Which Test Artifacts Testing Service Providers should Reuse and How? Experiences from a Case Study in the Chinese ICT Sourcing Market

Yikun Lu & Timo Käkölä
University of Jyväskylä, Finland
{yilu, timokk}@jyu.fi

Abstract
Software testing service providers are facing new requirements to shorten service times, lower costs, and increase service customization and quality. Reuse of test artifacts is a possible solution that can help providers to meet the requirements because reuse can improve software quality and productivity. However, the extant literature does not explain in depth which test artifacts should be reused and how. This paper focuses on ICT-enabled sourcing of software testing services in the Chinese market to identify the most important reusable test artifacts. There are two reasons for this research. First, most Chinese service providers are small or medium-sized and have to overcome obstacles such as the lack of advanced core technologies before they can play important roles in the global sourcing market. Second, testing is one of the best ICT services small- and medium-sized providers can provide to develop domain and technological knowledge required to overcome most obstacles.

1. Introduction
Software has become a key asset for competitive high technology products. It has also become ubiquitous in society. High software quality and competitiveness in software development have thus become critical concerns for software businesses. Increasingly, complex high quality systems are produced with constant, or even diminishing, human resources [44].

Software testing is an empirical investigation conducted to provide stakeholders with information about the quality of the products and/or services under test with respect to the contexts in which the products and services are intended to operate [25; 5; 28]. It is divided in static and dynamic testing. Static testing is the process of reviewing documents and detecting errors without executing the software. Dynamic testing is the process of detecting faults in the software by executing the software with appropriate test materials such as automated scripts or testspecific software components [7]. Software testing is crucial for software quality assurance [17]. It is also one of the most time- and labor-intensive activities, consuming between 30% and 50% of the total development cost [4; 38; 27]. Companies are often faced with time and resource constraints, limiting their abilities to complete testing efforts effectively [28].

Software testing has proven challenging enough to warrant the establishment of an industry of specialized testing service providers. With more and more incumbents entering the markets, competition within this industry has become increasingly fierce and time-based. To meet the competitive pressures, the providers need to be capable of developing mass-customizable and cost effective services that address the needs of their clients rapidly and profitably. Therefore, testing service providers need effective ways for executing software testing.

Testing effectiveness can be radically increased by creating reusable test artifacts (hereafter, domain test artifacts) for particular software application domains. Yet, conventional software testing is application specific, providing few test artifacts that could be reused across applications. Software artifacts and test artifacts are typically developed by different teams and described in separate documents, making test artifact reuse difficult [6; 16; 45].

Software product line engineering is a new paradigm through which software-intensive products, sharing common and variable features, can be derived quickly by developing and reusing common and variable domain artifacts for the product line [35]. Common domain artifacts deal with the features that are always included in the applications, while the variable domain artifacts provide the functional and quality elements that may vary or be totally excluded from the applications. Software product line engineering has received a great deal of attention for its potential to foster reuse of software artifacts across development phases [11], improving productivity [29] and software quality, reducing development costs [37], and shortening the time to market [13]. To reap
maximum benefits from software product line engineering, software providers should focus on specific application domains; create software product lines with a number of inter-related software products for the domains; establish requirements, reference architectures, and detailed designs for the domains; and radically improve testing effectiveness by creating domain test artifacts to test the requirements, reference architectures, detailed designs, and implementations of all applications [35].

Many methods and technologies have been developed to facilitate product line engineering, including design patterns, frameworks, and component implementation technologies (e.g., .NET and J2EE) [17]. However, reuse technologies have not been applied in testing service provisioning to a maximum possible extent. This is largely due to the fact that testing service providers serve multiple application domains lacking adequate commonality to warrant reuse investments.

Software product line engineering is knowledge intensive, requiring the learning, sharing, and accumulation of knowledge through knowledge repositories and social networks [15]. The creation and use of domain test artifacts benefit both testing service providers and clients. Effective use of knowledge gained from prior and current engagements enables the providers to reuse their best practices, to address problems that have occurred, and to improve overall performance and test service quality in current and future engagements [42]. Development budgets shrink because large numbers of high quality test artifacts can be built, used, and improved based on the experiences obtained when it is known that the artifacts will be used many times over long periods. These advantages, combined with the increased availability of domain artifacts, are the main reasons why reuse is becoming increasingly prominent [33].

The domain test artifacts must be stored in test artifact repositories accessible to all relevant stakeholders to enable artifact evolution and reuse. Application testers can then select, modify, and configure appropriate subsets of the artifacts to cover the features and non-functional characteristics of the applications being tested.

However, the extant research does not provide comprehensive guidelines for creating knowledge repositories for software artifacts [3; 8; 10]. Practical implementations of repositories and systematic research about the use of the repositories in real settings are also lacking. For example, Käkölä et al. [24] presents (1) detailed guidelines for designing repositories consisting of requirement, architecture, and detailed design artifacts for both domain and application levels, and their packaging into releasable entities, and (2) real life experiences from using such a repository. However, testing is out of the scope of Käkölä et al. [24]. In addition, much of the research is focused on large organizations in developed economies [12].

To facilitate the creation and use of domain test artifacts and associated repositories, this paper draws upon a literature review and a case study in the context of the Chinese ICT sourcing market. It probes the following research question:

- Which test artifacts the testing service providers should reuse during the testing life-cycle to shorten service times, reduce costs, and enhance service quality?

The chosen case organization, a medium-sized Chinese professional testing firm, is appropriate for this research for two reasons. First, the firm has developed its own test artifact repository, where all domain test artifacts are stored. Second, it has all the major characteristics of the Chinese ICT sourcing service providers. For example, it is growing fast and desiring entry into the international testing market. It can also offer various testing services. Research findings can thus be generalizable at least in the Chinese context.

The Chinese ICT sourcing market has been chosen as the context for this research because China has grown into one of the major sourcing service bases in the global ICT sourcing context [18] but the Chinese software industry is facing a great deal of challenges. Chinese providers need to overcome many obstacles before they can play important roles in the global ICT sourcing market. For example, they need to learn to scale up their production, develop competitive core technologies that can serve as domain artifacts, and develop and hire top-level executives and experts competent in international business [34]. Most Chinese providers are small or medium-sized. They typically leverage the mediated offshore sourcing model, delivering software services to larger foreign ICT clients that contract and interface with the actual end-clients onshore [23]. This business model usually restricts the providers to small, low-value projects and hampers the sharing of knowledge with end-clients, severely impeding the capability and business development of Chinese providers.

Software testing is one of the best ICT services small- and medium-sized providers can provide to develop domain and technological knowledge required to overcome most obstacles discussed above. For example, comprehensive testing services are practically impossible to develop without understanding the business domains of end-clients. Once such domain knowledge has been created, it can be deployed,
for example, to broaden the scope of services from testing to software development or to create new products, thus transcending the restrictions of the mediated sourcing model.

This paper contributes to software product line research and software testing research by delineating and analyzing the most important reusable test artifacts based on a case study and by illustrating how testing service providers should manage testing services to improve service effectiveness. This research shows that test plans, test cases, test reports, and the lessons-learned documents are the main domain test artifacts. Test artifact repositories are also crucial artifacts for testing service providers.

The paper is organized as follows. Section 2 discusses the case organization and the research methodology. Section 3 presents the findings from the case study, including what to reuse, how to reuse, and who should be responsible for reuse. Section 4 discusses the characteristics of knowledge that can be reused during the testing life-cycle. Conclusions and suggestions for future research conclude the paper.

2. Description of the research methodology and the case organization

This research uses a single qualitative case study to collect data covering the complete sourcing life-cycle for testing service providers, including the most important testing practices, the artifacts reused in these practices during the various phases of the life-cycle, and the people responsible for reusing the artifacts. As a result, this research provides a holistic, systemic understanding of the phenomenon of test artifact reuse in the context of ICT-enabled sourcing of software testing services [14].

Ltesting is a medium-sized (less than 50 employees) professional software testing services provider founded in 2002 [30]. It has rapidly established a leading position in the Chinese testing service market based on its rich testing experiences and professional services. It has established strategic partnerships with HP, IBM, and some other multinational companies. It offers various types of testing-related services, such as software testing services, test training services for individuals and companies interested in offering testing services, test management services, and consulting services for constructing software quality systems. When clients source testing work to Ltesting, it expects the clients to be closely involved in the sourcing engagements in order to ensure the sourced projects meet clients’ test requirements and help clients to obtain expected results. Ltesting implements most testing services offsite, being responsible solely for the test project. It also offers onsite testing services: its testers join clients’ test teams and are managed by the clients. To identify the most important domain test artifacts for the service provider, this research focuses on the projects following the offsite model.

Test teams are responsible for testing. Usually the teams have four roles: test manager, test analyst, tester, and seller (Table 1). Sellers serve as boundary spanners between clients and providers. Sellers are especially important for solving communication challenges in international sourcing when clients and vendors use different languages, have different cultures [31; 36]. Test teams can be organized flexibly based on the project characteristics, personnel workloads, and client requirements. For example, when the projects are small, testers need not be involved in test teams because test managers and test analysts can do their work.

The investigation has proceeded in the following stages. First, a reference model was selected from literature to understand the international sourcing life-cycle holistically from both clients’ and providers’ viewpoints. The eSourcing Capability Model for Service Providers (eSCM-SP) was selected as the reference model because it has been demonstrated to help various types of sourcing service providers to improve their capabilities related to both ongoing, phase-specific, and engagement-specific sourcing practices throughout the sourcing life-cycle [43]. According to the eSCM-SP, the life-cycle involves three phases from the provider’s viewpoint: initiation, delivery, and completion. This paper organizes findings related to the domain test artifacts based on the three phases to help providers reuse the artifacts during the life-cycle. Second, Chinese software industry and software testing services industry were studied [18; 31; 46;] and scientific literature was reviewed in an iterative fashion to identify the key characteristics of successful testing providers (e.g., international growth orientation, sophisticated web-based integrated information systems) and the most suitable candidates for an in-depth case study. Third, Ltesting was selected because it was successful and possessed the required characteristics. Fourth, the first author spent over three weeks observing life in the case organization, analyzing documents and memoranda, and interviewing key personnel. The in-depth interviews involved the CEO, all testing managers, and a number of test analysts to uncover the routine practices and information systems associated with testing work and major breakdowns disrupting work. Each interview was started by following a questionnaire and was concluded with an open discussion to address emerging issues. Interviews
were summarized and sent to the interviewees, who verified them and provided feedback as necessary.

<table>
<thead>
<tr>
<th>Title</th>
<th>Responsibility</th>
<th>Contribution to reuse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Manager</td>
<td>Test managers are responsible for test project planning, management, risk evaluation, and report review. During project completion, test managers summarize the projects and the lessons learnt. They have at least five years of relevant work experience.</td>
<td>Overall responsibility for management and maintenance of the test artifact repository. Identifying and selecting the test artifacts that will become domain test artifacts in the repository.</td>
</tr>
<tr>
<td>Test Analyst</td>
<td>Test analysts analyze test requirements; design test plans (together with the test manager); and design test cases. They have three to four years of relevant work experience.</td>
<td>Identify domain test artifacts that can be reused in the project. Support the conceptual integration of domain test artifacts in a specific project.</td>
</tr>
<tr>
<td>Tester</td>
<td>Testers conduct the specific test assignments. They have more than one year of work experience.</td>
<td>Submit change requests and defect reports to developers and maintainers of domain test artifacts.</td>
</tr>
<tr>
<td>Seller</td>
<td>Sellers communicate with clients, acting as bridges between clients and service providers. They need to have comprehensive testing knowledge, because they attend the testing service life-cycle from early bidding and negotiation through to service completion.</td>
<td>Receive feedback from clients concerning the test results achieved through reused test artifacts.</td>
</tr>
</tbody>
</table>

Table 1. Job Descriptions of Key Roles in Test Teams

Fifth, after the three-week visit, data about testing strategies, routine practices, organizational structure, and enabling classes of information systems, uncovered through observations, document analysis, and/or the interviews, were analyzed to create a preliminary software testing life-cycle model. Most significant breakdowns in routines were also analyzed because the model should help providers to improve their processes and competencies in order to eliminate most breakdowns in advance. The phases of the preliminary model were compared to the respective sourcing phases prescribed by the eSCM-SP. Testing related literature was also used. If the analysis indicated that major deviations existed or some information was missing, clarifications were requested from key informants through email. Data collection and analysis continued for several months using the internet to collaborate with the case organization.

3. Most important reusable test artifacts based on the case study

This section describes the most important domain test artifacts according to the software testing life-cycle model. Table 2 summarizes the test artifacts and life-cycle phases in which they are reused.

<table>
<thead>
<tr>
<th>Reusable test artifacts</th>
<th>Responsibility</th>
<th>Life-cycle phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test plan</td>
<td>Test manager and test analyst</td>
<td>Initiation</td>
</tr>
<tr>
<td>Test case</td>
<td>Test analyst</td>
<td>Delivery</td>
</tr>
<tr>
<td>Test procedure specifications and defect reports</td>
<td>Test analyst and Tester</td>
<td>Delivery</td>
</tr>
<tr>
<td>Summarized service reports</td>
<td>Test manager</td>
<td>Completion</td>
</tr>
</tbody>
</table>

Table 2. Reusable Test Artifacts

To facilitate reuse of test artifacts in later projects, Ltesting’s test manager will document in the end of each project the experiences that the test teams have gained and the artifacts they have created. Each artifact in the test artifact repository is the result of a packaging document that prepares an existing test artifact for reuse [40]. Each artifact package includes the documentation of the requirements and design specifications, the implementation environment, usage examples, and change requests for the artifact. To guarantee the reliability and quality of the artifact, each artifact package is validated before it is admitted into the repository. A sample application demonstrates how service projects can use the domain test artifacts. Even novices can then benefit from the organizational knowledge [32].

Test managers and test analysts work together to draft test plans. Test analysts design and review test cases. Test managers and test analysts must create a good conceptual understanding of the relevant domain test artifacts and artifact interfaces during service initiation to appropriately integrate the artifacts for effective service delivery. Test analysts then identify functional components of the application to be tested that are not unique to the application. It is probable that domain test artifacts for such components already exist [40]. Test analysts search for appropriate domain test artifacts from the repository. If the repository contains an artifact implementing (parts of) the required functionality, they validate that the arti-
fact can meet the requirements and/or modify it to meet the requirements. If no domain test artifacts can be found for common application-independent components, it is probably a good idea to start creating them as soon as possible, as they will save time and money and improve service quality later.

Generic test templates for multiple domains are commonly accessible from Internet and other channels. For example, IBM Rational Unified Process [39] offers comprehensive test templates for testing services such as functional tests, performance tests, and user interface tests. However, software businesses vary in terms of factors such as technologies, markets, competitive environments, software processes, the numbers and capabilities of personnel, and corporate cultures [18]. That is why most software process improvement initiatives and quality management toolkits recommend the customization and calibration of tools and templates to meet organizational requirements. Ltesting has decided not to use generic templates. It has developed own templates because they are more detailed and specific to local needs than generic templates.

“We will not use any template directly, but select the helpful parts. The most important [selection] criterion is that the selected parts meet the test requirements.” (CEO)

3.1. Reuse in test plan generation

In the initiation phase, the test manager and test analysts need to draft a preliminary test plan to decide whether it is feasible to bid and, if it is, to create a tender. The plan includes the estimated work effort (in person hours), the required time, and the price. If they win the contract, they need to draft a specific test plan to describe how they will meet client’s test requirements. The specific test plan defines the scope (i.e., the features to be tested and the testing activities to be performed); the methodologies and technologies to be used; the test artifacts to be reused; the people responsible for the tasks, and other resources; and the schedule of the testing activities [20]. Using historical information about previously tested similar applications can reduce the costs and improve the accuracy and relevance of test planning. For example, information about the planned and actual schedules and work hours spent on different testing activities help to estimate the needed time and resources (e.g., testers and equipment) of the projects being planned.

“Our previous experiences improve estimation accuracy.” (CEO)

When the historical data of the testing projects is stored in and accessible through appropriate repositories, it will be easier to decide which approaches should be used based on past experiences [22].

“We need to assess the test requirements and review relevant service experiences. Then we can identify the helpful artifacts that can be reused.” (CEO)

Ltesting records the relevant information about each project in a test artifact repository. During both preliminary and specific test planning, test managers and test analysts draw upon the repository and their personal service experiences to choose the best practices for meeting clients’ test requirements. During the research, all the interviewed managers emphasized the critical role of service experience during the service delivery.

“Rich service experience helps us understand quickly what the clients want and estimate the required work effort and time and appropriate human resources for the project, which is crucial to negotiate the right price and schedule for the project.” (Test manager)

The knowledge reused in test plan generation and test case generation has been created by test managers and test analysts during the completion phase of earlier projects. The people in the roles of test managers and test analysts change over time but the knowledge they create is so role and task specific that new people in the same roles usually do not have substantial challenges in deciding what information is useful and in putting the knowledge to effective use, because they generally understand their predecessors’ implicit knowledge and assumptions [1]. Therefore, they can more easily understand and deal with contextual information in the documentation that might be “incorrect, incomplete, or incoherent” and they can successfully reuse the raw, unprocessed records created as a by-product of knowledge work [32].

In the same way that design patterns are developed and stored for design reuse, test plan generation patterns can be created for test plan generation. Ltesting has developed test plan templates for different application domains, guiding test managers and test analysts to plan and do the necessary tasks without forgetting anything.

3.2. Reuse in test case generation

In the initiation phase, test analysts will draft and review test cases to meet all the test requirements. The generation cannot be fully automated but Ltesting uses different types of test case patterns to generate test cases for different types of functionality. For example, Ltesting uses a user interface test case pattern as a reference to draft test cases for user interface testing. As a result, test cases can be generated faster. Another benefit from establishing
test case patterns is that service providers can continue to provide testing services even when key staff members have left.

“We require our test managers and test analysts to work together in order to draft test plans and test case patterns for every project. If someone leaves, the remaining staff or a newly hired test analyst can continue the work based on the existing plan and patterns. However, none of the test artifacts are reused as they are. We will compare relevant projects and test case execution environments to the project at hand. If the situation is the same, we only need to revise test parameters [e.g., inputs and expected outcomes of test cases]. If the situation is similar, we have to identify the differences and select the useful parts to reuse. Generally, test case patterns are used as references for test case design.” (CEO)

Ltesting stores and structures test cases to several sub-repositories based on the types of test cases. For example, functional test cases, user interface test cases, and performance test cases are stored in respective repositories. Test cases are not structured into repositories according to business domains (e.g., banking, insurance) because there are functional and quality requirements such as user authentication, security, and access controls that require very similar test cases across business domains.

Test cases are produced after the test plan has been completed. The bodies of test cases should contain test sequences, which are valid for testing the target software. The behavior related to the test environment (e.g., the information about the software, hardware, and other factors enabling a stable state where the test cases for the tested piece of software can be run) should be recorded in preambles. The behavior of the test environment generally requires that the system has to be brought to a certain state before the actual testing can be initiated. Generally, testers select test cases in an iterative manner, starting with an initial test case set and selecting more test cases based on the experiences from executing the test cases [22]. This matured set of test cases should be stored in the test artifact repository for use by other stakeholders. For example, test managers and test analysts testing similar applications can get ideas from this set and design test cases productively.

To reuse and manage test artifacts and their interconnections to related artifacts, service providers need an integrated management system for requirement, feature, test, and release artifacts [24]. It should be used for all test projects and offer an easy and effortless connection to the test artifact repository, from which the test analysts are able to retrieve the necessary artifacts. Ltesting uses HP Quality Center Software (QC) [19] to manage the testing life-cycle in collaboration with its clients. QC offers a web-based globally accessible suite of applications, supporting all essential aspects of testing from requirements management through test execution to defect management. Ltesting can serve its end-clients directly based on QC. In addition, QC offers different roles for different stakeholders to ensure people can access only the information targeted to them. One of the most critical ongoing knowledge management activities on the organizational level is searching and identifying test cases that have been designed during specific engagements but are generalizable for reuse in other similar engagements. During the service lifecycle, test managers and test analysts can access the relevant test case patterns easily, helping them identify helpful artifacts, save time, and improve quality.

“During the service, we use QC to manage test cases. After the service delivery has been completed, test cases and other test artifacts are transferred to clients, but we will create test case patterns and store them in repositories for subsequent reuse.” (CEO)

Regression testing is an example of the reuse of test cases in the project level. It is the selective retesting of a modified software system to ensure that the bugs have been fixed and the newly added features have not created problems with the previously implemented functionality. To apply domain level reuse, test artifacts should be developed for a representative application in the domain and reused for testing other applications within the domain.

Test cases can be modeled through Unified Modeling Language [21] but Ltesting mainly uses natural language to describe test cases. The lack of standardized test case modeling hampers reuse. Ltesting would greatly benefit from using UML to model test cases. Test managers and test analysts should use UML not only to facilitate reuse but also to test features through UML designs without executing the software. In addition, UML facilitates test automation. Ltesting has recently started to use UML to describe new test cases and rewrite the old ones.

3.3. Reusing the test environment configurations and test procedure specifications during test execution

During the test execution phase, testers use test cases, implement them, conduct the tests after constructing a specific test environment, analyze the results, and generate test reports. A test environment is composed of parts such as hardware and software, connections, environment data, and maintenance tools and processes [2]. The parts are organized into
specific environment configurations enabling the tests. The configurations are deployed to simulate routines that the tested software would implement in practice. Ltesting divides test tasks into specific test assignments and presents the test requirements using a tree structure. The test execution sequence depends on the established tree structure. Newly added features are typically tested incrementally and finally the new combined system is tested. For example, testing can proceed through unit testing, integration testing, and system testing. Testers set up the test environment configurations and implement the process gradually by invoking test cases based on the hierarchy. The test environment configuration documents are developed by analyzing the domain. They can be reused.

Test results must be analyzed to facilitate reuse, because test plans may require modifications after the completion of each test cycle. The analysis provides the previous phases with feedback on test plan quality, test approach suitability, correctness of the test implementations, and the coverage of the test plan. Ltesting analyzes the results by collecting relevant data and documenting the results in test procedure specifications. These documents identify (1) special requirements for setting up the test environment and (2) the methods and formats for reporting the test results and measurements. These specifications are used to execute test cases and can be reused for testing applications belonging to the same domain.

Even if the artifacts were well organized and accessible through test artifact repositories, testers may have considerable difficulties finding, selecting, and applying the most appropriate artifacts, if they lack the domain knowledge to interpret the artifacts [32] or are newly hired and simply do not know enough about the repositories. In such cases, testers cannot execute test assignments as expected without help from test analysts. Testers of Ltesting sometimes had these problems, but Ltesting worked actively to reduce them by having regular interactions with the clients before and during the service life-cycles to improve the domain knowledge of test teams.

3.4. Reuse in test report generation

Ltesting uses two kinds of test reports. In the test execution phase, it uses defect reports to inform clients about the found defects. In the completion phase, it provides clients with summary reports about the services delivered.

During the test execution phase, service providers uncover software defects, locate the causes of defects, and correct them. It is possible to reuse the test defect reports generated for earlier application projects within a domain during the testing of new applications belonging to that domain. For example, user interface defect reports identify the causes of past user interface defects. The causes may be the same in new user interface development projects.

In the completion phase, the summary reports document the service process, found defects, defect analysis, the test execution environment, and used testing methodologies and technologies. However, service providers should go beyond simple documentation and create patterns and/or templates that they can actively reuse whenever they offer similar services for the same application domain in the future.

3.5. Reuse the summarized knowledge for improving future service deliveries

In the completion phase, Ltesting systematically reviews the service life-cycle, including the executed test cases, the test plan, and other work products produced during the previous phases, to determine the service quality and the lessons learned and to identify areas to perform better in the next projects. It conducts a formal review process, comparing the service deliverables to the test requirements specified in the test plan. If the review indicates that the conducted tests do not fully meet the test requirements, the test team will continue testing until all the requirements are fulfilled and performs a new review. When the review is positive, Ltesting will release the final defect report to the client. Without formal, repeatable evaluation processes, testing evaluations would be unnecessarily biased.

The review report resulting from the review process can be used to measure the quality of services and the performance of the provider and to improve the personnel skills. It can be reused with appropriate modifications in a particular domain to improve testing services for the same domain in the future. It can also be reused to better manage current projects. For example, when a new project is in the same domain as some previous projects, the test requirements and test sequences are similar across the projects. Project and employee performance can thus be reviewed phase by phase by comparing the project to the previous ones in order to identify problems in the project that have never occurred before and areas where the performance is below the expectations. Past experience from earlier projects may also show that some application areas in the project tend to have especially high defect densities and/or specific defects. Tests can then be targeted to those areas to uncover the defects.
Ltesting records the breakdowns in routines and their causes to the review report throughout the service life-cycle and takes corrective actions as necessary. For example, if a service delivery has been delayed by poor test artifacts, such as unreasonable test procedures or test plans, Ltesting refines or revises its service procedure and/or test artifacts. Additionally, the number and nature of errors found and a summary of actual metrics data (e.g., the realized effort) will be collected.

“We will summarize every project and compare it to previous similar projects that belong to the same domain, helping us to refine our test artifacts and improve our services.”(CEO)

4. Discussion

One of the main barriers to overcome when initiating knowledge management processes within a software business is the structuring of a knowledge repository to disseminate and reuse knowledge across the organization [24; 40]. A test artifact repository has been structured in Ltesting for storing and managing domain test artifacts representing abstract design solutions for families of testing problems. The repository has facilitated the identification, selection, and reuse of test artifacts, saved time in many cases, and reduced fluctuations in service quality.

“The repository has helped us reuse the artifacts during the services, saving some time and guaranteeing the service quality.” (Test manager)

Repositories are thus of critical importance to testing service providers.

Methods such as feature-oriented domain analysis [26] should be used to identify the components and configurations to be tested and then structure the tests in such a manner that the tests match the features of the product configurations. For example, Ltesting performs domain analysis for user interface testing. User interfaces are developed with different programming languages (e.g., C++, JAVA) based on different platforms (e.g., Windows, Linux) to support different screen resolutions. Domain analysis may reveal that the user interfaces must share a uniform style and a set of common features despite these differences. Common test principles can thus be used and domain test cases can be designed and reused to implement the tests for the features of these user interface variants.

Most mature reuse processes use the ideas of product-line architectures [9] and domain analysis [41] to discover families of products that share common features and qualities. In order to achieve the high levels of maturity in test artifact reuse, testing service providers need a strategic approach that focuses on a portfolio of related products in an application domain instead of unrelated projects. The products must possess enough common characteristics to make reuse investments viable. Service providers responsible for the testing phase cannot directly influence software product line engineering because their clients are responsible for developing the products. However, testing has a pervasive role in product line engineering because product line requirements, reference architectures, and reusable software components need to be tested in a coordinated, holistic fashion [35]. If service providers can develop competencies enabling them to deal with all these aspects of product line engineering, they can greatly help their clients in transitioning from project-based business models toward product-line oriented business models. In addition, if service providers can establish long-term partnerships with clients that are only interested in traditional software project business (i.e., the long-term development and maintenance of tailored client-specific systems), a large set of reusable test artifacts can be developed to test different releases of the tailored systems over time.

Based on the analysis above, many artifacts can be reused during the testing process. Two basic types of test knowledge are recognized: generic test knowledge, which applies to most projects (e.g. test plan templates), and project specific test knowledge that can be used in specific domain (e.g. test case patterns). Test artifact reuse differs from software reuse because the artifacts cannot be reused alone. They are always associated with the requirements, software components, interfaces, or features that are the subjects of testing. In order to produce reusable unit tests, software components have to be designed for reusability [27]. Analogously, component integration tests can be reused for subsystem testing if there is a reusable product line reference architecture shared across all applications to be tested [35].

5. Conclusions and further research

Software testing effectiveness should be increased by creating domain test artifacts. This research found based on a literature review and a case study that test plans, test cases, and test reports are the most important test artifacts to be reused during the testing life cycle. Lessons learned from the testing projects need to be accumulated in order to improve services in the future. Test environment and test procedure documents can be reused in specific domains. Test
case patterns and test plan templates are high-level artifacts, which can save time and help novices to perform test services effectively. Significant reuse work is implemented in the initiation phase when the test managers and test analysts choose suitable reusable test artifacts from the test repositories and define the best practices to meet client requirements. Testers provide valuable feedback concerning the reused artifacts to improve the quality of domain test artifacts and the usefulness of test artifact repositories. Test tools can be reused too. For example, HP Quality Center offers test management services which can be reused across various types of projects.

This research is limited to a single case study in a Chinese medium-sized testing service provider. Further research needs to examine in other organizations and countries, which domain test artifacts are the most important ones to reuse in order to help testing service providers to locate and reuse the artifacts more easily. Future research also needs to investigate how testing service providers could become central integrators of system, subsystem, and component-specific knowledge during the systems development phases that precede domain and application testing. If they could become such integrators, they could also catalyze the transitioning of their clients from project-based businesses toward software product line companies that strategically build and reuse knowledge assets to accelerate their product development and improve their product quality and end-user experience while lowering costs. We will investigate these areas and also participate in the international standardization work on software product line testing to help testing service providers improve their service qualities and shorten delivery times.

6. References

[24] Käkölä, T., M. Koivulahti-Ojala and J. Liimatainen,


