Journal of Computing and Information Technology - CIT 22, 2014, 4, 267–275
doi:10.2498/cit.1002344

An Empirical Study of Offshore Software Development: the Case of a Ticketing Application

Carlo Consoli[1], Paolo Rocchi[1,2] and Paolo Spagnoletti[2]

[1] IBM, Roma, Italy
[2] LUISS Guido Carli University, Roma, Italy

This is an industry report about a software development project that included a local team and an offshore team. Both teams contained highly qualified experts who had been selected before the project start up. After a year, which could be deemed as a break-in period, the project leaders observed that the quality and quantity of the software modules produced by the two teams were not up to expectation while costs had gone up. The management monitored the performance of the two groups by analysing the tests undertaken to ensure correctness of the modules. They reached the conclusion that the project failure was influenced by cultural differences between the onshore and offshore teams. The management became convinced that the current work-organization prevented knowledge delivery and knowledge acquisition, so they established a new organization as the solution to this problem. A rapid validation of the new work arrangement convinced all the involved experts that the diagnosis was right.

Keywords: global software development, offshore insourcing, large software project, project management

1. Background

The concept of global outsourcing started in the late 1980s when leading technology companies discovered Asian countries as untapped sources of high tech professionals at substantial lower labour costs. The increased acceptance of offshore outsourcing with IT departments and software vendors encouraged those firms to build excellence centres, which supplemented software production in offshore localities at comparatively inexpensive disbursal. Offshore software development outsourcing became a global delivery model through geographically dispersed companies and even went into effect as a modern business strategy for producing software at low costs. However, in a few years, managers of both outsourcing and vendor firms realised that cost considerations, generally assumed to be the main reason for offshore outsourcing, needed to be balanced with the focus on maintaining quality [1].

Since the birth of the computer era, experts have recorded high rates of failed or troubled IT projects. Various negative factors result in the software failures. "Bad communication" between the members of an IT project towers as one of the most significant factors. The surveys conducted in this area show that fifty to thirty per cent of project failures derive from insufficient or misleading exchange of information [2]. Offshoring aggravates this aspect of software production, as distance is a well-known barrier to communication. Time differences also interfere with interpersonal rapports when people operate in remote geographical areas. At last, a few software projects are only technical. Software specifications deal with the customer environment besides computing, and project managers involve professionals from very different educational and work background [3]. Within this scenario, *knowledge transfer processes* (KTP) turns out to be a core question. Knowledge transfer has become a critical argument in companies and institutions. Drucker holds "knowledge is information that changes something or someone either by becoming grounds for action, or by making an individual or group of individuals capable of different or more effective action" [4].

This statement written two decades ago is more current now than before, especially in offshore projects. Empirical investigations demonstrate the importance of KTP. Survey [5] conducted on modern technical literature identifies 22 unfavourable success factors (CSF) for vendors who operate in offshore projects and finds four 'human factors' that rank the first twelve levels of the CSFs classification. The authors hold that "Skilled Human Resources" ranks as the second level in important factors of success; "Efficient Outsourcing Relationships Management" ranks the fifth level; and "Knowledge of the Client Language and Culture" and "Knowledge Transfer" rank as the tenth and twelfth levels, respectively. Global work involves people who speak different languages, who have different ways of working and communicating and who understand a variety of cultures: all this constitutes a growing communication challenge for KTPs [6]. The survey [7] casts light on the performances of joined Western and Asian software development teams, which are influenced by inherent cultural disparities and differences in the structures of the relevant national software communities. Vendors demonstrate responsiveness to the knowledge transfer issue [8] and 70 project managers engaged in offshore projects underlining the importance of good knowledge management [9]. Some researchers tackle the KTP argument from the technical stance and suggest the use of standard development methods and tools [10] [11].

Experts attend the human factors in order to optimize KTPs [12][13][14]. The present study shares this stance and makes a condensed report on an industry case where the impact of knowledge management turned out to be crucial for the success of the project. This report begins with presenting the control of software production. The next section gives an account of additional investigations undertaken to understand apparent and unexpected failures provoked by KTP issues. Finally, the report explains the new organisation of the project and its validation in relation to human communication and cooperation.

2. Software Development Monitoring

An Italian company that transports goods and passengers by railways – herein called *IT Client* – signed an outsourcing project with IBM – herein called the *IT Provider* – for developing a *ticketing application*. The aim of the project was to redefine and expand the entire ticket trading system of the client. The project began in late 2009 and two teams – herein called *A* and *B* – were in charge of the software development. The former was the *offshore team*, which included a variable number of programmers living in India: from 50 to 70, depending on the workload. The latter was the *domestic team* consisting of 20 experts involved in the analysis phase.

The following groups of software practitioners carried out the operations:

1. The *Development Team* including A and B.
2. The *Test Team* arranged and executed test cases and discovered software errors.
3. The *Release Team* registered all the functions implemented and tested. On request, it provided statistics on the work in progress. This team delivered the modules to the customer as soon as the modules of the ticketing application were ready.

During 2010, teams A and B did not produce significant outcomes. It may be said that 2010 was a break-in period. In early 2011, practitioners began to work steadily, and the project leaders perceived significant difficulties in the great amount of tests necessary to monitor the software quality. Managers overlooked the number and typology of defects, which are more telling on the technical plane, looked at the quantity and kind of tests.

The control system of the software production was arranged in the following manner. Developers subdivided the ticketing application into six *macro-functions* or *modules*. They devised **F** different functional tests for each module and executed the following number of functional tests per module:

$$\mathbf{FTN}\ (\text{functional test number}) = \mathbf{B} \times \mathbf{F}$$

where **B** is the number of compilations of a single module.

Developers progressively joined the modules and created five *builds*. For example, build #1234 included the modules from 1 to 4. Experts devised **S** regression tests for each build

and undertook this numbers of regression tests per build.

$$\textbf{RTN}\ (\text{regression test number}) = \textbf{S} \times \textbf{R}$$

where **R** is the number of times a build was compiled.

#	Module	B	F	FTN
1	Base Functions	73	63	4,599
2	Billing Functions	16	6	96
3	Advanced Ticket Purchase Options I	46	16	736
4	Advanced Ticket Purchase Options II	72	29	2,088
5	Ticket Purchase with Subscription	44	22	968
6	Advanced Ticket Purchase Options III	48	90	4,320
Total FTN				**12,807**

Table 1. Amount of functional tests achieved during the first semester in 2011.

# Build	S	R	RTN
12	5	63	315
123	15	69	1,035
1234	24	85	2,040
12345	15	114	1,710
123456	16	136	2,176
Total RTN			**7,276**

Table 2. Number of regression tests achieved during the first semester in 2011.

In conclusion, developers undertook

$$\begin{aligned}\text{Grand Total} &= (\text{Total FTN} + \text{Total RTN}) \\ &= (12,807 + 7,276) = 20,083\end{aligned}$$

This value was determined by the enormous amount of errors found in the software modules, and it contradicted the profiles of the skilled developers belonging to A and B. The high number of tests strengthened the idea that something was wrong and resulted in the low performance of the offshore and domestic teams. Managers settled to investigate the reason of this failure by means of further inquiry.

Table 3 exhibits the states of software defects classified according to the common terminology. Defects classified as *In Progress*, *Pending* and *Reopened* will generically be termed *Open defects* hereafter.

New	Defect never seen before.
In Progress	Defect subject to ongoing remedial work.
Pending	Defect pending remedial action while experts gather additional information.
Resolved	Defect that has been fixed.
Reopened	Defect that still occurs.
Closed	Definitely fixed defect.

Table 3. Classification of software defects under testing sessions.

The test manager separately surveyed New and Open defects during the first semester of 2011.

- The amount of Open defects was growing steadily from 50 (February 2011) to 130 (June 2011) (Figure 1). Let N_1 equal the number of defects on the 1^{st} of February, N_2 equal the number on the 1^{st} of June, and G the number of days in the interval. The amount of Open defects increased at this average rate:

$$\Delta = \frac{N_2 - N_1}{G} = \frac{130 - 50}{120} = \frac{80}{120} = \\ = \frac{6.7\ \text{Open defects}}{10\ \text{Days}}$$

- Meanwhile, New defects were decreasing.

Augmenting Open defects and diminishing New defects demonstrated that several software errors were causing cascade failures. The leaders meant to analyse this conclusion using the Release Note that is a software tool for monitoring the status of defects in the course of operations.

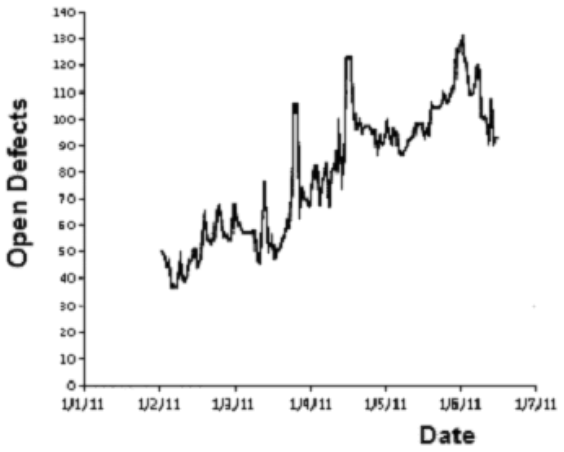

Figure 1. Number of open defects in the first semester of 2011.

The Release Note provided a diagram that exhibits the six states listed in Table 3, in addition to the special block 'Release Note', which indicates the status of defects just resolved and under registration by means of the Release Note (Figure 2). Teams (I), (II) and (III) responsible for handling precise states appear on the far left (Figure 2). For instance, the Development Team was in charge of the defects in Pending, Resolved and In Progress. The flow diagram also shows the transitions of defects from one state to another with transition frequencies. For instance, 2% of new defects evolve toward the Pending status.

The regular steps to handle New errors are the following:

$$\text{New} \rightarrow \text{Resolved} \rightarrow \text{Release Note} \rightarrow \text{Closed}$$

Empirical evidence shows that only 88% of New defects became Resolved; 89% of defects passed from Resolved to Release Note; and 84% officially registered in Release Note were Closed. In sum, scarcely more than half of New defects were closed throughout the regular procedure:

$$(0.88 \times 0.89 \times 0.84) = 0.65$$

The remaining defects followed a winding pathway. For example, as many as 16% of Resolved defects were tested anew for partial corrections (see Release Note → Reopened in Figure 2). Nearly 6% of Closed defects followed similar abnormal treatment (see Close → Reopened in Figure 2). A non-negligible amount of defects crossed the states In Progress → Pending; others followed the pathway: Pending → Resolved → In Progress. Finally, 11% of Resolved defects could not be registered by Release Notes due to various reasons and were lost:

$$(100\% - 89\%) = 11\%$$

The project leaders calculated the number of defects that remained in the same state from approximately February to May 2011, using a simulation program, and obtained the results in Table 4.

		Percentage
1	New	4%
2	In Progress	55%
3	Pending	92%
4	Resolved	11%
5	Reopened	1%

Table 4. Percentage of defects remaining in the same status during the period.

It is worth explaining how these values – in particular 2 and 3 – do not derive from the priority of defects. Software defects with different urgency levels shared the same destiny. For example, a software error with high priority was revamped and closed for a short while, but a subsequent regression test often placed it into the Open status anew. This cyclic mechanism occurred more than once. In substance, the Release Note and the simulation program provided evidence of the bad handling of defects, and, in turn, the abnormal software development.

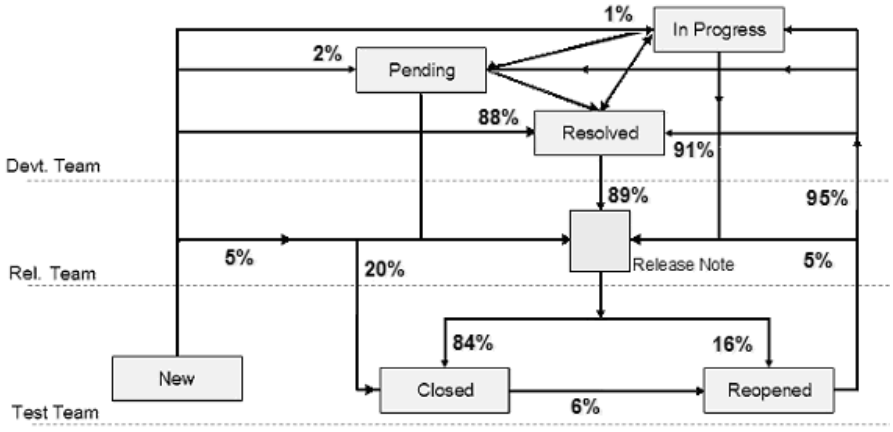

Figure 2. Flowchart of defects' states.

Practical observations of everyday job corroborated the statistical data reported above and clarified the dimension of the project failure. Managers noted how groups A and B were overloaded and spent most energy fixing the software defects rather than developing new code. Secondly, the high number of tests incurred time delays and high costs.

In conclusion, teams A and B were showing rather chaotic behaviour. This defeat could depend on the development of knowledge transfer since both teams included individuals characterised by high professionalism. Inefficient KTPs were influencing the operations in a dramatic manner.

3. Structural Analysis and Reorganization

A special control group of experts began to analyse the work organization of the project in search of the reasons for the abnormal KTPs.

They found out that, in the early beginning, commercial obligations counselled the management of IBM to set up a specific work organization whose entities corresponded to the entities of the IT Client for each of the first four levels (see horizontal dotted lines at the right side of Figure 3). In substance, the provider's structure mirrored the client's structure at the upper levels, and this arrangement helped the managers to cooperate in reaching common goals. The manager profiles were as follows:

- The *Executive Manager* was formally responsible for all the aspects of the work and for the relations with the customer.
- The *Project Manager* was responsible for technical questions.
- The *Technology Executive* was responsible for the overall architectural design of the ticketing application.
- The *Demand Manager* defined the detailed requirements and ensured the adherence of the ticketing application to the customer's needs.
- The *Test Manager* assured that software defect rate was constantly kept under the agreed thresholds.
- The *Project Management Office* (PMO) defined and maintained standard procedures for the project management.
- The *Quality Manager* was responsible for quality assurance.
- The *Release Manager* was responsible for the development, build and release of the ticketing application; in particular, he furnished a guide to the development teams located in Italy and in India.
- The *Infrastructure & Service Manager* ensured that the infrastructure services met the negotiated requirements in terms of size, performance and availability of the system.

The control group emphasized how the upper levels of the Provider organisation had been influenced by the signed contract, whereas the lower levels had been arranged beyond precise criteria of effectiveness. Teams A and B had contrasting features. In particular, the control group remarked:

1. The Indian team was somewhat large (50 to 70 programmers, as described above) and rigidly structured according to hierarchical levels. There were *managers, general coordinators, area coordinators, specialised developers* and *generic developers*. They adopted standard methodologies and used advanced software tools such as Rational, but they followed somewhat rigid work-procedures. Most of the team A members were young and lacking professional experience in large software projects. The Indian team was chiefly in charge of coding and had a low level of responsibility.
2. By contrast, the Italian team B included architects, analysts and developers with extensive experience and knowledge of the target market. The latter group took several details for granted, whereas the former group was completely unaware of technical requirements, needs of the customer, defects to correct, etc.
3. Testing was centralised in order to ensure full control of the software development. As a result, the Italian test team suffered from an overload of activity, which stressed the communication between domestic and offshore developers.

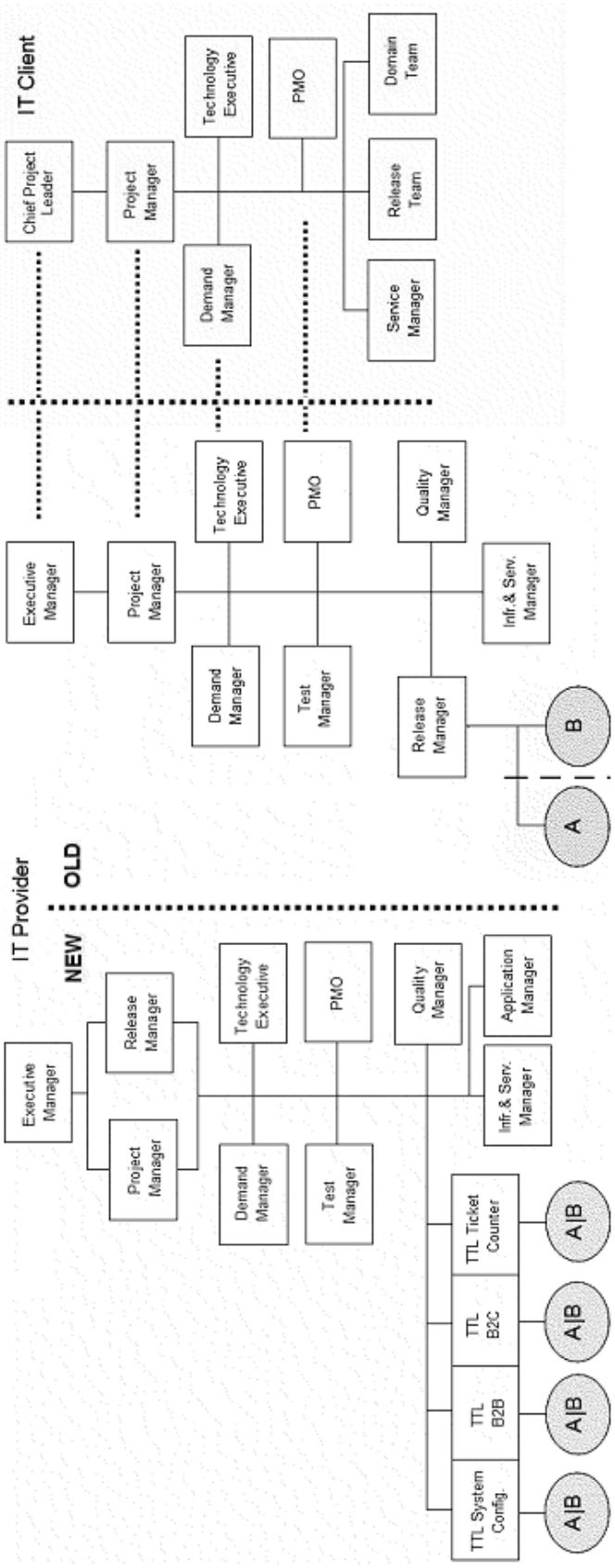

Figure 3. IT Provider new structure (far left), old structure (centre) and IT Client structure (far right).

4. For team B, it was not a straightforward task to explain the requirements of the IT Client and the Indian team. The latter had linguistic difficulties in reading some expressions typical of the Italian transport sector. There were considerable flaws in relation to the delivery of knowledge and knowledge acquisition by the Indians. The offshore team had very little domain knowledge and no understanding of how their development work fitted with the operations of the client.
5. The coding activity of team A was managed by chiefs of the parent organization who viewed the offshore support merely as a low-cost production facility with an abundant supply of cost-effective labour for low-level activities.

It was evident how the work organization of the IT Provider was arranged basically to answer contract obligations and was not suitable to manage knowledge transfer issues. Top managers decided to redesign the structure of the entities involved in the project. They moved the Release Manager near the Project Manager so that the two could gain better understanding of the software development in progress. The most significant changes that took place can be seen in the lower levels of the IT Provider structure and mean to enhance the collaboration between Italian and Indian developers.

Domestic and offshore developers were subdivided into small groups. That is to say, ex-members of teams A and B made eight groups that were paired off (Figure 3 far left) and reported to four Technical Team Chiefs (TTCs) responsible for the following areas of railway ticketing:

- *Ticket Counter* = Base functions for ticket selling.
- *Business to Business* (B2B) = Internet transactions amongst the IT Client and other transport companies.
- *Business to Consumer* (B2C) = Internet transactions amongst the IT Client and the travellers.
- *System Configuration* = Miscellany of technical functions.

Italian and Indian developers belonging to a precise area share the same issues and goals and cooperate better. In addition:

- Two Indian experts moved to Italy and acquired detailed information on the needs and characteristics of the Italian market through special training. They were appointed to facilitate integration between multicultural and multilingual environments. These professionals were wholly involved in optimising communication between the on-shore and off-shore resources.
- The Indian team was charged to carry out unit tests and functional tests before the corresponding Italian team.
- A dozen Indian developers learned the ticketing methods established by the IT Client and became able to suggest corrections for the software errors to offshore developers.
- The project leaders established that defects were to be treated following the correct path: New → Resolved → Release Note → Closed. Vicious circles and abnormal transitions were formally forbidden. As a result, a negligible number of defects went into the Pending status.
- All testing processes were monitored in the "*war room*", which included experts from both the on-shore and the off-shore sides. The war room members followed the status of each module, and analysed and weighted issues in real time.

4. Validation of the New Structure

Some project leaders identified 18 significant functions of the ticketing application before the reorganisation described above, and forecasted the resources necessary to validate those functions.

Each function should have required 406 tests, including unit and regression testing. The test

	Test per Function	Total Test Number	Total MDs	Elapsed Time (days)
Prospects	406	7,308	76	15
Definitive Data	96	1,728	18	4

Table 5. Summary of the validation process.

workload should have required 76 MDs (man-days) and should have caused 15 days of delay (Table 5, upper row).

The project leaders arranged the resources to check the 18 functions of the ticketing applications once the reorganisation was finished, and made the inventory in the close of testing. They observed that the grand total of tests dropped from 7,308 to 1,728; the MDs came down from 76 to 18, and the elapsed time fell from 15 days to four. It is worth mentioning that the regression errors no longer occurred. The human resources and the release times dropped down by up to one fourth of the resources previously supplied. A sound net 76% savings on costs and efforts was achieved.

These relevant differences in the number of tests and the amount of resources necessary to ensure the quality of production mean that the new work organization of the IT Provider created better software modules. The novel governance structure created to enable knowledge sharing across the organizational boundaries of the offshore environment was working and the KTPs issues were beyond any reasonable doubt.

5. Discussion and Conclusion

The IT Provider recruited skilled developers before the present project start. It took adequate measures to satisfy the requirements negotiated with the Client. When blatant failures emerged during the development of the ticketing application, the project leaders anticipated unexpected and hidden issues and superposed difficulties in knowledge transfer. For this reason, the project leaders conducted complex inquiries, which the present paper accurately recounts.

Empirical data convinced the managers that the cultural gap between Italian and Indian developers and the diverging daily methods of work were a unique origin of inefficient outcomes and economic losses. In substance, the lesson learned by the project leaders fits with current literature, which illustrates the negative effects provoked by KTPs issues. This kind of issues can go so far as to cause depreciable quality of software products, low performance, unacceptable time delays, out of control costs and other noteworthy difficulties.

A unified and integrated solution that ensures perfect KTPs does not exist in the literature; thus, a structural rearrangement of the project organisation can be seen as a contribution provided by the present paper to the scientific community. In particular, the small teams created to ensure close communication amongst local and remote practitioners are the key solution that we emphasize. People working in small groups can learn from each other about what is working better. A small number of people encourages interaction; it keeps discussion manageable and enables discussion to happen: it becomes easier to tackle problems. The individuals can get to know each other and become comfortable with each other. In substance, the introduction of small teams turns out to be the essential organisation measure devised to enhance KTPs in this case study.

An anticipatory report of this work has been delivered in [15].

References

[1] A. GOPAL, B. R. KOKA, The Role of Contracts on Quality and Returns to Quality in Offshore Software Development Outsourcing. *Decision Sciences*, **41**(3), 491–516, 2010.

[2] M. KEIL, P. CULE, K. LYYTIMEN, R. SCHMIDT, A Framework for Identifying Software Project Risk. *Comm. of the ACM*, **41**(11), 76–83, 1998.

[3] A. GOPAL, T. MUKHOPADHYAY, M. S. KRISHNAN, The Role of Software Processes and Communication in Offshore Software Development. *Comm. of the ACM*, **45**(4), 193–200, 2002.

[4] P. DRUCKER, The New Realities. *Transaction Publishers*, Revised edition, 2003.

[5] S. U. KHAN, M. NIAZI, R. AHMAD, Critical Success Factors for Offshore Software Development Outsourcing Vendors: A Systematic Literature Review. *Proc. of the 4th IEEE Conf. on Global Software Engineering*, 207–216, 2009.

[6] T. KENDRICK, *Identifying and Managing Project Risk: Essential Tools for Failure Proofing Your Project*. Amacom, New York, 2009.

[7] D. LEE, A. SMITH, M. MORTIMER, Cultural differences affecting quality and productivity in Western/Asian offshore software development. *Proc. of the 3rd International Conference on Human Computer Interaction*, 29–39, 2011.

[8] A. MATHRANI, S. MATHRANI, D. PARSONS, Knowledge Management Initiatives in Offshore Software Development: Vendors' Perspectives. *J. of Universal Computer Science*, **18**(19), 2706–2730, 2012.

[9] N. V. A. PRABHU, R. LATHA, K. SANKARAN, G. KANNABIRAN, Impact of Knowledge Management on Offshore Software Development: An Exploratory Study. *Proc. of the 3rd Intl. Conf. on Advanced Computing*, 121–128, 2011.

[10] K. MATSUYAMA, Quantitative Comparison Between Domestic and Offshoring Projects in the Software Factory Environment Driven by UML-Modelled Development Standard. *Proc. of the 2nd IEEE Intl. Conf. on Information Management and Engineering*, 590–596, 2010.

[11] F. SALGER, S. SAUER, G. ENGELS, A. BAUMANN, Knowledge Transfer in Global Software Development: Leveraging Ontologies, Tools and Assessments. *Proc. of the 32nd ACM/IEEE Intl. Conf. on Software Engineering*, **2**, 211–214, 2010.

[12] G. LEE, J. A. ESPINOSA, W. H. DELONE, Task Environment Complexity, Global Team Dispersion, Process Capabilities and Coordination in Software Development. *IEEE Trans. on Software Engineering*, **39**(12), 1753–1771, 2013.

[13] A. RAI, L. M. MARUPING, V. VENKATESH, Offshore Information Systems Project Success: the Role of Social Embeddedness and Cultural Characteristics. *MIS Quarterly*, **33**(3), 617–641, 2009.

[14] S. SARKER, S. SARKER, Exploring Agility in Distributed Information Systems Development Teams: An Interpretive Study in an Offshoring Context. *Information Systems Research*, **20**(3), 440–461, 2009.

[15] C. CONSOLI, P. ROCCHI, P. SPAGNOLETTI, P. NICO, Reorganizing an Offshore Software Project with the Goal of Favoring Knowledge Transfer. *Proc. of the 9th Int. Conf. on Software Engineering Advances*, Nice (France), in printing, 2014.

Received: January, 2014
Revised: September, 2014
Accepted: September, 2014

Contact addresses:
Carlo Consoli
IBM
Roma
Italy
e-mail: carlo.consoli@it.ibm.com

Paolo Rocchi
LUISS
Guido Carli University
Roma
Italy
e-mail: procchi@luiss.it

Paolo Spagnoletti
LUISS
Guido Carli University
Roma
Italy
e-mail: pspagnoletti@luiss.it

CARLO CONSOLI graduated in Computer Science from the University of Rome "La Sapienza" with a thesis focusing on artificial intelligence and natural language processing. He is currently working as an expert of information technology at IBM. Consoli has been involved in research projects framed in the field investigation of the Internet technologies, predictive analytics, and the semantic web.

PAOLO ROCCHI recevied the degree in Physics from the University of Rome. He retired from IBM as docent emeritus and is presently a contract professor at the LUISS University. He has driven investigations over various theoretical fields including general systems theory, reliability theory, linguistics, information theory, probability calculus. He has also investigated an assortment of applied topics in ICT. He has written over one hundred works including a dozen books. Newspapers and television have commented Rocchi's professional activity in successive stages and his biographical entry was included in "Who's Who in the World" (Marquis, 2002, 2003, 2004).

PAOLO SPAGNOLETTI is assistant professor of Information Systems and Organization at LUISS Guido Carli University where he coordinates the Research Centre on Information Systems (CeRSI). He received his Ph.D. from LUISS and has been a visiting fellow at LSE, Georgia State University and University of Agder. He serves as vice-president of the Italian chapter of AIS and as ERCIS member. His research interests are in digital platform, digital transformation, and IT governance, privacy and security. His works are published in Information & Management, Communications of AIS, Intl. J. of Accounting Information Systems, J. of Theoretical and Applied E-Commerce Research, J. of Information System Security and conference proceedings. He has edited three books and since 2012 he is Executive Editor of the Springer series LNISO.