Software Process Improvement for Component-Based Software Engineering: An Introduction to the OOSPICE Project

Fritz Stallinger	Alec Dorling	Terry Rout	Brian Henderson-Sellers	Bruno Lefever
Kepler University Linz Altenberger Straße 69 A-4040 Linz, Austria fs@sea.uni-linz.ac.at	University of Boras Allegatan 1 S-501 90 Boras, Sweden alec.dorling@hb.se	Software Quality Institute, Griffith University Nathan, Qld 4111, Australia t.rout@sqi.gu.edu.au	University of Technology, Sydney P.O. Box 123, Broadway, NSW 2007, Australia brian@it.uts.edu.au	Computer Associates Bd de la Woluwedal 34/13 B-1200 Brussels, Belgium bruno.lefever@ca.com

Abstract

Software Process Improvement is generally regarded a key to economic success by increasing the quality of software systems, accelerating time-to-market and decreasing development costs. Component-based software engineering, as an emerging development paradigm, targets very similar goals by focusing on the assembly of software systems from components and emphasising software reuse. This paper firstly provides an overview on the breadth and complexity of component-based development and then considers software process assessment and improvement in the context of component-based software engineering, identifying the major deficiencies of both fields with respect to the other. Based on these insights, an introduction to the EU-funded project OOSPICE (IST-1999-29073) focussing on overcoming these deficiencies is provided by detailing the project's domain, its rationale, objectives and outcomes.

Keywords: software process assessment, software process improvement, object-oriented development, component-based software engineering.

1 Introduction and Background

In the last few years, the software engineering community has witnessed the growing popularity of Component-Based Development (CBD), refocusing software development from core in-house development to the use of internally or externally supplied components. While this approach generally promises increased efficiency and flexibility in software development, it severely impacts traditional software engineering processes, e.g. by establishing a fundamental separation and independence between specification and implementation, the focus on architectural tasks, the split of the development activities into multiple parallel *component provisioning* and *application assembly* tracks, and the complex dependencies between project management and component management in general.

Component-Based Software Engineering (CBSE) as an emerging discipline (e.g. [10]) is targeted at improving the understanding of components and of systems built from components and at improving the CBD process itself. Nevertheless, the increasing popularity of CBD has widely been driven by the provision of standards on the product side (e.g. component models such as COM/DCOM, JavaBeans, CORBA), while widely accepted and systematic approaches to the standardization of development processes for CBSE are generally missing.

On the other hand, the field of Software Process Improvement (SPI), and in particular of assessment-based software process improvement, shares very similar goals to CBSE – shorter time-to-market, reduced costs and increased quality – and provides a wide spectrum of approaches to the evaluation and improvement of software processes (CMM [25], SPICE [19], etc.). Although this discipline has made considerable advances in the standardization of these approaches (e.g. ISO 15504 [20]) as well as of the underlying process models, it generally lacks tailoring and customisation to CBSE, e.g. with respect to terminology or the adequacy and granularity of the underlying process or assessment models.

The EU-funded international research and development project OOSPICE (IST-1999-29073) aims at overcoming the above shortcomings experienced when applying current SPI approaches to CBD. Based on the principles of empirical software engineering, it focuses on the processes, technology and quality in software development using CBD. Its main objectives are a unified CBD process model and underpinning metamodel, a CBD assessment methodology consisting of an assessment model, an assessment method and an assessment software tool, and a CBD methodology together with a CBD software tool.

The remainder of the paper first summarises the breadth, and hence complexity, of CBD in practice and theory, also addressing the relationship between object-oriented development and CBD (Section 2), then consid-

Proceedings of the 28 th Euromicro Conference (EUROMICRO'02)
1089-6503/02 $17.00 © 2002 **IEEE**

ers process improvement and assessment in a CBD context (Section 3), and – based on this analysis – provides an introduction to the OOSPICE project by describing the project's domain, its rationale, objectives and specific outcomes (Section 4). A summary and conclusions round off the paper.

2 The Practice of Component-based Software Engineering

The much-sought goal of widespread software reuse promised to be realised during the late 1990s. Object-oriented development matured technically and managerially to the extent that high-quality systems and subsystems could be produced using a limited number of (almost standardised) techniques (cf. [4]). This demonstrated the power of considering the work of a software system as being achieved by a set of separate units of behaviour, which co-operate via well-defined interfaces. Since separate units implemented as objects proved useful, the more general notion of separate units – components – could and does work [2][26]. Moreover, distributed objects (e.g. [24]) proved how systems of objects in separate memory spaces were highly effective, especially when packaged as components [15]. The demands of rapidly changing business environments [8], especially those involving the Internet, have led to solution providers devising or adopting a plethora of ways to design and build software systems from third-party and their own components.

In 1997, Gartner estimated that CBD was worth $US 1.4 billion with a growth rate of 30% per annum [1]; some recent estimates have pegged growth at 80%. According to many analysts, an essential key to overcoming the software productivity bottleneck is increasing the reuse of components, i.e. to assemble systems from more-or-less finished building blocks. However, this simple idea is hard to achieve [1]. The advent of CBD approaches has radically changed the way in which systems are analysed, architected, implemented, transitioned and evolved. There is a huge variation in how CBD ideas are realised and research in the area has mushroomed, as exemplified by workshops at international software engineering conferences such as ICSE and TOOLS.

The term *component* itself has a wide variety of meanings in the current state-of-the-art [29]. Many authors require them to be binary units (e.g. [36]). A more pragmatic view is that a faceted characterisation of components should allow for source-code components [23]. Some authors only refer to them as packaged objects (e.g. [30]). However, participants of the ICSE '98 workshop concluded: "Object technology is neither necessary nor sufficient for CBSE" [3]. Nevertheless, many of the definitions capture the idea of non-trivial, replaceable elements of a system. Individual components are selected because they to some extent fulfil a required function of the system. Ideally components are assembled in a well-defined architecture [22]. Often they must exhibit object characteristics and are designed and implemented as objects.

Companies gain competitive advantage by acquiring third-party components off the shelf. They often have to develop their own components when third-party components do not fully satisfy system requirements [32] – this complicates CBD processes and their assessment. Systems or applications produced using CBD may themselves be offered as third-party components. Often such reuse is inadvertent and the size and complexity of components can range widely. In business systems, components typically represent the implementation of a business concept or business process that can be reused and deployed as a reusable element. However, time-to-market pressure may dictate the use of components for system infrastructure or even inter-component glue [31].

Hence, CBSE raises questions about whether existing software engineering approaches are appropriate in this emerging field. CBSE is *integration-centric*, emphasising, as it does, the selection, acquisition and integration of components from in-house sources and/or external vendors. The latter may include the use of commercial off-the-shelf (COTS) products or open source components. In other words, components may be either 'black box' or 'white box' leading to significant variation in CBD processes and their assessment.

At the implementation level three *component infrastructure technologies* have become *de facto* standards: the OMG's CORBA (Common Object Request Broker Architecture), Sun's JavaBeans and Enterprise JavaBeans, and Microsoft's (Distributed) Component Object Model (DCOM/COM). Notwithstanding, successful CBD companies employ an even wider variety of component and object technologies – and new ones continue to emerge. Although low-level CBD implementation technologies (as above) are widely accepted in the industry, methods and processes supporting CBSE are still not commonplace. Like other development approaches, CBSE would benefit greatly from the availability of *repeatable* processes for building, selecting and assembling components. Furthermore, current approaches to process improvement (e.g. ISO 15504 [19][20]) do not address those issues.

Finally, CBD relies heavily on the availability of high-quality components that fulfil a role in the intended system. A necessary precondition for widespread use of components is that they are of *trusted quality*. Current approaches towards component specifications typically focus on issues relevant for architecture and design (the *interface*) and do not adequately address quality related issues. Predicting the effects of integrating components from various sources is an essential requirement in a CBSE process. One also needs to determine the expected quality of the individual components in order to find out if

the integrated system meets the level-of-service requirements.

3 Process Assessment and Improvement

The assessment of process capability is a relatively new technique in the software industry, with formal approaches dating back to the pioneering work of Humphrey [16]. The emergence in the early 1990s of a number of competing approaches to process assessment led to the establishment of the SPICE Project and the development of an international standard, ISO 15504 [27][28]. To ensure its work would be appropriate and accepted, the SPICE Project has conducted extensive trials in industry [34], thereby defining an approach to the validation of standards that has had significant influence.

Studies by the Software Engineering Institute [13], supported by data from the SPICE Trials [35][21], have demonstrated the efficacy of process improvement approaches based upon the results of process assessment. Experiences with the use of ISO 15504 have demonstrated the potential flexibility of process assessment as a valuable technique for management in software and systems development [33]; although the value to organizations low on process capability is questionable [7]. Current work is leading to the evolution of ISO 15504 to a generic framework for process assessment [18], applicable across a wide range of domains, and provides an opening for the international acceptance and transition of the results of the SPICE Project. Under the revised architecture (currently under development), specific process models may be established for different domains, with the status of the models ensured through appropriate standardisation mechanisms.

Findings from assessments, however, indicate that some approaches to assessment have difficulties in evaluating process capability for CBD. These studies have tended to confirm expert views that the practices in CBD in many of its forms are not always readily mapped to the process models used in most assessment approaches. Tailoring of process assessment approaches for CBSE has been almost non-existent though available evidence indicates a need.

4 The OOSPICE Project

The OOSPICE Project[1] (Software Process Improvement and Capability dEtermination for Object Oriented / Component Based Software Development), which commenced in December 2000, is focussed on the industry practice of CBD and will deliver new processes, methods and tools that can be applied practically. Based on the principles of empirical software engineering it combines four major concepts: CBD, object-oriented development, software process assessment and software process improvement. Its main objectives are a unified CBD process model and underpinning metamodel, a CBD assessment methodology consisting of an assessment model, an assessment method and an assessment software tool, and a CBD methodology together with a CBD software tool.

4.1 Domain and Rationale

The domain of the OOSPICE project is CBSE in which complex software systems are often rapidly produced, for example, to meet the time-to-market of e-commerce. In general the technology involves designing an appropriate architecture for independently produced components (*architecting*), component *assembly* and component *provisioning* (often by third parties). CBD, encompassing these three aspects, is of increasing importance, especially for rapid software development.

However, in constituting the core rationale for the project, companies are noted to be experiencing many problems in CBD, *inter alia*, the following:

1. Companies believe that CBD is valuable and a key to enhancing both quality and productivity but do not have the knowledge or skills to obtain the benefits.
2. Organisational structures and processes are not adapted to CBD.
3. Inappropriate technical and management approaches are being used for CBD, for example, in risk estimation, project planning and quality assurance.
4. Information concerning availability, quality and reliability of software components is deficient.
5. Prevailing culture is resistant, rejecting newness and mistrusting components.
6. Information about capability (e.g. [14]) of component suppliers and their component construction process is inadequate.
7. Components are not trusted because they are not well specified or are not self-describing or reflective, especially where components are themselves complex assemblies of components.

Furthermore, various suitable processes for CBD are currently still emerging and their general applicability is being debated. Developers of component-based systems find it hard to determine and assess the appropriateness of available components.

Finally, especially in the context of reuse, the notion of software components has been around since McIlroy proposed it in 1968. With the emphasis on reuse in object-oriented development, together with a multiplicity of characterisations of what components are (e.g. from Microsoft, Sun, OMG) and the use of object-style interfaces

[1] This project is called OOSPICE for reasons to do with the relevance of object technology to the production of components. The focus of the project is CBD with the assumption that process models suited to object-oriented development will have to be addressed. The name of the project harks back to its origins when the plan was to modify SPICE to deal with object-oriented development. While developing the project, it was concluded that component-based software development needed to be addressed more than object-oriented development.

to components, practitioners and researchers remain unclear about the distinction between components and objects (e.g. [11]).

4.2 Focus of Work

An accepted principle in software engineering is that process quality determines product quality [17][5]. This is the basis of process assessment and improvement approaches like CMM [25] and SPICE [19]. The updating and improvement of this technology for CBD as performed as a focus of work for OOSPICE requires detailed analysis and codification of the wide variety of currently offered approaches, like OPEN [9] or Catalysis [6]). Based on identified best-practice approaches, it is the aim of OOSPICE to devise suitable process models and an assessment methodology.

There are no widely accepted processes specifically for CBD let alone their assessment. Furthermore, the relationship between CBD and the way in which components themselves are produced is also not clear, for example, the impact of object technology used for components like JavaBeans. Preliminary work by the OOSPICE partners has shown that established SPI approaches such as CMM [25] and SPICE [19] are not ideally suited to either object-oriented development or CBD. This constitutes a significant gap in technology. Hence, the project focuses on the processes employed for *both* the provision (development) of reusable software components and for the architecting and assembly of larger systems from acquired components. Thus, issues of reuse and the resulting modifications to existing processes have to be considered.

It is intended that the process model for CBD to be created in this project will be able to serve two functions: as a basis for a generic methodology for CBD and also as a reference model and source of process definitions for assessment of capability. Anchoring such an integrated approach also at the highest level in the model structure by identifying and documenting the underpinning metamodel will ensure the consistency of the approach throughout the project deliverables; it will also provide a significant innovation in the development of resources for process assessment, and an advance in the theoretical understanding of this technique. The metalevel description is the basis for CASE tool support and is commensurate with the current metalevel activities in the Software Process Engineering standardisation activities of the Object Management Group.

In exploring the relevant processes, the principles and techniques of process assessment, particularly as set out in ISO 15504, are applied.

In relation to producing *components*, the project seeks to answer the following questions:

- Are there specific and unique aspects of producing components that require or support the development of new or modified process reference models?
- Are the results of the *assessment* of processes for producing components of use and value in developing confidence in the suitability and value of these software components for use in CBD of systems?

In relation to CBD of systems, the project seeks to answer the following questions:

- Are there specific and unique aspects of processes for CBD that require or support the development of new or modified process reference models specifically for CBD?
- Are there specific issues involved in the application of assessment of capability of CBD processes for process improvement or capability determination?

Successful resolution of these questions involves the generation of process assessment models for both component production and component-based systems development. It also involves the development or modification of a process metamodel.

4.3 Objectives and Outcomes of the OOSPICE Project

OOSPICE is a project based on the principles of empirical software engineering focusing on the processes, technology and quality of software systems produced using CBD which is achieving substantial software reuse. Among its main objectives are a unified CBD process model and underpinning metamodel, a CBD assessment methodology, component-provider capability profiles, a CBD methodology and an extension to the ISO 15504 process assessment standard.

In detail the following major technological objectives have been identified for the project:

1. *Evaluation*: To evaluate current theory and practice in *CBD, using both published and unpublished sources,* including the study of selected industry projects.
2. *Unified CBD process model*: To propose a unifying theoretical CBD process model with reference to best practice, underpinned with a process metamodel.
3. *CBD assessment methodology*: To develop an industry-validated CBD process assessment methodology (model, method, tool).
4. *Component-provider capability profiles*: Define capability profiles for component providers through the analysis of results of the CBD process assessment methodology.
5. *CBD methodology*: Specify a CBD methodology (management and technical processes, methods and tools) for 'architecting' and assembly of independently produced components (e.g. COTS) and for component provision – including complex components produced using CBD.
6. *Dissemination and Use Planning*: Ensure international acceptance and take-up of results through dissemination, standards and licence schemes, augmented by

provision of web-based information and appropriate training material.

In terms of advances in knowledge and understanding the project is intended to achieve specific outcomes, defined and reflected in the project's deliverables. These outcomes are closely linked to the identified project objectives and include:

1. A more detailed understanding of processes relevant and specific to CBD; this will be encapsulated in the process model.
2. An understanding of issues to be addressed in process assessment in a component-based development environment; these will be addressed specifically in the assessment methodology.
3. Development of a specific assessment model and method for process assessment in CBD; the assessment model and the corresponding assessment tool will be specific deliverables.
4. Evaluation of the applicability of results of process assessment in CBD; this will occur in the evaluation of the industrial trials of the assessment and development methodologies.
5. Understanding of the process requirements for organisations seeking to move towards more effective use of CBD; this is encapsulated in the development methodology for CBD.
6. Understanding of the relationship between process capability ratings and the suitability of components developed using such processes; this will be documented in the justification for the recommended target profiles, and in the mechanism for constructing capability ratings.
7. Development of target capability profiles for determination of relevant capabilities for developers of components in an open market; these are a specified deliverable.

In achieving these outcomes the task of major initial importance is the definition of the process model. It should be noted that this process model [12] extends the current state-of-the-art in that it provides a model suitable for both assessment purposes and as the basis for a defined CBD methodology.

5 Summary and Conclusions

Component-based development is gaining increased popularity. It severely changes traditional software engineering processes, while generally accepted and systematic approaches to the standardization of development processes for CBD are still missing and there is a huge variation observable in how CBD ideas are currently realised. Software process improvement, on the other hand, shares similar goals of CBD but generally lacks tailoring and customisation to CBSE. The EU-funded international research and development project OOSPICE (IST-1999-29073) aims to overcome these gaps and the shortcomings experienced when applying current SPI approaches to CBD. It provides a bridge between process and process engineering, on the one hand, and process and process capability assessment on the other hand, bringing together these two sub-disciplines of software engineering. In line with the project objectives, technological innovation in the OOSPICE Project is in the following areas: the production of a unified process model and underpinning metamodel for CBD, addressing the variation mentioned above; the development of a CBD methodology that conforms to the unified process model for CBD; the definition and validation of an assessment methodology for CBD; the integration of user trials of both assessment and development methodologies; the definition of target capability profiles to assist identification of process-oriented risk in selection of component providers; the extension of relevant standards, particularly ISO 15504, to encompass object-oriented and component-based software development; and the definition of a mechanism for establishing a constructed capability from the capability profiles of different component providers where a system is produced using CBD and is itself offered as a component.

6 Acknowledgements

The OOSPICE Project is developed as EU-funded shared-cost RTD project under contract number IST-1999-29073 within the European Commission's Fifth Framework Programme.

The OOSPICE project partners are:

- Institute of Systems Sciences, Department of Systems Engineering and Automation, Kepler University Linz (Austria) (Coordinator)
- Computer Associates (Belgium)
- WAVE Solutions Information Technology GmbH (Austria)
- Huber Computer Datenverarbeitung GmbH (Austria)
- University of Boras (Sweden)
- Volvo Information Technology (Sweden)
- Griffith University (Australia)
- Centre for Object Technology and Application Research (Australia)

We wish to thank the Australian Research Council for funding the Australian contribution to this project.

We also wish to thank Middlesex University, School of Computing Science for contributing to the OOSPICE project proposal forming the basis of parts of this paper.

7 References

[1] Blechar, M., Gartner Group Report: Chasing the Elusive Productivity of Components, November 1999.

[2] Brown, A.W., Component-Based Software Engineering, IEEE Computer Society, 1996, ISBN 0-8186-7718-X.

[3] Brown, A.W. and Wallnau, K.C., The Current State of CBSE, IEEE Software, vol. 15, Oct. 1998.
[4] Bruegge, B. and Dutoit, A. H., Object-oriented Software Engineering: Conquering Complex and Changing Systems, Prentice-Hall, Upper Saddle River, 2000.
[5] Cignoni, G.A., Software Process Technologies and the Competitiveness Challenge, Conradi, R. (ed.): Software Process Technology, 7th European Workshop EWSPT 2000, Springer 2000, pp. 151-155.
[6] D'Souza, D.F. and Wills, A.C., Objects, Components, and Frameworks with UML. The Catalysis Approach, Addison-Wesley, 1999.
[7] Fayad, M.E. and Laitinen, M., Process assessment considered wasteful, Comm. ACM, 40 (1997), no.11, pp. 125-128.
[8] Feldman, S., The Objects of E-Commerce, OOPSLA '99 keynote address, Denver, November 1999.
[9] Graham, I., Henderson-Sellers, B. and Younessi, H., The OPEN Process Specification, Addison-Wesley, 1997.
[10] Heineman, G.T. and Councill, W.T., Component-Based Software Engineering. Putting the Pieces Together, Addison-Wesley, Boston, MA, USA, 2001.
[11] Henderson-Sellers, B., Pradhan, R., Szyperski, C., Taivalsaari, A. and Wills, A.C., Are components objects? A panel discussion, OOPSLA'99 Companion, ACM Press, 1999, pp.7-8.
[12] Henderson-Sellers, B., Stallinger, F. and Lefever, B., Bridging the Gap from Process Modelling to Process Assessment: the OOSPICE Process Specification for Component-Based Software Engineering, Procs. Euromicro 2002, IEEE Computer Society Press, 2002.
[13] Herbsleb, J., Carleton, A., Rozum, J., Siegel, J. and Zubrow, D., Benefits of CMM Based Software Process Improvement: Initial Results, Report CMU/SEI-94-TR-13, Software Engineering Institute, Pittsburgh, August 1994.
[14] Herbsleb, J., Zubrow, D., Goldenson, D., Hyes, W. and Paulk, M., Software Quality and the Capability Maturity Model, Comm. ACM, vol. 40 (1997), no. 6., pp.30-45.
[15] Hoque, R. and Sharma, T., Programming Web Components, McGraw-Hill, New York, 1997.
[16] Humphrey, W.S. and Sweet W.L., A Method for Assessing the Software Engineering Capability of Contractors, Report CMU/SEI-87-TR-23, Software Engineering Institute, Pittsburgh, September 1987.
[17] Humphrey, W.S., Managing the Software Process, Addison-Wesley, Reading Mass., 1989.
[18] ISO/IEC JTC1/SC7, High Level Design for the Revision of ISO/IEC TR 15504:1998, Document SC7 N2210, Oct. 1999.
[19] ISO/IEC TR 15504: 1998 – Software process assessment. International Standards Organization, 1998.
[20] ISO/IEC 15504, 2002, Information Technology – Process Assessment. International Standards Organization (in press).
[21] Jung, H., Hunter, R., Goldenson, D.R. and El-Emam, K., Findings from Phase 2 of the SPICE Trials, Softw. Process Improve. Pract., no.6, 2001, pp.205–242.
[22] Nierstrasz, O. and Tsichritzis, D. (eds.), Object-Oriented Software Composition, Prentice-Hall, 1995.
[23] Ojha, A., 10 Tips for Organizations Shifting to CBD, http://www.componentworld.nu/cw/lead/lead8.htm, Oct. 1999.
[24] Orfali, R., Harkey, D. and Edwards, J., The Essential Distributed Objects Survival Guide, John Wiley, New York, 1996.
[25] Paulk, M.C., Curtis, B., Chrissis, M.B., and Weber, C.V., Capability Maturity Model for Software, Version 1.1. Report CMU/SEI-93-TR-24, Software Engineering Institute, Pittsburgh, February 1993.
[26] Ring, K. and Ward-Dutton, N., Ovum Report: Overview Componentware: Building it, Buying it, Selling it, 1998.
[27] Rout, T.P., The SPICE Project: Past, Present and Future, Software Process '96, Brighton, December 1996.
[28] Rout, T.P., SPICE and ISO 15504, in Marciniak, J. (ed.), Encyclopedia of Software Engineering, Wiley, 2002.
[29] Sametinger, J., Software Engineering with Reusable Components, Springer-Verlag, Berlin, 1997.
[30] Schmid, H.A., Business Entity Components & Business Process Components, Journal of O-O Programming, Oct. 1999.
[31] Schneider, J.-G. and Nierstrasz, O., Components, Scripts and Glue, in Barocca, L., et al. (eds.), Software Architectures – Advances and Applications, Springer-Verlag, 1999.
[32] Seacord, R.C., Custom vs. Off-The-Shelf Architecture, Technical Note CMU/SEI-99-TN-006, 1999.
[33] Simon, J.-M., Assessment Using SPICE: A Case Study, in SPICE – The Theory and Practice of Software Process Improvement and Capability Determination, El Emam, K., Drouin, J.-N., Melo, W. (eds), pp. 395–420, 1997.
[34] Smith, R., The SPICE Trials, Software Process Newsletter, no. 8, Winter 1997.
[35] SPICE Project, Phase 2 Trials – Interim Report, June 1998.
[36] Szyperski, C., Component Software: Beyond Object-Oriented Programming, Addison-Wesley, Reading, MA, 1998.