Software Component Models: Past, Present and Future

Kung-Kiu Lau
School of Computer Science
The University of Manchester, UK
kung-kiu@cs.man.ac.uk

ABSTRACT

In the early years of the CBSE Symposium, much research was focused on identifying the desiderata of CBSE [3] and developing different approaches to CBSE. However, a common framework for defining and analysing CBSE approaches with respect to these desiderata was only introduced later: this was provided by the notion of *component models* [9, 12, 13, 6]. Every CBSE approach is underpinned by a component model, and therefore the study of component models, in particular how to define ones that can potentially meet the desiderata of CBSE, is pivotal to the success of CBSE.

We have surveyed and studied existing CBSE approaches and their corresponding component models [12, 13], and as a result we have: (i) shown that early approaches/models do not fully meet the CBSE desiderata; (ii) identified criteria for designing component models that can better meet the CBSE desiderata; (iii) defined a new component model according to (ii); (iv) defined a taxonomy of existing component models based on the desiderata.

In addition to the classic desiderata described in [3], nowadays CBSE has to address new challenges posed by an unprecedented increase in the scale and complexity of software applications, in particular safety-critical ones. As a result, there are new CBSE desiderata for which we need to define new models.

In this tutorial, we will: (i) present a taxonomy of existing component models, both old and new; (ii) discuss how well they meet the classic desiderata; (iii) discuss criteria that new models must meet in order to address future CBSE challenges.

The CBSE Symposium celebrated its fifteenth anniversary in 2012 [14]. For the next 15 years, the study of component models will continue to play a pivotal role in future CBSE success. This tutorial aims to contribute to this effort.

CBSE'14, June 30–July 4, 2014, Marcq-en-Baroeul, France.
ACM 978-1-4503-2577-6/14/06.
http://dx.doi.org/10.1145/2602458.2611456 .

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: [component-based software engineering]

General Terms

Software Component Models

Keywords

Software Components; Composition

1. SOFTWARE COMPONENT MODELS

The cornerstone of any CBSE methodology [16] is its underlying component model [9, 12, 13, 6], which defines what components are, how they can be constructed and represented, how they can be composed or assembled, how they can be deployed; and how to reason about all these operations on components in terms of the usual accompanying quality metrics.

The tutorial will look at the past, present and future of component models.

1.1 The Past

Early CBSE research focused on identifying desiderata and on defining different approaches to CBSE.

Classic CBSE Desiderata. Components should pre-exist; what components are; components should be produced and deployed independently; composites should be possible; components should be copiable and instantiable, etc. Widely accepted, 'classic', desiderata are described in [3].

Idealized Component and System Life Cycles. An idealised component life cycle is one that meets the CBSE desiderata. An idealised system life cycle uses pre-existing components developed during the component life cycle; it thus intersects the idealised component life cycle at the point at which components have been built.

Early Component Models. Early component models are based on: (i) object-oriented frameworks, e.g. CCM [15] and EJB [7], where components are objects and composition is by object delegation; and (ii) first-generation ADLs (architecture description languages), e.g. ACME [8] and ArchJava [1], where components are architectural units (with ports) and composition is by port linking.

Early Component and System Life Cycles. Early component life cycles deviate from the idealized one. Early system life cycles often subsume component life cycles, i.e. they do not use pre-existing components but rather identify and develop components afresh for each system.

1.2 The Present

Currently, the majority of component models in use are based on second-generation ADLs, e.g. Fractal [4] and SOFA [5]. Whereas first-generation ADLs generally lack tool support, in particular for component repositories and for connector generation, second-generation ADLs provide much more tool support.

Current Component and System Life Cycles. Current component life cycles also deviate from the idealized one somewhat, mainly in the use of repositories.

New Component Models. New component models have been defined that are not based ADLs: web services [2] and X-MAN [11]. These models use coordination as a composition mechanism, and better meet the CBSE desiderata than earlier models.

A Taxonomy of Component Models. The result of our study of component models, old and new, is summarised in a taxonomy based on the classic desiderata. This taxonomy has five categories, and include the following component models:

ACME-like ADLs, CCM, COM, EJB, Fractal, JavaBeans, Koala, KobrA, .NET, OSGi, Palladio, Pin, ProCom, SOFA, UML 2.0, PECOS, X-MAN, and Web Services.

1.3 The Future

The classic CBSE desiderata alone are not sufficient for meeting new challenges for the future.

Scale and Complexity. An unprecedented increase in the scale and complexity of software applications poses new challenges. CBSE is well-placed to meet these challenges, but to do so it must meet additional desiderata.

Safety. Safety is another challenge. An increase in scale and complexity makes it more difficult to ensure safety. Again, CBSE is well-placed to meet this challenge, but has to meet additional desiderata to do so. For V&V, adapting the V model [17] for CBSE needs to be done correctly.

Future Component Models. Future component models must have compositionality in all aspects relevant to scale, complexity and safety. Compositionality to ensure scale and complexity entails hierarchical system construction; whereas compositionality to ensure safety entails compositional V&V.

Future Component and System Life Cycles. To support compositionality, new life cycles like the W Model [10] have to be developed.

2. REFERENCES

[1] J. Aldrich, C. Chambers, and D. Notkin. Component-oriented programming in ArchJava. In *First OOPSLA Workshop on Language Mechanisms for Programming Software Components*, pages 1–8, 2001.

[2] G. Alonso, F. Casati, H. Kuno, and V. Machiraju. *Web Services: Concepts, Architectures and Applications*. Springer-Verlag, 2004.

[3] M. Broy, A. Deimel, J. Henn, K. Koskimies, F. Plasil, G. Pomberger, W. Pree, M. Stal, and C. Szyperski. What characterizes a software component? *Software – Concepts and Tools*, 19(1):49–56, 1998.

[4] E. Bruneton, T. Coupaye, and M. Leclercq. An open component model and its support in Java. In *Proc. 7th CBSE, LNCS 3054*, pages 7–22. Springer -Verlag, 2004.

[5] T. Bures, P. Hnetynka, and F. Plasil. SOFA 2.0: Balancing Advanced Features in a Hierarchical Component Model. In *Proc. SERA 2006*, pages 40–48. IEEE, 2006.

[6] I. Crnkovic, S. Sentilles, A. Vulgarakis, and M. Chaudron. A classification framework for software component models. *IEEE Transactions on Software Engineering*, 37(5):593–615, Oct. 2011.

[7] L. DeMichiel, L. Yalçinalp, and S. Krishnan. *Enterprise JavaBeans Specification Version 2.0*, 2001.

[8] D. Garlan, R. Monroe, and D. Wile. ACME: An architectural interconnection language. In *Proc. CASCON'97*, pages 169–183, 1997.

[9] G. Heineman and W. Councill, editors. *Component-Based Software Engineering: Putting the Pieces Together.* Addison-Wesley, 2001.

[10] K.-K. Lau, F. Taweel, and C. Tran. The W Model for component-based software development. In *Proc. 37th EUROMICRO Conference on Software Engineering and Advanced Applications* (*SEAA*), pages 47–50. IEEE, 2011.

[11] K.-K. Lau and C. Tran. X-MAN: An MDE tool for component-based system development. In *Proc. 38th SEAA*, pages 158–165. IEEE, 2012.

[12] K.-K. Lau and Z. Wang. A taxonomy of software component models. In *Proc. 31st SEAA*, pages 88–95. IEEE Computer Society Press, 2005.

[13] K.-K. Lau and Z. Wang. Software component models. *IEEE Transactions on Software Engineering*, 33(10):709–724, October 2007.

[14] J. Maras, L. Lednicki, and I. Crnkovic. 15 years of CBSE Symposium – impact on the research community. In *Proc. 15th CBSE*, pages 61–70. ACM, 2012.

[15] OMG. *CORBA Component Model, V3.0*, 2002. `http://www.omg.org/technology/documents/formal/components.htm`.

[16] C. Szyperski, D. Gruntz, and S. Murer. *Component Software: Beyond Object-Oriented Programming.* Addison-Wesley, second edition, 2002.

[17] The V-model. Development standard for IT-systems of the Federal Republic of Germany, IABG. `http://www.v-modell.iabg.de`.