PRACTITIONER PERCEPTION OF COMPONENT BASED SOFTWARE DEVELOPMENT

A. Phippen
School of Computing
University of Plymouth
Plymouth
United Kingdom
andy@jack.see.plym.ac.uk

S.M. Furnell
Department of Communication and Electronic Engineering
University of Plymouth
Plymouth
United Kingdom
sfurnell@plymouth.ac.uk

H.D. Hofmann
ABB Corporate Research Center
Heidelberg
Germany
h.hofmann@web.de

ABSTRACT

Component-orientation can be viewed as one of the leading edge technologies in software development, in particular for the development of eCommerce systems. The industrial origins of component-orientation have resulted in a knowledge base in the area that is heavily anecdotal and subject to vendor bias. Empirical evaluation of component-based techniques within industrial software development projects has resulted in a number of theories that are at odds with conventional wisdom. The surveying of practitioners enables these theories, and conventional wisdom, to be further tested. The assessment presented in this paper highlights problems with both the technologies themselves and also the wider organisational issues that need to be addressed when adopting such techniques. A combination of case study and survey research enables effective conclusions to be drawn regarding the impact of component-orientation upon the software development process.

KEYWORDS

Component based software development, COM, CORBA, Technology assessment.

INTRODUCTION

It is often stated that component-oriented software development underpins the implementation of eCommerce systems (for example, see Hess 2000). The level of software reuse afforded by such an approach ideally suits the assembly and extension approach to software development that is often seen in eCommerce applications. The influence of component-orientation can be seen in the client side of web applications, extending standard interface functionality with downloadable controls and plugins. More importantly, it plays a growing part in the extension of server side functionality – the foundation of eCommerce applications. A common approach to extending the functionality of a web server is to interface functionality developed in the form of software components with the web server object model via a component standard (for example, Microsoft's Component Object Model (Microsoft 1997) or OMG Common Object Request Broker Architecture (CORBA) (OMG 1995)).

In the more general area of software development, component-orientation is also seen as state of the art, with much literature predicting it as the model of software development that will finally enable large scale software reuse (for example, see Brown 1998, Chappell 1997, Computer Weekly 1998, Kiely 1998). However, it is difficult to assess how the development process is affected by such technologies.

While component orientation originates from academic research over thirty years ago (Mcillroy 1969), it has only been in recent times, with industry support, that component orientation has become a viable approach to software development. It is now acknowledged (see Maurer 2000) that component-orientation comes primarily from industrial innovation with little influence from the academic domain. As such, much of the information regarding the technique can suffer from lack of evidential support and from vendor bias. We can, at best, identify a 'conventional wisdom' regarding component-orientation and how it affects the development of software. We refer to this as conventional wisdom as it is knowledge that has developed without a sound body of evidence to support it. Drawing from industrial sources (for example Brown 1998, Chappell 1997, Computer Weekly 1998, Kiely 1998, McInnis 2000) we can identify a number of different outcomes through the use of component orientated techniques:

1. Component-orientation increases development productivity through software reuse.

2. Component-orientation enables cross-platform and cross-language interoperability
3. Component-orientation will reduce maintenance costs and increase reliability
4. Component development is made possible through component standards
5. Component-orientation provides functionality to aid in the distribution and scalability of applications

Research carried out by the authors has found a great deal of difference between perceived beliefs regarding the use of component-orientation and the reality of their use. The research aimed to empirically assess the impact of component-orientation upon software development. Two distinct areas of research were carried out. Initially, case studies in two large-scale industrial software projects enabled an in depth assessment of component-orientation. A second strand of research aimed to validate and develop findings from the case studies by assessing the experience of other practitioners. In initial discussion, this paper reviews the case studies and the theories drawn from them. It then focuses upon the second phase of the research, detailing the approach taken in carrying out this assessment, and discussing the findings in relation to both case study findings and also the conventional wisdom regarding component-orientation identified above. Conclusions drawn from this discussion are put forward considering the suitability of component-orientation as a future mainstream software development technique.

CHALLENGING THE PERCEPTION OF COMPONENT-ORIENTATION

Initial study into the impact of component-orientation upon software development centred upon two case studies. The first case study used a CORBA component model in the development of a telecommunications architecture across disparate network technologies. It was particularly focussed upon the integration of mobile and fixed network technologies. It was a project whose development teams were distributed across Europe, with approximately thirty developers in eight different locations. Architectural designers were also distributed in other locations across Europe

The second case study was based on a network management Independent Software Vendor (ISV) in their first year of operation. The organisation was a Small to Medium sized Enterprise (SME) who wished to componentise the business functions to be able to offer similar functionality with both traditional custom software applications and web based software products.

Propositions defined for the study of the cases focussed upon assessing the degree of conventional wisdom that would hold within a practitioner environment. In each case the adoption and use of component technologies introduced distinct problems that challenged our initial beliefs in the use of component-orientation. We developed a number of theories regarding component-orientation from case study findings:

1. Adopting and using component technologies in software development processes will affect process activities.
2. An awareness of the issues involved in the adoption and use of component technologies can ease their integration.
3. Component technologies ease the development, integration and deployment of distributed systems.
4. Uncontrolled adoption and use of component technologies can have a negative affect upon a development project.
5. Adopting and using component technologies in software development processes will affect process activities.
6. An awareness of the issues involved in the adoption and use of component technologies can ease their integration.
7. Similar issues with component-orientation occur when using different technologies from the same field (i.e. Microsoft based, rather than OMG based technologies).
8. Problems in the use of component technologies can be avoided through greater knowledge

The case studies were extremely valuable in determining that there were issues in the use of component-orientation that were not readily addressed through knowledge available to practitioners from industrial literature. However, in order to strengthen the generalisability of results, it was decided to carry out a survey of other practitioners who had used component based technologies.

SURVEY METHOD AND CONSTRUCTION

The survey was conducted in order to obtain quantifiable opinion on case study results and to assess the normality of experiences within the studies. This, in turn, would either strengthen or reject theories developed during case study research. It was decided that rather than use a traditional survey approach (for example, postal or telephone), an online, World-Wide Web (WWW) based survey would be used.

It was important to obtain responses from practitioners actively involved in the development of component-based systems. As potential respondents were to be contacted via email, a list of email addresses was required. The most effective information resource in addressing both of these requirements in obtaining responses was to go to mailing list archives in the area. By going to list archives, email

addresses could be obtained from developers who were active and experienced in the area of component-based development. In general, questions and discussion from the chosen archives (CORBA-DEV and DCOM@discusss.microsoft.com) asked in the mailing lists were also complex in nature – therefore demonstrating a good level of knowledge in the area. Additionally, two personnel from each of the earlier case studies completed the survey to see whether responses from project developers would reflect case outcomes.

The survey focussed was divided into two distinct sections:

Use of component technologies: To establish the respondent's experience using component based techniques.

Component technologies and the software development process: Focusing more upon findings from the case studies - a set of questions derived from the theories developed from the case studies.

Initial questions were generally presented in a closed form with the opportunity to elaborate for the respondents only in a few cases. However, the section of the survey derived directly from case study theory took the form of bipolar agree/disagree questions, where a statement is presented and the respondent is asked to what degree they agreed or disagreed with the statement. Based upon survey responses, it would seem that these attempts to avoid guiding the respondent to reflect case study findings were successful.

SURVEY FINDINGS

Two hundred practitioners were emailed during March 2000. Forty-three responses were obtained, providing a response rate of 22%. As expected from the type of respondents selected for the survey, experience in component orientation was good, with a mean of 3.1 years.

In terms of types of experience that respondents had, Table 1 illustrates a distinction between those with COM- and CORBA-related experience. These are broad definitions, COM experience encapsulating COM, DCOM and COM+, and CORBA experience encapsulating CORBA and Enterprise JavaBeans (EJB). As an outcome from the case studies was that there *may* be differences in experience depending on whether CORBA- or COM-related technologies are used, it was important to be able to distinguish experience based upon knowledge of the different technologies. The "neither" response came from the respondent who had used the "CORBA-like" model.

Table 1 - COM & CORBA related experience among respondents

Opinion	%age
COM related	15
CORBA related	10
Both	17
Neither	1

The level of project experience was also high. On average, respondents had used component technologies on over six projects. The projects varied in scale from small investigations right through to pan-enterprise applications. Distribution across project types was quite even, with "product" and "enterprise" projects being the most common scale. Elaboration of types of projects from respondents suggested that a good proportion (40%) of respondents had experience of component orientation in eCommerce-type applications.

Practitioner Perception based upon Case Study Theories

As previously stated, the aim of the survey was to determine the generalisability of the earlier findings from case studies. Many of the theories developed from the studies centre around the adoption of component technologies into the development process. We had found that if this adoption was not controlled severe problems could be experienced. Therefore, our first question directly addressed this issue, asking whether the respondent believed that the integration of component orientation was straightforward. While 74% of respondents stated that integration was a straightforward process, the number of negative responses is significant. Certainly, it demonstrates that our experiences in the case studies were not entirely isolated. Those who did experience problems elaborated on their response, a lot highlighting problems with the technologies themselves. Additionally, comments were made relating to organisational and personnel issues.

Question 2: Component-orientation is easily adopted into the development process

This question was posed because our first case studies seemed to demonstrate that adopting component-orientation into a development process was problematic. The results from the survey (illustrated in Table 2) would suggest that the first case experience was not the norm and that component technologies can be adopted in a straightforward manner.

It should also be noted that while the majority response for this question has been positive, there is still a fair proportion of respondents who do not believe that

component technologies are easily adopted into the development process. Therefore, while our difficult experiences were certainly in the minority, they were by no means unique.

Table 2 - Component technologies can be easily adopted

Opinion	%age
Strongly agree	0%
Agree	54%
No opinion	3%
Disagree	27%
Strongly disagree	8%
No response	8%

Question 3: Component technologies can be adopted independently of wider organisational consideration

Again, deriving from our experiences in the case studies, this is also an issue that is introduced in industrial literature, which states that an organisational embracing of component orientation is required in order to exploit its potential (for example Computer Weekly 1998, Jacobsen et. al. 1997, Kiely 1998).

There is a more or less equal split in the responses here between those who agreed or strongly agreed with the statement, and those who disagreed or strongly disagreed. If the survey respondents reflected case study findings, we would expect those who agreed with the question to have experienced problems with adoption and use (as occurred in one of our case studies), while those who disagreed had a far more straightforward adoption (as occurred in our other case study). The survey responses showed no such patterns.

Table 3 - Component technologies can be adopted independent of organisation issues

Opinion	%age
Strongly agree	10%
Agree	30%
No opinion	10%
Disagree	35%
Strongly disagree	10%
No response	5%

Question 4: Project management is unaffected by component technologies

Table 4 demonstrates a very strong response disagreeing with the statement presented in the questionnaire. It confirms one of the issues arising from one of the case studies, where component orientation was considered to be an implementation technology that was not of concern for the project management. This response greatly strengthens the opinion that this approach to the use of component technologies was wrong, and that project managers need to be aware of the issues in their use as much as developers.

Table 4 - Project management is unaffected by component technologies

Opinion	%age
Strongly agree	7%
Agree	5%
No opinion	2%
Disagree	45%
Strongly disagree	36%
No response	5%

Question 5: Component-orientation makes software reuse easy

One of the underlying philosophies of component orientation is that it makes software reuse possible on an industrial scale. Industry literature (for example Chappell (1997), McInnis 2000) is especially keen on the reuse aspect of component orientation. The case studies had experienced mixed results in generating large-scale reuse: the firs case study had not been at all successful in developing reusable components, whereas the second developed a highly reusable component library. The response from respondents in the survey (see Table 5) would also indicate that the first case study experience was not typical - the majority of respondents either agreed or agreed strongly with the statement.

Table 5 - Component orientation makes software reuse easy

Opinion	%age
Strongly agree	26%
Agree	55%
No opinion	2%
Disagree	10%
Strongly disagree	2%
No response	5%

However, a significant proportion (28% in total) either disagreed or strongly disagreed. This promoted an examination of responses against the type of technologies used. It was found that the majority of negative responses came from respondents who only had experience with CORBA technologies. This would highlight a different in reuse based upon the choice of technology.

Question 6: Using component technologies is straightforward

This question relates to the complexity of component technologies, in the view of practitioners who have used them. This, in turn, impacts upon their adoption into the

mainstream. The interest arises in comparison with some of the more positive responses (for example, the responses to the questions "Adoption is straightforward" and "reuse is easy"). One might assume that those positive outcomes signal the ease of use of component technologies. However, the fact that the majority response were to the contrary suggests that it is only when developers are fully aware of issues in the use of technologies that they become truly easy to use.

Table 6 - Using Component Technologies is Straightforward

Opinion	%age
Strongly agree	7%
Agree	26%
No opinion	17%
Disagree	43%
Strongly disagree	2%
No response	5%

Question 7: Component based development makes system deployment easier
Question 8: Component based development makes system maintenance easier

The final two questions addressed issues related to the underlying philosophy regarding the use of component technologies that yielded inconclusive results from the case studies. Initial results (see Table 7) show divided opinion on the issue of deployment.

Table 7 - Component based development makes system deployment easier

Opinion	%age
Strongly agree	17%
Agree	24%
No opinion	21%
Disagree	26%
Strongly disagree	7%
No response	5%

Another purported strength of component orientation is that it eases system maintenance. Theoretically, the use of interfaces, black box and binary reuse means that a component can be bug-fixed and plugged into a live system without any component clients needing to be brought down in the maintenance (for example, see Szyperski 1998). As this issue could not be tested in either of the case studies (as, in each case, they were only studied until the first version release of the software), this final question was used simply as a test of practitioner experience. It would seem, given the positive responses to the question that this aspect of component orientation is borne out by practitioner experience.

Table 8 - Component orientation makes system maintenance easier

Opinion	%age
Strongly agree	26%
Agree	55%
No opinion	2%
Disagree	10%
Strongly disagree	2%
No response	5%

COMPARING SURVEY RESPONSES WITH CASE STUDY THEORY

This section considers how the survey responses have influenced the theories regarding component orientation developed from the case studies:

Adopting and using component technologies in software development processes will affect process activities
There are some very positive responses in the survey that strengthen this proposition. In particular questions related to project management, deployment and all resulted in responses that would confirm the effect the component-orientation has on development activities.

An awareness of the issues involved in the adoption and use of component technologies can ease their integration
The major theme that runs through responses in this survey reflects the fact that learning and understanding of component technologies is *the* issue in using them. Therefore, this theory has been greatly strengthened by survey results.

Component technologies ease the development, integration and deployment of distributed systems
The distributed aspect of component-based development was not explicitly addressed in the survey, but positive responses to question 8 highlight the fact that component technologies can be used to address the low level elements of distributed development.

Uncontrolled adoption and use of component technologies can have a negative affect upon a development project
Drawing from the central outcome of the survey relating to the need for understanding, the proposition is demonstrated to have some validity. Undoubtedly, the experiences of the first case study are very much in the minority among component practitioners. They are not, however, unique. This in itself strengthens the issues identified in this case study as possible outcomes when using component technologies, if such use if not carefully considered.

Similar issues with component-orientation occur when using different technologies from the same field (i.e. Microsoft-based, rather than OMG-based technologies)

Several questions have highlighted differences in experience relating to the types of technologies used by respondents. However, we cannot illustrate any explicit trends throughout the survey (i.e. there is nothing to suggest that CORBA will always result in poor development, whereas COM will always results in effective development). Therefore, once again, were are drawn back to the issue of front-loading knowledge when using component-oriented techniques – with an awareness of the issues and an understanding of the technologies, effective development can be achieved, regardless of their type.

Problems in the use of component technologies can be avoided through greater knowledge of the technologies involved

It has certainly been illustrated in the case study that awareness and understanding are the important issues in using component technologies.

CONCLUSIONS

Component technologies represent a significant contribution to the domain of software engineering and the deployment of related systems. However, evidence suggests that whilst the conceptual advantages of such approaches are recognised, the practical experiences of developers are often somewhat different. The survey results indicated that while component-based development can indeed be seen to deliver benefits, these are most likely to be realised if the correct perception of the technology has been adopted by both the developers and their parent organisations. An understanding and appreciation of the propositions supported by the survey will ensure that such a perception can be successfully achieved.

REFERENCES

Hess (2000). "A .NET Primer". Microsoft Corporation. http://msdn.microsoft.com/workshop/essentials/hess/hess12112000.asp

Brown, J.(1998) "Software Strategies – The Component Decision", The Forrester Report. http://www.forrester.com/

Chappell, D. (1997). "The Next Wave – Component Software Enters the Mainstream". Chappell & Associates. http://www.chappell.com

Computer Weekly (1998). "Forget Objects, Use Components". Computer Weekly, 19 November 1998.

Jacobsen, I., Griss, M., Jonsson, P. (1997) "Software Reuse - Architecture, Process and Organization for Business Success". ACM Press. ISBN 0-201-92476-5

Kiely, D. (1998). "The Component Edge – An Industrywide Move to Component-Based Development Holds the Promise of Massive Productivity Gains". Information Week April 13, 1998.

Maurer (2000). "Components: What If They Gave a Revolution and Nobody Came?". IEEE Computer June 2000.

McIllroy, D. (1969). "Mass Produced Software Components". Appearing in Naur, Randell & Buxton (eds.) (1976). "Software Engineering Concepts and Techniques – Proceedings of the NATO Conferences". Petrocelli/Charter. New York.]

McInnis (2000). Component Based Development: Concepts, Technology and Methodology. Castek Software Factory Inc. http://www.cbd-hq.com/

Microsoft (1997). "Windows DNA – Windows Distributed interNet Applications Architecture". http://www.microsoft.com/dna/

Microsoft Corporation (1997). "Vertical Industry Specifications Supported in Windows DNA". http://www.microsoft.com/industry/

OMG (1995). "CORBA: Architecture and Specification. Version 2". OMG 1995.

Rogerson, D. (1996)." Inside COM". Microsoft Press. ISBN 1-572-31349-8.

Schuman & Presser (1996). "Questions and Answers in Attitude Surveys". Sage Publications. ISBN 0-7619-0359-3.

Szyperski, C. (1998). "Component Software – Beyond Object-oriented Software". Addison-Wesly. ISBN 0-201-17888-5