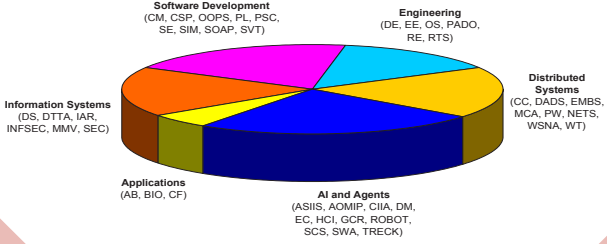
2010
Symposium on Applied Computing



Association for Computing Machinery

Advancing Computing as a Science & Profession

Main Menu

- Committee
- Sponsor
- Chair Messages
- Table of Contents
- Keyword Index
- Author Index
- Reviewer Index

- CD-ROM Help
- Search
- Copyright

Hosted by
University of Applied Sciences Western Switzerland (HES-SO)
and Ecole Polytechnique Fédérale de Lausanne (EPFL)

Sierre, Switzerland
March 22 - 26, 2010

APPLIED COMPUTING 2010

The 25th Annual ACM Symposium on Applied Computing

Sierre, Switzerland • March 22 - 26, 2010

If you have questions regarding the installation, please contact:

The Printing House, Inc.
Phone: +1-608-873-4500 Fax: +1-608-873-4558
Hours: Monday through Friday, 8 am - 5 pm CST
e-mail: graphics@printinghouseinc.com

Main Menu

Message from the Symposium Chairs

On behalf of the Organizing Committee, we welcome you to the 25th Annual ACM Symposium on Applied Computing (SAC 2010), hosted by University of Applied Sciences Western Switzerland. This international forum has been dedicated to computer scientists, engineers and practitioners for the purpose of presenting their findings and research results in various areas of computer applications. The organizing committee is grateful for your participation in this exiting international event. We hope that this conference proves interesting and beneficial.

The Symposium is sponsored by the ACM Special Interest Group on Applied Computing (SIGAPP), whose mission is to further the interests of computing professionals engaged in the design and development of new computing applications, interdisciplinary applications areas, and applied research. This conference is dedicated to the study of applied research of real-world problems. This event provides an avenue to discuss and exchange new ideas in the wide spectrum of application areas. We all recognize the importance of keeping up with the latest developments in our current areas of expedites.

SAC 2010 offers Technical Tracks and Poster Sessions. The success of the conference can be attributed to the substantial contribution of talented Track Chairs and Co-Chairs. Each track maintains a program committee and a set of highly qualified reviewers. We wish to thank the Track Chairs, Co-Chairs, Committee Members and participating reviewers for their hard work and effort to make the SAC 2010 conference a high quality conference. We also thank our invited keynote speakers, Dr. Willy Zwaenepoel, EPFL, Lausanne, Switzerland, and Dr. Bertrand Meyer, ETH Zurich, Switzerland, for sharing their knowledge with SAC attendees. Most of all, special thanks to the authors and presenters for sharing their experience with the rest of us and to all attendees for joining us in Crans-Montana/Sierre, Switzerland this year.

The local organizing committee has been a central contributor to the success of the SAC 2010 conference. Our gratitude goes to the Conference Vice-Chair Dr. Michael Schumacher of University and Local Chair Dr. Christelle Monnet of University of Applied Sciences Western Switzerland in Sierre. We also extend our thanks to the Publication Chair, Dr. Dongwan Shin, New Mexico Tech, Socorro, New Mexico, for his tremendous effort in putting together the conference proceedings, Posters Chair Dr. Jiman Hong, Soongsil University, Seoul, Korea, for his hard work to make a successful Poster Program, and Tutorials Chair Dr. Boi Faltings, EPFL, Lausanne, Switzerland, for arranging an exciting set of Tutorials. A special thanks goes to our Program Chairs Dr. Mathew J. Palakal, Indiana University Purdue University, Indianapolis, Indiana, and Dr. Chih-Cheng Hung, Southern Polytechnic State University, Marietta, Georgia, for coordinating and bringing together an excellent Technical Program.

Again, we welcome you to SAC 2010 and the beautiful city of Crans-Montana/Sierre, Switzerland. We hope you enjoy the SAC 2010 conference and your stay in Switzerland. Next year, we invite you to participate in SAC 2011 to be held in Tai Chung, Taiwan. The conference will be hosted by the Tunghai University.

Sung Y. Shin and Sascha Ossowski
SAC 2010 Conference Chairs

Message from the Program Chairs

Mathew J. Palakal
Indiana University Purdue University, Indianapolis, USA

Chih-Cheng Hung
Southern Polytechnic State University, Marietta, USA

Welcome to the 25th International Symposium on Applied Computing (SAC 2010). For the past 24 years, SAC has become a major international venue for computing researchers and applied practitioners to convene and share ideas on recent developments in a variety of applied areas of Information Technology. The success of SAC has been the consolidation of a wide range of applied areas into specialized modules called Tracks. Each of the Tracks are then organized and administered by experts in the respective areas by instituting program committees, carrying out blind reviews according to the ACM guidelines, and finally selecting the highly qualified papers for the Track. Since its inception six years ago, the Poster Sessions at SAC have become a tradition, and this year again the Poster will be an integral part of the Technical Program at SAC 2010.

The open Call for Track Proposals generated 47 proposals, and after prescreening the proposals, 43 Tracks were finally accepted for SAC 2010. The prescreening and selections were made based on the success of those Tracks in the previous SACs as well as targeting new and emerging areas. The Call for Papers for these Tracks attracted 1,414 abstract submissions and 1,353 final paper submissions from 70 different countries. The final submitted number of papers indicates a 96% submission rate. The submitted papers underwent the blind review process and 364 papers were finally accepted as full papers for inclusion in the Conference Proceedings and presentation during the Symposium. The final acceptance rate for SAC 2010 is 26.9% for the overall track. In addition to the accepted full papers, 88 papers that received high enough review scores were accepted as short papers for the Poster Program.

The Technical Program Organization of SAC 2010 is made possible through the hard work of many people from the scientific community who have volunteered and committed many hours to make it a success. Much credit goes to all the Track Chairs for making SAC 2010 Technical Sessions a huge success. Some of the popular Tracks had an unprecedented submissions and having three blind reviews for each paper was certainly a major challenge. Once again this year, we follow the previous years' tradition of organizing various tracks into six different themes. The Symposium Proceedings and the technical presentations are focused around these themes to form a series of related track sessions.

On behalf of the entire SAC 2010 Organizing Committee, we congratulate all the authors for having their papers accepted in their respective Tracks, and we wish to thank all of those who made this year's technical program a huge success. Specifically we wish to thank the speakers, track chairs, reviewers, program committee members, session chairs, presenters, and all the attendees. We also wish to convey our special thanks to the local organizing committee lead by Christelle Monnet from the University of Applied Sciences, Western Switzerland.

We wish you all a pleasant stay in Sierre, hope you have a great time at SAC 2010, and you will have the opportunity to share and exchange your ideas and foster new collaborations. We would also like to take this opportunity to convey to you the news that the 26th International Symposium on Applied Computing (SAC 2011) will be held in Taipei, Taiwan. We hope to see you all and your colleagues at SAC 2011.

On the Use of Patterns to Recover Business Processes

Ricardo Pérez-Castillo, Ignacio García Rodríguez de Guzmán and Mario Piattini
Alarcos Research Group, University of Castilla-La Mancha
Pº de la Universidad, 4 13071, Ciudad Real, Spain, +34 926 295 300
{ricardo.pdelcastillo, ignacio.grodriguez, mario.piattini}@uclm.es

ABSTRACT
Legacy systems keep key business knowledge from companies over time. This knowledge is hidden in the source code lines and must be recovered through software archeology processes to maintain and help the legacy systems to evolve, so that the ROI and lifespan of those systems can be improved. This paper proposes a set of patterns to obtain, in a deterministic manner, business models from the source code of legacy systems. Thus, the business process models recovered from the legacy systems preserve the business knowledge and can be used to modernize and maintain the legacy systems.

Categories and Subject Descriptors
D.2.11 [**Software Engineering**]: Software Architectures – *Patterns.* I.5.1 [**Pattern Recognition**]: Models – *structural.*

General Terms
Design, Experimentation, and Theory

Keywords
Business Process, Patterns, Software Archeology, Modernization, ADM, Legacy Systems.

1. INTRODUCTION
There is a vast number of highly functional and operational *Legacy Information Systems* (LIS) in companies. These LIS are not immune to software ageing and must be maintained. The full replacement of these systems from scratch has a great impact in technological and economical terms. Moreover, LIS represent a huge business value for companies [5]. Therefore, the most appropriate kind of maintenance is evolutionary maintenance based on re-engineering processes.

Indeed, the typical re-engineering process has shifted to so-called *Architecture-Driven Modernization* (ADM) over the last years [8]. ADM advocates carrying out re-engineering processes following the principles of model-driven development. A key activity in the ADM process is the conceptual representation of the LIS at a higher abstraction level throughout the reverse engineering stage. According to [6], there are three different views with respect to the achieved abstraction level: (i) *technical view*, which is related to migration to another language; (ii) *application/data architecture view*, which focuses on obtaining a model of the system design; and (iii) *business view*, which increases the degree of abstraction, since a business architecture model is obtained.

Obtaining an abstract model from the LIS source code is easy with the first and second choices, since there are design patterns that have been successfully adopted by the software industry [4] that can be applied in the reverse engineering stage. However, the transformation of the third choice is not clear, since there is a huge abstraction gap between the source code and the business model. For this reason, this paper proposes a set of patterns that recognizes certain structures in a source code model and infers specific elements in the target model of the business process. This paper also presents a case study where the proposed patterns are applied to a real-life system.

The remainder of this paper is organized as follows: Section 2 summarizes the related work; Section 3 shows the proposed business patterns in detail; and finally, Section 4 addresses the conclusions and future work.

SAC'10, March 22-26, 2010, Sierre, Switzerland.

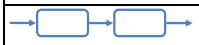
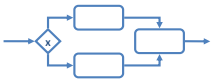
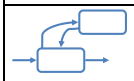
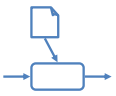
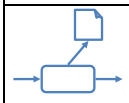
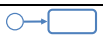
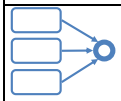
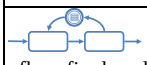
2. BACKGROUND
Information Systems support most of the business logic defined in the *Business Processes* (BP). Thus, organizations demand the ability to view their BPs in order to modify and improve them, so that they can deal with the changes in their company's environment and maintain their competitiveness. However, LIS can be an obstacle to the BP management, since these systems have been able to evolve independently from the organization's BPs. This fact implies that LIS embed a lot of business knowledge over time [7]. Therefore, there are BPs hidden in the LIS that are not defined in the original set of BPs. Thus, the recovery of hidden BPs from the LIS by means of the ADM process is necessary to obtain the current BPs.

Patterns have been frequently used in the software community. *Gamma et al.* [4] propose a set of design patterns focusing on implementation model details; and *Fowler* [3] presents a set of analysis patterns. Also, patterns have been used in ADM processes. *Pérez-Castillo et al.* [9] propose a set of patterns to recover web services from relational databases according to an ADM process.

Moreover, business patterns have been widely used to model business architectures: *Zhao et al.* [10] present a pattern language for designing e-business architectures; *Dodani* [2] proposes pattern-driven solution engineering for e-commerce applications; *Aalst et al.* [1] describe a number of workflow patterns to identify comprehensive workflow functionality. In spite of the fact that patterns have been used to reuse representations of business knowledge, patterns have never been used to recover BP models from LIS models.

3. BUSINESS PROCESS PATTERNS
The proposed set of patterns consists of ten business process patterns grouped into three categories: (i) structural patterns deal with the built elements and their combinations such as process, task and sequence flows; (ii) data patterns deal with data objects and how these objects are related to other elements; and (iii) event patterns build all the elements involved in the event management. Table 1 summarizes the set of patterns.

Table 1. Business Process Patterns

	Business Process Pattern
STURCTURAL	 **P1. BPD Skeleton.** This pattern creates the root structure of the BP model. It creates a BP diagram for each KDM code model. Also, it builds a pool element with a nested process in the BP diagram for each package of the KDM code model.
	 **P2. Sequence.** This pattern takes any callable piece of code from the KDM code model (depending on the programming language: method, function, procedure, and so on) and maps them into tasks in the BP diagram. In addition, the sequence of calls to callable units is transformed into a set of sequence flows in the same order between the tasks built from the callable unit respectively.
	 **P3. Branching.** This pattern transforms each conditional jump of the source code that has two mutually exclusive choices into an exclusive gateway and two different sequence flows in the BP model. Typically those exclusive conditional branches are related with the if ... then ... else or switch clauses in several programming languages. The exclusive gateway represents the condition that is evaluated and the two sequence flows represent two conditional transitions that depend on the value (true or false) of the evaluation.
	 **P4. Collaboration.** Each call to external callable unit (i.e. API libraries or external components outside the LIS) is transformed into an auxiliary task as well as two sequence flows: the first from the source task to the auxiliary task and the second returning to the source task.
DATA	 **P5. Data Input.** This pattern builds a data object in the BP model for each input data within a callable unit in the KDM code model. Also, this pattern builds an association between the data objects and the task previously built from the callable unit. This pattern only considers as input data the parameters or arguments of the callable unit, but it does not consider the auxiliary variables within the callable unit.
	 **P6. Data Output.** Each piece of output data involved in a callable unit is transformed by means of this pattern into a data object as well as an association from the task (built from the callable unit) to the data object. This pattern excludes as output data the auxiliary and intermediate data in the body of the callable unit. The output data is the data returned by the callable unit or external data related to databases or files.
EVENT	 **P7. Start.** The task building from the callable unit that starts the execution of any program or application of the LIS is considered the initial task. Therefore, a start event is built into the BP diagram and a sequence flow from this event to the initial task is also created.
	 **P8. Implicit Termination.** This pattern builds an end event in the BP model. Then, it creates sequence flows from 'end task' and those flows merge in the end event. A task is considered as an 'end task' if this task does not have any outgoing sequence flow.
	 **P9. Conditional Sequence.** This pattern transforms each conditional call into a sequence flow fired under a conditional intermediate event through to the task related to the callable unit. This pattern makes it possible to create arbitrary cycles in the BP diagram.
	 **P10. Exception.** Each call to callable unit under any exception is transformed into a task for the piece of source code that handles the exception as well as a sequence flow fired under an error intermediate event. Indeed, this pattern can be understood as a specialization of the pattern P9.

4. CONCLUSIONS

This paper presents a set of patterns to detect business processes in source code. With the proposed patterns, BPMN models can be obtained from the source code models obtained from legacy systems. Thus, the recovered business knowledge of the legacy systems is preserved and can be used to modernize and maintain the legacy systems, and preserve the alignment between the business logic and the information system.

The set of patterns has been validated by means of a case study where the patterns were applied to a legacy system. This case study reported that the proposed set of patterns makes it possible to obtain BPMN models in a cohesive and non-coupling manner.

Due to the fact that large LIS result in large BP models, the future extensions of this research will focus on introducing clustering techniques in order to reduce the size of the BP models. In addition, case studies with enterprise legacy systems will be carried out where system and business experts will evaluate whether the obtained BPs faithfully represent the company's operations. These case studies may also help to detect new business structure needs that will allow the definition of more refined patterns.

5. ACKNOWLEDGMENTS

This work was supported by the *FPU Spanish Program*; and the *JCCM*'s projects: ALTAMIRA (PII2I09-0106-2463), INGENIO (PAC08-0154-9262) and PRALIN (PAC08-0121-1374).

6. REFERENCES

[1] Aalst, W.M.P.v.d., A.H.M.t. Hofstede, B. Kiepuszewski, and A.P. Barros., "Workflow Patterns". Distributed and Parallel Databases, 2003. 14(3): p. 5-51.

[2] Dodani, M.H., "Pattern Driven Solution Engineering". Journal of Object Technology, 2003. 2(2): p. 27-33.

[3] Fowler, M., Analysis Patterns: Reusable Object Models. 1997: Addison-Wesley.

[4] Gamma, E., R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Longman Publishing Co. ed. 1995, Addison Wesley.

[5] Ghazarian, A., A Case Study of Source Code Evolution, in 13th European Conference on Software Maintenance and Reengineering (CSMR'09), R. Ferenc, J. Knodel, and A. Winter, Editors. 2009, IEEE C.S. p. 159-168.

[6] Khusidman, V. and W. Ulrich, Architecture-Driven Modernization: Transforming the Enterprise. DRAFT V.5. http://www.omg.org/docs/admtf/07-12-01.pdf. 2007, OMG.

[7] Mens, T., "Introduction and Roadmap: History and Challenges of Software Evolution ". Software Evolution (Springer Berlin Heidelberg), 2008. 1: p. 1-11.

[8] OMG. ADM Task Force by OMG. 2007 9/06/2009 [cited 2008 15/06/2009]; Available from: http://www.omg.org/.

[9] Pérez-Castillo, R., I. García-Rodríguez de Guzmán, I. Caballero, M. Polo, and M. Piattini, PRECISO: A Reengineering Process and a Tool for Database Modernisation through Web Services in 24th Annual ACM Symposium on Applied Computing (SAC'09). 2009: Hawaii, USA. p. 2126-2133.

[10] Zhao, L., L. Macaulay, J. Adams, and P. Verschueren, "A pattern language for designing e-business architecture". J. Syst. Softw., 2008. 81(8): p. 1272-1287.