Recurring Structures of Subcontract Management in System Outsourcing

Haji Gul Wahaj and Valentino Vranić
haji.wahaj@stuba.sk, vranic@stuba.sk
Institute of Informatics, Information Systems and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava

Despite numerous efforts to address system outsourcing problems in the subcontract management system, the existing available solutions do not appear to be sufficient. They are fragmented into individual patterns, neither composing legible pattern languages to address the subcontract management systems nor being integrated into comprehensive pattern languages for managing system outsourcing in general. Communication among stakeholders is another issue that takes a long time because of their naturally hierarchical or multi-layered structure. And usually, false information reaches the sub-contractor at the bottom of the chain, which affects a project, and causes delays in deliverables. We attempt to overcome these problems by providing a set of four organizational patterns for the subcontract management system. Using them in combination with other available patterns and practices, we demonstrate a pattern language for better communication among different involved organizations. This paper focuses on better relationships between stakeholders. The relationships contain an inter-organizational communication and coordination among the stakeholders. A pattern story of a real case illustrates the overall use of this pattern language.

Categories and Subject Descriptors: [**Software and its engineering**] Patterns

General Terms: Patterns
Additional Key Words and Phrases: System outsourcing, multi-sourcing, software engineering, subcontract management, organizational patterns

ACM Reference Format:

Wahaj, H. G. and Vranić, V. 2022. Recurring Structures of Subcontract Management in System Outsourcing. HILLSIDE Proc. of Conf. on Pattern Lang. of Prog. 29 (October 2022), 15 pages.

1. INTRODUCTION

Outsourcing refers to a contract-based business in which a client forwards a module or system (such as the whole project) to a third party (an external service provider). It is intended to reduce cost by transferring part of work to a third party rather than completing it within the organization. In brief, outsourcing is the delegation of operations or tasks to a third party. The third party may perform tasks better, faster, and cheaper [Khan et al.(2009)]. Outsourcing has become an effective business strategy proliferating in the software development industry.

Besides the advantages, some common disadvantages might be a lack of communication, adequate skills and understanding gap between involved parties, poor design of the contract document, and legal and cultural challenges [Skjoett-Larsen(2000)]. There are always some gaps between the client, contractors, and subcontractors. Within a political context, there are many stakeholders with conflicting goals, and outsourcing is a severe challenge for public contract managers. The difficulties in software subcontracting often relate to contract terms and conditions, requirements engineering, project management, guaranteeing the overall quality of the subcontracted components, and the communication required in the different phases of the subcontracting period.

Some projects fail due to ineffective hierarchical or multilevel communication between involved parties. Hierarchical or multi-layered contract management systems often fail. For instance, a software project is outsourced to Company A with a total cost of $100K. Company A then establishes a contract to complete the project with Company B with a total cost of $80K. Finally, Company B establishes another contract with Company C with a total cost of $50K. A hierarchical contract brings communication and coordination challenges resulting in low quality or even project failure. Aside from the impact on quality, communication and coordination are affected in hierarchical contracts, and irrelevant information or directions reach the lowest level of sub-subcontractor. It is pertinent to define the authorities of the main contractor Company A in the contract document. Contract documents should specify which authority the main contractor contains. For example, the main contractor may or may not bind further subcontracts.

Numerous researchers have pointed out the significance of an accurate outsourcing contract. For instance, Gong et al. addressed that a contractor alone is not a guarantee of success when a client engages with a supplier in a system outsourcing arrangement [Gong et al.(2007)]. Some contracts worth billions of dollars have failed due to a lack of proper outsourcing contract management [Mclaughlin and Peppard(2006)]. Guers et al. also presented that the use of subcontractors leads to both restructuring and tensions within the workforce [Guers et al.(2014)].

The requirements for a service level agreement (SLA)[1] should be clearly spelled out in a contract that is comprehensible and comprehensive. An SLA usually defines the agreement level of performance, serviceability, and availability of operations in predefined terms. The vendor agrees to perform stipulated tasks within a specified time limit. Usually, penalties are imposed if the vendor fails in the deliverable or to achieve the goals on time [Marilly et al.(2002)].

The contract of outsourcing software might be considered either single-sourced or multi-sourced[2] Multi-sourcing is a recent and modern concept in which multiple vendors handle all outsourced activities. A vendor company simultaneously works with several vendors according to their domain of expertise in the software outsourcing business. The multi-sourcing solution is a better option to replace the multi-layered contract management system.

One of the challenging tasks for the software engineering team is to understand the fundamental requirements of a customer comprehensibly. This is to find out the exact limits a business can accept for a particular job under the contract. For people who work under a subcontract, the same understanding is required. Furthermore, communication and coordination are adversely impacted, and irrelevant information or directions reach the sub-subcontractors at the bottom of the chain.

One of the proposed efficient communication tool might be *inter-organizational technology-mediated communications*, which provides equal access to all involved organizations through organizational patterns. It contributes in the solution of the above-mentioned problems mentioned in this study. Organizational patterns arise from the core problems that an organization observes. Patterns are the recurring suitable solutions to a problem in a specific context [Sutherland and Coplien(2019)]. The patterns were observed and mined from our personal experience while working with a public organization.

The rest of the paper is structured as follows. Section 2 provides general information about the case we studied. Section 3 introduces the background information on the organizational patterns. Sections 4–7 presents the newly

[1] Service level agreements define what services a provider will provide and the service standards it must meet, and they are contracts between service providers and their customers.

[2] A single-sourced contract involves dealing with a single service provider, while multi-sourced contract involves several contractors who are experts in their respective fields.

discovered organizational patterns. Section 8 discusses the organizational patterns. Section 9 presents related work. Section 10 draws some conclusion and outlines future work.

2. THE DEVELOPMENT OF THE ELECTRONIC NATIONAL IDENTITY DOCUMENT SYSTEM

An outsourced project named the Electronic National Identity Document (eNID) system was considered as a study based on our personal experience and observations. Our eNID team worked on the project for almost a decade. The eNID team was located in the Ministry of Communications and IT (MCIT) of Afghanistan. The main contractor's headquarter was located in UAE with several subcontractors worldwide. MCIT had only direct communication with the main contractor remotely. During the process of implementing the project, several advantages and disadvantages were observed. It was a national project based on electronic government in Afghanistan. The ministry of communications and IT, the public sector was the project owner. The main contractor was from the private sector, Firm ABC. The firm had several joint ventures, subcontractors, and sub-subcontractors worldwide. The eNID project was not only software development, but it was a whole solution outsourcing. The first author was responsible for the whole project as the eNID project Manager.

The project had many modules such as Resident Management (RM) or eTazkira, Driving License (DL), Vehicle Registration (VR), Public Key Infrastructure (PKI), Bio-metrics, interconnection, infrastructure development, point-to-point network building, maintenance, and support and capacity building. The eNID architecture was developed jointly by the MCIT technical team, the national IT board of Afghanistan, the main contractor, and the Technical Assistance Team (TAT) from the international coalition. A *module* in this paper means a business module or a particular independent activity. It is neither considered a specific phase in the software development life cycle, nor a part of software development.

During the eNID project implementation, several challenges were experienced. There were several layers of subcontracts involving the main contractor and subcontractors (see Figure 1). In the contract document, there were several issues: the SLAs that the service provider was required to follow were not mentioned, penalties were not mentioned when the contractor failed to deliver the promised deliverable, ownership of the source code was not specified to be transferred to the project owner, the payments process method was based on time instead of deliverables, and there were no explicit provisions to address any conflict between the project owner and the contractor. Hierarchical communication and coordination among involved parties always took a long time. Such hierarchical communication was caused by a delay in deliverables (see Figure 1).

Conversely, several best practices were experienced during and after the handover process. The contract document was amended three times, and the required terms and conditions were added to the contract document. A technology-mediated, inter-organizational communication platform with simultaneous communication capability was proposed. As a result, if the business modules are independent and their integration and management are not a challenge, we recommend multi-sourcing the modules to expert vendors rather than dealing with sub-subcontractors. To overcome or at least reduce the mentioned challenges an appropriate language based on organizational patterns is proposed.

3. USING THE PATTERNS

This paper documents four patterns (highlighted blue in Figure 2) of recurring structures of the subcontract management system. We have observed these patterns in practice during the last decade while we were engaged with the eNID outsourced project. Iterative, creative, and recurring structures were being taken into consideration while documenting the mentioned patterns [Harrison(1999), Harrison et al.(2006), Meszaros and Doble(1997), Wellhausen and Fießer(2011)].

This study is based on our personal experience gained from working with the eNID project. As many studies confirm these issues in the available literature [Khan et al.(2009), Garrett(2005), Skjoett-Larsen(2000), Khan et al.(2011), Khan and Khan(2012), Mclaughlin and Peppard(2006), Guers et al.(2014)], we focused on the most appropriate, relevant, and accurate information, best practices and some proposed solutions. Several re-

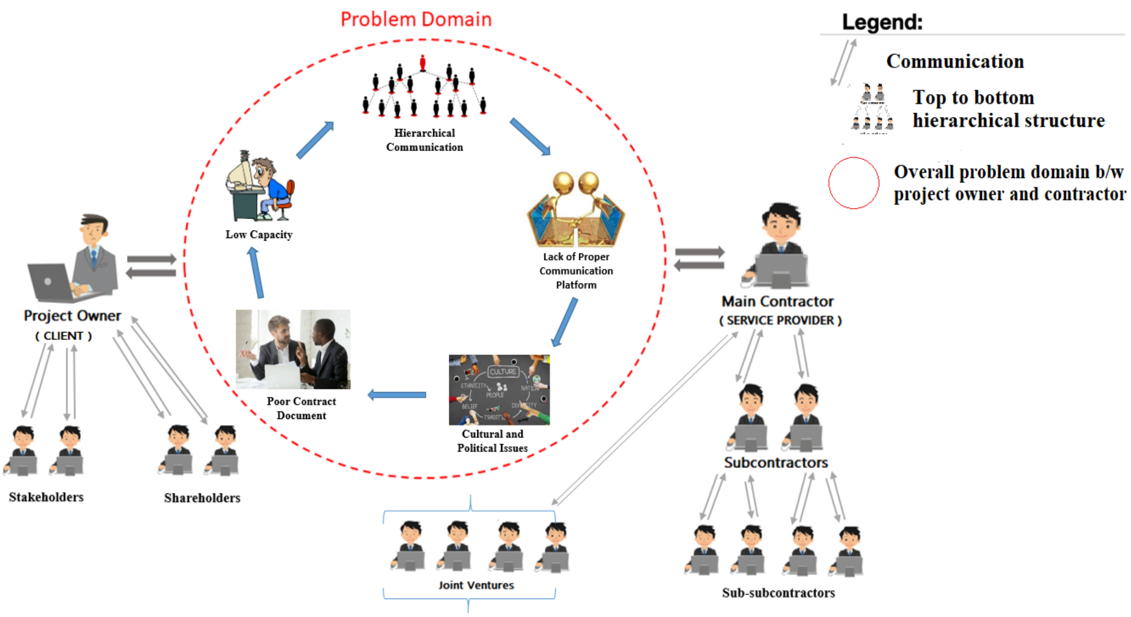

Fig. 1. Common problems observed during the study.

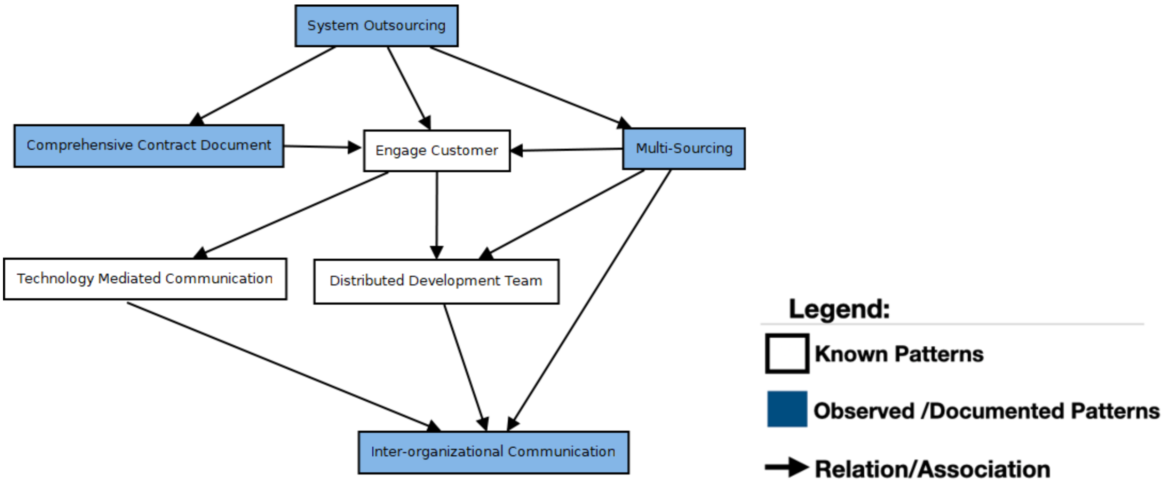
System Outsourcing
Comprehensive Contract Document
Engage Customer
Multi-Sourcing
Technology Mediated Communication
Distributed Development Team
Inter-organizational Communication
Legend:
Known Patterns
Observed /Documented Patterns
Relation/Association

Fig. 2. Organizational patterns of better relationships among stakeholders in system outsourcing.

searchers focus more on outsourcing a product mainly development a software rather than in-house development [Pahsa(2012), Johnstone et al.(2003), Handley and Benton Jr(2009), Miguel et al.(2009), Ali et al.(2020), Tervonen and Mustonen(2009), Krancher and Stürmer(2018)].

3.1 Story Behind the Patterns

Prior to documenting, it is worthwhile to explore the patterns in real-world practice. Let's tell the readers a short story about a public organization, MCIT of Afghanistan. MCIT had a system outsourcing contract with the private sector Firm ABC, an international IT vendor company with its headquarters in the UAE. The name is fake (for privacy reason), but the rest of the story is real. In the story, we mention the corresponding organizational patterns italicized in parentheses, see Figure 2). MCIT was facing a shortage of software developers and the required capacity. It was not just software development, but a large project (the entire solution) that included third-party software development, network expansion, hardware procurement, and installation. Furthermore, there was a need for third party software and hardware maintenance and support, and MCIT could not perform the mentioned tasks alone. Therefore, MCIT decided to outsource the whole solution to the private sector (System Outsourcing, Section 4).

Due to the involvement of several vendor organizations in distributed locations, there were usually misunderstandings in deliverables and a need for redevelopment. Thus, MCIT introduced individuals to each development team to work closely with them (Engage Customer [Coplien and Harrison(2004)]).

Furthermore, there were many problems like time consumption, misunderstanding, and wrong messages exchanged with the sub-subcontractor due to the hierarchical structure of the involved vendor companies, affecting the project. Finally, it was decided to bind separate contracts for each independent module[3] with relevant contractors based on their experience and expertise (Multi-Sourcing, Section 5).

Firm ABC had scattered offices for software development and providing other relevant services in several countries worldwide such as UAE, Korea, Malaysia, USA, South Africa and Afghanistan (Distributed Development Team [Waseeb et al.(2021)]).

The contract document completely favored the Firm ABC, and it was poorly drafted. There was no mention in the contract document of whether or not the main contractor is allowed to enter into further contracts with subcontractors. There was a need for an enriched and comprehensive contract document to clarify all terms and conditions (Comprehensive Contract Document, Section 6).

Managing a project that involves several stakeholders causes problems in communication and coordination [Calvo-Manzano et al.(2010)]. The above explanations indicate that a single deliverable always took much time due to the hierarchical structure of the organizations involved. There were different weekends' holidays and working hours due to scattered involved organizations in several regions worldwide. Besides different technology-mediated communication tools (Technology-Mediated Communication [Waseeb et al.(2021)]), there was a need for an enhanced communication platform to produce tickets for each task and provide parallel access for all involved organizations

Later, a technology-based enhanced communication platform for all involved organizations was developed (Inter-organizational communication, Section 7). Freshdesk was used for ticketing, while WhatsApp was used for instant, spontaneous, and other ad hoc communication. E-mails were used for formal communication and zoom for video conferencing. The ticketing system for each task was a huge success. This way, each stakeholder had an opportunity to monitor the progress of a specific task easily. Upon completing a task, the system closed the ticket.

3.2 Pattern Format

We expressed the patterns (to follow in Sections 4–7) in Coplien and Harrison's pattern format [Coplien and Harrison(2004)] with the conflict of the most prominent contradicting forces expressed in the *but* form proposed by Vranić and Vranić [Vranić and Vranić(2019)].

This is the format:

<**Pattern Name**>
An optional picture illustrates the pattern.
. . . the context in which the pattern occurs.

[3] A module in this paper is considered a business module or a particular activity rather than a specific phase in the software development life cycle or instead a part of a software development.

✣✣✣

The text in bold describes the actual problem as a conflict of the two most prominent contradicting forces.

Therefore:

Here, the text in bold describes the solution.

– An optional part with resulting consequences upon applying the given pattern.

✣✣✣

– Optional description to explain the pattern.

4. SYSTEM OUTSOURCING

. . . the system is significant because it is the whole solution. Due to various reasons such as lack of expertise, working domain knowledge, and the required resources, it is neither feasible nor efficient to continue with in-house development.

✣✣✣

Many organizations, especially those that are public, prefer to develop their own quality systems, but most of them lack the qualified employees, the necessary skills, and the capacity to develop their own quality systems since system development is not their main area of expertise.

Developing a complex system requires experts with proven domain knowledge and experience inside the organization. But building a local technical expert team is a challenging task. Technical experts are either in high demand or often unavailable in the local market.

Some organizations prefer to recruit qualified technical staff, but recruitment also takes a lot of time and might not be feasible for all organizations financially.

In-house development benefits from local communication, coordination, and first-hand access to the required information. Additionally, an organization benefits as the technical team's capacity steadily increases with in-house development. But there is always a risk of project failure due to lack of technical skills, hidden costs, unrealistic estimated schedule, poor design, and documentation. And often, small technical teams experience a delay in deliverables [Ali et al.(2020), Tervonen and Mustonen(2009), Khan et al.(2011)].

Due to privacy concerns, clients outsource system or software development to a trusted vendor. Each organization prefers to outsource its system to a trusted vendor. But often, the trustworthy vendors come with more expensive proposals.

Therefore:

Outsource development of the system (the entire solution) to a reliable, trustworthy, and professional vendor to achieve a secure, reliable, and satisfactory system. This approach allows an organization to concentrate on its primary business processes.

For a better system outsourcing, write down a comprehensive and a rich contract document to clarify whether the vendor is allowed to bind further contracts with subcontractors or not (Comprehensive Contract Document, Section 6). Support efficient engagement with the customer through multiple channels using several communication techniques and tools such as version control, blogs, wikis, forums, emails, Microsoft teams, Webex, and so on (Technology-Mediated Communication [Waseeb et al.(2021)]). If the involved organizations are located in several regions facing different cultures and working hours (Distributed Development Team [Waseeb et al.(2021)]), use a communication platform that contains the ticketing system (Inter-organizational Communication, Section 7).

✣✣✣

If software outsourcing is appropriately managed, it has many advantages, such as efficiency gains, cost savings, enhanced flexibility, access to superior expertise [Kremic et al.(2006), Hodosi and Rusu(2008)]. Others have also observed that the vendor company develops the software better, cheaper, and faster [Khan et al.(2009), Tervonen and Mustonen(2009), Khan et al.(2011)].

Several organizations outsource their systems (whole solutions) to other organizations that are more domain experts such as the National Database and Registration Authority (NADRA)[4] of Pakistan.

Nigeria outsourced its National Identity Document (NID) system to NADRA. The service provider developed the system in Nigeria. The system is integrated with the world's leading Advanced Braking Indication System (ABIS) system and is used effectively for the issuance of identities. In this project, a back-end system for the national ID card is implemented, the national identity management portal is created, and AFIS integration, production and front-end enhancements are implemented.

Bangladesh outsourced their driving licence system to NADRA. The vendor has developed a comprehensive driving licence system and provides regular consulting services to Bangladesh. And in 2016, Fiji Elections Office (FEO), an organization of Fiji outsourced the election management system to NADRA. This system strives to deliver better and faster decision-making while protecting the voting rights of Fijian civilians.

Sudan outsourced their civil registration system to NADRA. The civil registration system was designed for the registration and tracking of vital events such as birth, death, marriage and divorce status. This was done in the centralized national database of the Sudanese population. And Kenya outsourced their electronic passport system to NADRA. The vendor developed a comprehensive system that contains advanced features such as PKI, RFID chip, biometrics, fingerprints, IPI-invisible personal ID, 2D bar-code and machine readable zone. As a result of the above explanations, the system outsourcing pattern is applicable where the entire solution is needed. It is recommended to concentrate more on the organization's core business, outsource the software to a trusted vendor, and not compromise privacy in order to save some money.

The planned outsourced system is huge. It is a complete solution that contains several independent business modules. It requires multi-vendors (multi-sourcing).

5. MULTI-SOURCING

. . . a large system (the whole solution) is needed to be outsourced. The system has the ability to be divide in smaller and more manageable independent business modules. The integration of all business modules is not a serious issue. Meanwhile, false information interchange, delayed deliverables, hierarchy-based communication, and adding some percentage of each vendor's net profit in a hierarchical vendor structure are the main reasons toward multi-sourcing.

✣✣✣

The hierarchical vendor structure may be easier for a client to integrate and manage tasks easily. But their hierarchical communications are slow and result in increased costs because of their chain of command. Consequently, false messages and information may reach the bottom-level sub-subcontractor, causing delays in deliverables and sometimes the failure of projects.

Single source contract management is efficient and effective for small systems. But large projects or a whole solution containing several business modules require several vendors based on their domains of expertise.

The hierarchical contract management system is more manageable because a client deals only with a single vendor. But often, it forwards false instructions to the lowest-level subcontractor. This way, the problem causes a delay in the deliverable or even failure of the system.

Single sourcing has high indirect costs and is risky due to its dependency on a single service provider. But multi-sourcing has lower indirect costs and is flexible to uncertain events as clients rely on more than one vendor.

[4] https://www.nadra.gov.pk/international-projects/

A hierarchical or multi-layered subcontract management system involves several vendors. The vendors may have the required expertise and knowledge of their working domain, but each vendor adds some percentage of their net profit that effects quality of the system.

Others have also observed that multi-layered contract management system suffers from the product [Ali et al.(2020), Miguel et al.(2009), Dhar and Balakrishnan(2006), Fabriek et al.(2008), Alam et al.(2012), Calvo-Manzano et al.(2010), Rao(2009)].

Therefore:

Assign different business modules to various service providers based on their domain expertise; if the modules are independent, managing several contractors and integrating business modules are not serious issues for the project owner. Sign separate contracts with each service provider (multi-sourcing).

A single source contract deals with a single organization and can be the most efficient solution for small system development. Nevertheless, multi-sourcing is better for bigger systems that contain independent business modules and the integration of mentioned modules should not be a serious issue for the project owner organization. The current outsourced system is a whole solution (big system) that contains several independent business modules. So, there is a risk of system failure dealing with a single service provider. Because a single service provider may not have all the required expertise required to develop a whole solution.

Multi-sourcing will bridge the gap of misunderstanding and hierarchical communication that results in false messages being exchanged and is time-consuming. Furthermore, failure in a single module may not cause a failure of the whole solution.

✣✣✣

Multi-sourcing is an appropriate alternative to a multi-layered contract management system. It is the future of outsourcing. It is proven to be a significant cost saving and decreases strategic risk [Cohen and Young(2006)]. Multi-layered or hierarchical contract management systems also have many vendors that are usually unsuccessful, and communication takes a lot of time [Rao(2009), Alam et al.(2012), Ali et al.(2020), Calvo-Manzano et al.(2010), Fabriek et al.(2008), Miguel et al.(2009), Dhar and Balakrishnan(2006)]. But multi-sourcing secures the system from being a failure, as multi-sourcing increases flexibility and quality [Goldberg et al.(2014), Cohen and Young(2006), McFarlan and Nolan(1995), Dhar and Balakrishnan(2006), Hodosi and Rusu(2008)].

In addition, multi-sourcing has the advantage of a choice among “best of breed” vendors, with lower costs resulting from vendor competition and enhanced agility and adaptability to the dynamic environment [Cohen and Young(2006)]. Some authors [McFarlan and Nolan(1995), Dhar and Balakrishnan(2006)] focus more on multi-sourcing instead of hierarchical or multi-layered outsourcing.

It is a general fact that integration and management of several vendors may be a difficult job for some project owners in multi-sourcing to deal with. Nevertheless, multi-sourcing is more applicable where there are several independent business modules and there is a need for different service provider vendors to have different relevant experiences and domain knowledge. Furthermore, integration and management of several vendors should not be a big issue for the project owner or client. Generally, as long as the verifiable metric is not dependent on the client’s efforts, multi-sourcing works better than single sourcing, but only if the project outcome and the verifiable metric do not align (if they align, then multi-sourcing strategies perform equally well). The single-sourcing strategy performs more effectively if the verifiable metric depends on vendor and client efforts [Bhattacharya et al.(2012)].

Crowdsourcing Software Development (CSD) has emerged as a new software development paradigm [Zhang et al.(2020)]. TopCoder is a famous example of CSD that contains 1.5 million-plus community members[5] who take the clients’ requirements and break them down into small manageable chunks of work. TopCoder works with a client company to define the modules and integrate the solutions [Mao et al.(2017), Zanatta et al.(2018)].

[5] https://www.topcoder.com/thrive/articles/what-is-topcoder

To avoid future conflicts and clarify the authorities and responsibilities of all stakeholders it is essential to have a comprehensive contract between all the parties involved.

6. COMPREHENSIVE CONTRACT DOCUMENT

. . . during outsourcing the system, it is required to cover more details than just signing the contract. It is the right time to discuss and disclose the required Service Level Agreements (SLAs), and/or terms and conditions. Binding documents under contract is neither feasible nor efficient without clarifying whether the principal contractor can or cannot bind further contracts with subcontractors.

✣✣✣

The system outsourcing agreement stipulates that all involved parties will sign a contract document. Still, the poorly designed arrangements cause a delay in delivering the deliverables and lead to blame games between the project owner and the service provider.[6]

A common recurring challenge is further contracting a product by a vendor, but usually, it is not mentioned in the contract document whether the vendor can or cannot bind further contracts with subcontractors.

During the outsourcing of a system, a written contract document is required among stakeholders. But often, the agreements do not contain all the required terms of conditions.

Usually, the involved parties write the international contracts in one of the international languages. But often, the local employees do not know either the international language or the exact meaning of the technical terms and conditions used in the contract.

Writing an adequate contract requires experts with proven experience and domain knowledge. But creating a team of experts (lawyer, procurement, and technical) for developing a contract in a small organization with such competencies is challenging. Such challenges usually include a lack of experts in the local market, high demand, or organizations often do not prefer to recruit many employees.

Therefore:

Write down a comprehensive contract document in both international and local languages. Include all the relevant terms, conditions, and the required SLAs. Verify the agreement by either internal or external lawyers, technical, procurement, and financial experts before the official signing of the contract.

A service level agreement describes the service level that a client expects from his or her vendor. A performance management plan contains metrics that are used to measure the quality of services and penalties for non-performance. Contracts with technology vendors must include this clause. Many contract delays are caused by incorrectly scheduled system timelines. It is common for external service providers to fail to deliver on time, communication between project participants to be inadequate, and unpredictable external factors, such as natural disasters, to occur as a result of variations in project scope. The project deadline does not match the project goals and deliverables.

Continuing without a valid and comprehensive contract document is neither feasible nor efficient. To reduce the upcoming threats posed by outsourcing, it is essential to create a well-written contract document. Make sure the contract is translated into both languages. Each page should be signed and stamped with the page number by both clients and vendors.

✣✣✣

Other researchers stress the importance of developing comprehensive contract documents and proper contract management. It should contain all SLAs [Dhar and Balakrishnan(2006), McFarlan and Nolan(1995)]. Be clear and specific about the contract before signing it. The system process should be tested and bench-marked before

[6] The poorly designed arrangements refer to lack the required terms in conditions inside the text of a contract; and the blame games between the project owner and the service provider might be conflicts of payments, deadline, quality of deliverables, etc.

outsourcing [Hite(2003)]. There should be a well-detailed plan before signing the contract. Reference should be made to the available resources, competencies, and costs of the future terms and SLAs [Hite(2003)].

There may be advantages to a multi-layered or hierarchical subcontract management system. In addition, it contributes to a lack of communication and incorrect instructions being sent down to bottom-level developers. Additionally, each level contractor adds some benefits and costs, which negatively impact the quality of the final product.

Different vendors provide parts of the product to large enterprises. The stakeholders such as joint ventures, subcontractors or even sub-subcontractors might be located in various geographical regions worldwide. Additionally, there might be different time zones, cultures and holidays at weekends. Therefore, it would neither be feasible nor efficient to have a joint physical or virtual meeting scheduled at the same time to have all entities and participants present. Therefore, an enhanced inter-organizational communication platform is required.

7. INTER-ORGANIZATIONAL COMMUNICATION

. . . the system has been outsourced, and a first-hand contract was signed between the client and the vendor. The vendor has signed further contracts with scattered subcontractors, sub-subcontractors, and joint ventures. Control, coordination, and collaboration require strong, accurate, and on-time mass communication with each involved organization simultaneously. And each task needs to be assigned to a specific organization and monitored.

✣✣✣

The project owner, main contractor, and subcontractors are located in different geographical regions with different cultures and time zones. They still work towards a common goal, but there are spatial and temporal distances. This could be because there is a vast difference in working hours and different weekend holidays according to their duty stations. Therefore, attending a joint meeting at a certain time for all is challenging.

Based on responsibilities and chain of command, it is essential to communicate issues in a hierarchical form, but it takes much time and will cause deliverables to go beyond the timeline.

Face-to-face communication during outsourced project implementation enhances trust, project success, and reduces information risks. But face-to-face communication is not feasible due to the scattered nature of the involved organizations in different regions.

Therefore:

Create an inter-organizational technology-mediated platform to communicate with all stakeholders simultaneously. Additionally, develop a ticketing system inside the platform to assign responsibilities to each organization and make sure the progress on each task will be visible to all stakeholders.

Utilize technology-supported communication by engaging rich tools for conferencing, messaging, and collaboration.

A technology-mediated platform or tool like Freshdesk, Bugzilla, Jira, or version control systems like CVS, SVN, and Git should be used. These platforms and tools are available all the time. All involved organizations can work as a single team towards a common goal remotely. Reference [Dhar and Balakrishnan(2006)] also proposes such best practices.

This pattern is closely related to (Technology-Mediated Communication [Waseeb et al.(2021)]). All organizations involved in communication have equal access to the circle communication platform, but the newly discovered pattern has extra features. Additionally, it adds a ticket system, visibility to all stakeholders, and simultaneous communication, and is viewed as being used between multiple organizations rather than within a single organization.

The Inter-organizational Communication pattern proposes a technology-based platform. All involved organizations such as client, vendor, subcontractor, sub-subcontractor, and joint ventures have equal access to the platform. Using the platform, all information and instructions can be distributed to all organizations. The platform contains a ticketing system that assigns relevant tasks to specific organizations. Thus, all involved organizations can comment and will be able to monitor the progress on each deliverable.

8. DISCUSSION

Verification of the newly discovered patterns takes place with the help of pattern experts from the community. We plan and arrange writers' workshops on pattern mining with our supervisors, students and other patterns experts from the community. As a result of the workshops mentioned above, patterns and their chunks get discussed and improved according to the proposed recommendations. Furthermore, we take part in international conferences and workshops on pattern mining, such as Pattern Mining Workshop (PLoPourri events within PLoP 2022)[7]. This way, the newly discovered patterns will get further enhanced.

In Section 2, we discuss the challenges encountered during the eNID study's implementation of an outsourcing project that encompassed the entire solution. The most common problem domain in such a hierarchical vendor system that contains subcontractors, sub-subcontractors, and joint ventures is mentioned in Figure 1.

To overcome these challenges, we have come up with the following results and recommendations:

(1) If a product (software or a business module) is small, it should be single-sourced to a competent, experienced, and domain-knowledge expert vendor.
If tasks are integrated, the choice of sourcing model becomes more complex. The appropriate choice depends on the alignment between performance metrics and project revenue, the verifiability of project revenue, and moral hazard. Firms prefer single-sourcing when the verifiable performance metric and project revenue are perfectly aligned, or if the project's revenue is completely verifiable [Bhattacharya et al.(2018)].

(2) If a system or a whole solution is huge, containing several independent business modules. In addition to allowing the system owner (client) to integrate and manage the outsourced project with multiple vendors, the project should be multi-sourced to a number of domain knowledge experts based on their expertise. Companies that feel comfortable with the outsourcing paradigm are increasingly looking to multi-sourcing as a collaborative, inter-organizational approach to value creation.
The practice of multi-sourcing, in which best-of-breed IT services are stitched together from multiple, geographically dispersed service providers, represents the cutting edge of modern organizational structures. Although major advances have been made in the literature on information systems (IS) and strategic management in the last ten years, the focus has been on client-vendor relationships. It is demonstrated that an extrapolation of such a dyadic relationship does not address the complex linkages between incentives, effort, and output that emerge when multiple vendors, who are competitors, cooperate and coordinate with the client to accomplish the client's goals [Bapna et al.(2010)].
If the relative effectiveness of multi-sourcing and single-sourcing can be determined by identifying the effects of three factors: task modularity, how closely a performance metric aligns with project revenue, and how easily project revenue can be verified. As long as the verifiable performance metric and project revenue are not completely aligned, multi-sourcing strictly outperforms single-sourcing when tasks are modular [Bhattacharya et al.(2018)].

(3) If a system or a whole solution is huge, containing several independent business modules. And the system owner (client) does not have the ability to integrate and manage the outsourced project with several modules. In order to have a quality product, it should be single-sourced, and several domain knowledge vendors may not be able to provide all capabilities. In such a case, the vendor can further outsource some business modules to its subcontractors, joint ventures, etc. This way, the client will focus more on the quality of the deliverables and the integration of modules and managing tasks will be the responsibility of the main contractor. But communication between different vendors and on-time delivery remains challenging.

[7] https://hillside.net/home/news/335-pat-min-workshop

Subcontracting has both advantages and disadvantages. In well-managed subcontracting projects, contractors (vendors of outsourcing projects) can get low-cost services without sacrificing quality and they can concentrate on what makes them superior. However, if the subcontracting is excessive or misused, it can also cause many negative impacts, such as problems with software quality and security, non-payment, corruption, etc. [Wang and Shi(2009)].

(4) It was mentioned above that an inter-organizational technology-mediated communication platform is recommended to provide parallel access to all involved vendors. This will also reduce misunderstandings and delays in messages or information exchange by establishing hierarchical communication. The reference [Dhar and Balakrishnan(2006)] also proposes such best practices. More information about the framework is available in Section 7.

(5) The contract document should be enriched, comprehensive and contain all the required service level agreements in all cases. The contract document should be able to provide clear clarification for any upcoming unexpected conflicts. The contract document should address both predictive and non-predictive issues, assure disaster recovery, and prevent blame games between parties. In addition to proper contract management, other researchers emphasize the development of comprehensive contract documents. It should include all service level agreements [Dhar and Balakrishnan(2006), McFarlan and Nolan(1995)]. Before contract signing, the process of outsourcing should be analyzed and bench-marked, and accompanied by a well-detailed plan [Hite(2003)].

Clearly, the common results of the framework are less or more useful in each of the two particular cases mentioned above. However, a common outcome of the framework would best fit all of these cases and solve the whole issue, so more study and research are needed.

9. RELATED WORK

Pavan Kumar et al. presented a workbench, as a pattern language, based on their experience and review assignments. This work identifies patterns, anti-patterns, and proposes tools appropriate to vendors and recommends a workbench environment [Maddali et al.(2006)].

Andreas Daniel et al. discovered three patterns for communication between multiple parties. As a result, they encapsulate the most common patterns for establishing secure communication over an insecure channel without external interference. The discovered patterns are common explanations of the fundamental communication protocol necessary to create secure communication channels. They provided a helpful starting point to define what pattern should be used while designing a secure communication system and which standard protocol can be implemented to achieve assured messaging [Sinnhofer et al.(2016)].

Waseeb et al. mined six organizational patterns for distributed software development. They relate them to each other and to other known patterns establishing a pattern language for the organization of distributed software development to overcome the existing problems of sitting product dispersed teams [Waseeb et al.(2021)].

Duan Weihua's research shows the kinds of outsourcing relationships based on the type of relationship and the level of the project. It also proposes the relationship patterns model of system outsourcing on the enterprise scale of two parties. In addition, a system outsourcing relationship evolution paths model was developed [Duan(2018)].

Dahar et al. demonstrates that they identified the principal risk factors and best practices in global system outsourcing. In addition, they delved into some crucial issues on system outsourcing, particularly the challenges and benefits. Finally, they presented case studies of two global outsourcing organizations and validated some of the claims made by previous researchers on system outsourcing [Dhar and Balakrishnan(2006)].

Gantman and Fedorowicz analyzed data from 82 collaborative projects to determine whether public-sector IT outsourcing (ITO) practices match the practices that researchers have documented in the private sector. They have focused on the ITO decision process components and outcomes identified as challenging for public-sector ITO. [Gantman and Fedorowicz(2020)].

Peng Ben-hong and Zong Qi propose multi-sourcing if there is much task conflict. The system owner should choose several vendors to complete the development of a complex module [Ben-hong and Qi(2009)].

Goles' research indicates that the efficient system outsourcing relationship has an essential influence on the success of enterprise system outsourcing [Goles(2001)].

Siffatullah Khan and Abdul Wahaid proposed an Outsourcing Contract Management Model (OCMM) to assist vendor organizations in successful management and execution of the contract in pre-contract, during the agreement and post-contract administration [Khan and Khan(2012)]. Unlike the work proceeding in the above paragraphs, this paper presents a set of four newly discovered organizational patterns; the sequence of the four patterns and relevant practices from the literature describes a pattern language. It bridges the gap of communication and coordination among involved parties in an outsourced system platform. This paper recommends multi-sourcing for large projects instead of a multi-layered contract management system and emphasizes developing rich contract documents.

10. CONCLUSION AND FUTURE WORK

Despite enormous efforts in managing outsourced systems, currently, existing solutions do not seem to be sufficient. Individual patterns fragment into fragmented patterns that do not form coherent patterns to address organizational working joint platforms that include all involved organizations. For example, the client, vendor, subcontractor, and joint venture. Moreover, due to their geographical locations, unfamiliarity with technology-mediated communication technologies, and poor contract design, they are disconnected.

We propose four organizational patterns that will improve relationships among stakeholders in subcontract management of system outsourcing to overcome the above issues. We relate them to each other and other well-known patterns and practices, thereby creating a pattern language for the joint working platform of all involved entities. Based on the overall idea of a pattern language, we can construct patterned stories about actual active organizations.

Some additional organizational best practices for multi-agent collaborative system outsourcing can be observed in the public sector of developing countries.

ACKNOWLEDGMENTS

We want to thank our shepherd, Cecilia Haskins, for her enthusiasm and insightful comments. Moreover, we would like to express our gratitude for Kyle Brown's valuable feedback on early versions of this article. Furthermore, we would like to thank Michael Weiss for moderating and providing helpful comments on our paper at PLoP 2022. Last but not least, we would like to thank Eduardo Guerra and Mary Lynn Manns for their effort in organizing the PLoP 2022 conference.

The work reported here was supported by the Scientific Grant Agency of Slovak Republic (VEGA) under grant No. VG 1/0759/19, by the Operational Programme Integrated Infrastructure for the project: Research in the SANET network and possibilities of its further use and development (ITMS code: 313011W988), co-funded by the European Regional Development Fund (ERDF), and by the Slovak Research and Development Agency under the contract No. APVV-16-0213.

REFERENCES

Asad Ullah Alam, Siffat Ullah Khan, and Irshad Ali. 2012. Knowledge sharing management risks in outsourcing from various continents perspective: a systematic literature review. *International Journal of Digital Content Technology and its Applications* 6, 21 (2012), 27.

Sikandar Ali, Jiwei Huang, Siffat Ullah Khan, and Hongqi Li. 2020. A framework for modelling structural association amongst barriers to software outsourcing partnership formation: An interpretive structural modelling approach. *Journal of Software: Evolution and Process* 32, 6 (2020), e2243.

Ravi Bapna, Anitesh Barua, Deepa Mani, and Amit Mehra. 2010. Research commentary—Cooperation, coordination, and governance in multisourcing: An agenda for analytical and empirical research. *Information Systems Research* 21, 4 (2010), 785–795.

Peng Ben-hong and Zong Qi. 2009. Modular Outsourcing: Pattern and Selection. 1 (2009), 358–361.

Shantanu Bhattacharya, Alok Gupta, and Sameer Hasija. 2012. Single sourcing versus multisourcing: The role of effort interdependence, metric-outcome misalignment, and incentive design. (2012).

Shantanu Bhattacharya, Alok Gupta, and Sameer Hasija. 2018. Single sourcing versus multisourcing: The roles of output verifiability on task modularity. *MIS Quarterly* 42, 4 (2018), 1171.

Jose A Calvo-Manzano, Tomás San Feliu, and Andreas Pursche. 2010. The forgotten practices of subcontracting. (2010), 1–4.

Linda Cohen and Allie Young. 2006. *Multisourcing: Moving beyond outsourcing to achieve growth and agility*. Harvard Business Press.

James O Coplien and Neil B Harrison. 2004. *Organizational patterns of agile software development*. Prentice-Hall, Inc.

Subhankar Dhar and Bindu Balakrishnan. 2006. Risks, benefits, and challenges in global IT outsourcing: Perspectives and practices. *Journal of Global Information Management (JGIM)* 14, 3 (2006), 59–89.

Weihua Duan. 2018. Types, patterns and evolution paths of IT Outsourcing Relationship. (2018), 232–237.

Matthias Fabriek, Mischa van den Brand, Sjaak Brinkkemper, Frank Harmsen, and Remko Helms. 2008. Reasons for success and failure in offshore software development projects. (2008).

Sonia Gantman and Jane Fedorowicz. 2020. Determinants and success factors of IT outsourcing in the public sector. *Communications of the Association for Information Systems* 47, 1 (2020), 14.

Gregory A Garrett. 2005. Managing complex outsourced projects. (2005).

Marius Goldberg, Axel Kieninger, and Hansjörg Fromm. 2014. Organizational models for the multi-sourcing service integration and management function. 2 (2014), 101–107.

Timothy Goles. 2001. *The impact of the client-vendor relationship on information systems outsourcing success*. University of Houston.

Hugo Gong, Mary Tate, and Shawn Alborz. 2007. Managing the outsourcing marriage to achieve success. (2007). `https://eprints.qut.edu.au/114791/`

Constance Guers, Christophe Martin, and Jean-Luc Wybo. 2014. The impact of the use of subcontracting on organizational reliability and safety. , 1027-1035 - ISBN 9781138026810 pages. `https://hal-mines-paristech.archives-ouvertes.fr/hal-01063681`

Sean M Handley and WC Benton Jr. 2009. Unlocking the business outsourcing process model. *Journal of operations management* 27, 5 (2009), 344–361.

Neil B Harrison. 1999. The language of shepherding. *Pattern languages of program design* 5 (1999), 507–530.

Neil B Harrison et al. 2006. Advanced Pattern Writing Patterns for Experienced Pattern Authors. (2006).

R Hite. 2003. *Information Technology-DOD needs to leverage lessons learned from its outsourcing projects*. Technical Report. GAO-03-371, Report to the Subcommittee on Readiness and Management Support . . . .

Georg Hodosi and Lazar Rusu. 2008. Information technology outsourcing: A case study of best practices in two Swedish global companies. *MCIS 2008 Proceedings* (2008), 7.

Richard Johnstone, Claire Mayhew, and Michael Quinlan. 2003. Outsourcing Risk? The Regulation of Occupational Health and Safety Where Subcontractors are Employed. 22 (01 2003).

Siffat Ullah Khan and Abdul Wahid Khan. 2012. Outsourcing Contract Management Model (OCMM). (Engslish) [On Outsourcing Contract Management Model]. *ResearchGate* (2012). `DOI:https://doi.org/publication/235768113`

Siffat Ullah Khan, Mahmood Niazi, and Rashid Ahmad. 2009. Critical Barriers for Offshore Software Development Outsourcing. , 79-86 pages. `DOI:https://doi.org/10.1109/APSEC.2009.16`

Siffat Ullah Khan, Mahmood Niazi, and Rashid Ahmad. 2011. Barriers in the Selection of Offshore Software Development Outsourcing Vendors: An Exploratory Study Using a Systematic Literature Review. *Inf. Softw. Technol.* 53, 7 (jul 2011), 693–706. `DOI:https://doi.org/10.1016/j.infsof.2010.08.003`

Oliver Krancher and Matthias Stürmer. 2018. Explaining Multisourcing Decisions in Application Outsourcing. , 122 pages.

Tibor Kremic, Oya Icmeli Tukel, and Walter O Rom. 2006. Outsourcing decision support: a survey of benefits, risks, and decision factors. *Supply Chain Management: an international journal* (2006).

Pavan Kumar Maddali, Satish Chandra, and Shankar Kambhampaty. 2006. Pattern Oriented Workbench for IT Service Providers. (2006), 1–12.

Ke Mao, Licia Capra, Mark Harman, and Yue Jia. 2017. A survey of the use of crowdsourcing in software engineering. *Journal of Systems and Software* 126 (2017), 57–84.

Emmanuel Marilly, Olivier Martinot, Hélène Papini, and Danny Goderis. 2002. Service level agreements: a main challenge for next generation networks. (2002), 297–304.

F Warren McFarlan and Richard L Nolan. 1995. How to manage an IT outsourcing alliance. *MIT Sloan Management Review* 36, 2 (1995), 9.

Des Mclaughlin and Joe Peppard. 2006. It back-sourcing: from make or buy to bringing it back in-house. *Proceedings of the 14th European Conference on Information Systems, ECIS 2006* (01 2006), 1735–1746.

Doble J Meszaros and Jim Doble. 1997. G. A pattern language for pattern writing. 131 (1997), 164.

Jiménez Miguel, Mario Piattini, and Aurora Vizcaino. 2009. Challenges and Improvements in Distributed Software Development: A Systematic Review. *Advances in Software Engineering* 2009 (04 2009). DOI:https://doi.org/10.1155/2009/710971

A. Pahsa. 2012. *Systems Engineering and Subcontract Management Issues*. 297–309. DOI:https://doi.org/10.5772/32841

Nagesh Mukunda Rao. 2009. Challenges in execution of outsourcing contracts. (2009), 75–79.

Andreas Daniel Sinnhofer, Felix Jonathan Oppermann, Klaus Potzmader, Clemens Orthacker, Christian Steger, and Christian Kreiner. 2016. Patterns to establish a secure communication channel. (2016), 1–21.

Tage Skjoett-Larsen. 2000. Third Party Logistics — From An Interorganizational Point of View. *International Journal of Physical Distribution and Logistics Management* 30 (03 2000), 112–127. DOI:https://doi.org/10.1108/09600030010318838

Jeff Sutherland and James O Coplien. 2019. *A Scrum book: The spirit of the game*. Pragmatic Bookshelf.

Ilkka Tervonen and Timo Mustonen. 2009. Offshoring Test Automation: Observations and Lessons Learned. , 226-235 pages. DOI:https://doi.org/10.1109/ICGSE.2009.30

Valentino Vranić and Aleksandra Vranić. 2019. Drama Patterns: Extracting and Reusing the Essence of Drama. (2019).

Yi Wang and Huihui Shi. 2009. Software outsourcing subcontracting and its impacts: an exploratory investigation. 1 (2009), 263–270.

Shakirullah Waseeb, Waheedullah Sulaiman Khail, and Valentino Vranic. 2021. Establishing a Pattern Language for the Organization of Distributed Software Development. (2021), 1–9.

Tim Wellhausen and Andreas Fießer. 2011. How to write a pattern? A rough guide for first-time pattern authors. , 9 pages.

Alexandre Lazaretti Zanatta, Leticia Machado, and Igor Steinmacher. 2018. Competence, collaboration, and time management: barriers and recommendations for crowdworkers. In *2018 IEEE/ACM 5th International Workshop on Crowd Sourcing in Software Engineering (CSI-SE)*. IEEE, 9–16.

Zhenyu Zhang, Hailong Sun, and Hongyu Zhang. 2020. Developer recommendation for Topcoder through a meta-learning based policy model. *Empirical Software Engineering* 25, 1 (2020), 859–889.