tests, both in in-house or third party, and help to manage information when disclosure happens.

vii. Define and monitor security metrics: A project development team will not be able to deal with what it cannot determine. However, applying efficient analytics monitoring attempt can be a challenging aspect. In spite of this, metrics are important factor to a general software security attempt. They are essential in determining the present security stance to the organization, and concentrating on the most crucial weaknesses, and exposing how effectively or improperly the investments in improved security of an organization are performing.

viii. Realizing how applications will be employed, and how they could possibly be misused or infected.

ix. Potential preventive controls and their value and efficiency.

Output of Phase 3:

The outputs of this phase are: 'what should be' needed artifacts: scenarios, misuse cases diagrams, models, templates, threat modeling diagrams. Outputs also include the security metrics, and reported security issues, forms, initial cut at security and software requirements.

4.3.7. Phase 4: Destiny Phase of SAIT

This phase focuses on what will be the future system; this is more concrete than the dream phase, where the developer and user look at what is the foreseeable system that they will develop and use, see figure 16. The aim of this phase is to define the actions that need to be taken. Then the support from the organization can be gathered, as now the ideas of the system that need to be designed will become much clearer. Now the participation of stakeholders will be visible, and a general consensus should be available as how the system will be designed.

FIGURE 16: Destiny Phase of SAIT.

- The developers and users focus on the expected software, which will be developed and employed.
- They will define the activities that have to be considered.
- Support the organization to gather information related to the ideas about the system, which becomes apparent for design.
- Requirement inspection (the requirements are analyzed for the vagueness of the ultimate security requirements).

Output of Phase 4:

The outputs of this phase are: 'what will be' the initial selected requirements, the documentation of the decision-making process and the rationale.

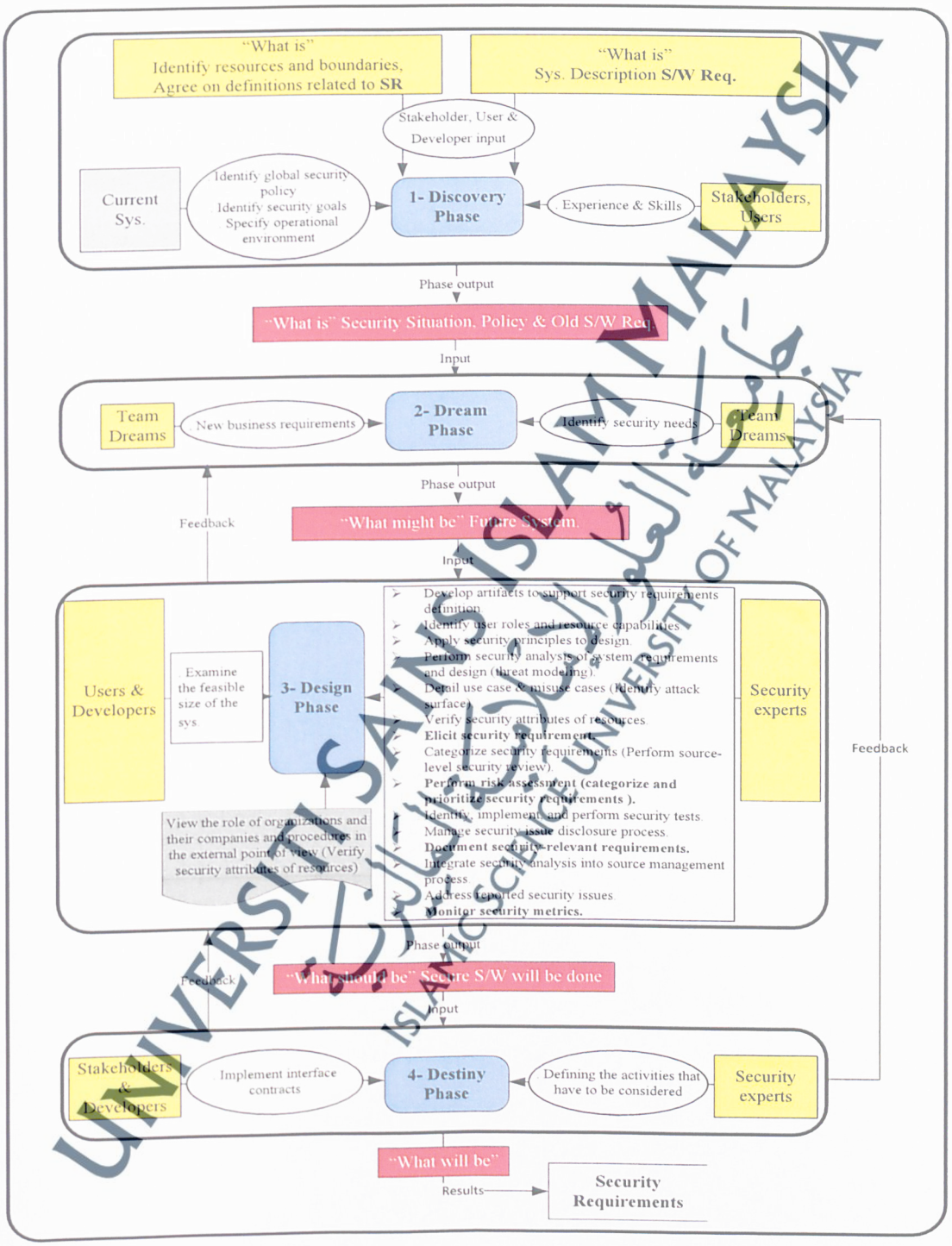
Table 15 summarizes the proposed integrated technique in SAIT phases and the main steps inside each phase, viewing the security issues concerned.

TABLE 15: Summarizing phases of SAIT based on concern of the security issue

Phase Name	Executers	Security issue	Expected results
Discovery	• Stakeholder • User • Developers	• Discover security situation • Security definitions agree • Resources and trust boundaries • Identify global security policy	• 'What is the' Security situation in the old system. • 'What is' the old software/business requirements needed in future system • 'What is' the security definitions agreed • 'What is' Resources and trust boundaries • 'What is' The global security policy • 'What is' The existing situation of the old system (Software Requirements)
Dream	• Stakeholder • User • Developers • Security experts	• Security definitions agreed, which are related to business/software requirements	• 'What might be' System boundaries, assets, resources besides the agreed security definitions, related to business/software requirements
Design	• Stakeholder • Developers • Security experts	• Misuse cases • Risk assessment • Threat modeling • Reported security issues	• 'What should be' Needed artifacts: scenarios, misuse cases, models, templates, forms, and threat modeling diagrams. • Risk assessment results. Initial cut at security and software requirements • Reported security issues
Destiny	• Security experts • Developers	• Initial selected security requirement	• 'What will be' Initial selected requirements, the documentation of the decision-making process and rationale

The steps in the proposed SAIT in the table 14 and 15 are illustrated in figure 17, where there are four phases to complete the eliciting user and security requirements process through SAIT.

FIGURE 17: Proposed Secure Appreciative Inquiry Technique (SAIT) Process.



4.4 Summary

In this chapter, the integration of CLASP and SQUARE with AI was done under AI phases (Discovery, Dream, Design and Destiny) to gain Secure Appreciative Inquiry Technique (SAIT). SAIT is a technique adopted to elicit both software and security requirement. Next, the evaluation of the risk assessment to the software in the design phase using some tools or methods like the security metrics which will be discussed in the following chapter.