Component-based Development Infrastructure: A Systematic Approach

Leonid A. Kalinichenko

Institute for Problems of Informatics
Russian Academy of Sciences
Vavilov 30/6, Moscow, V-334, 117900
Phone: 7 095 2372031, Fax: 7 095 3107050
E-mail: leonidk@synth.ipi.ac.ru

1 Introduction

It is widely recognized now that OMG needs serious reconsideration of the middleware architecture to be used as a basis for componentware. Current OMA release was very good for the beginning, for the core interoperation modeling. Deficiencies of the current strategy for making systems with reuse are well known. We cite here just two recent papers providing good analysis of the situation.

"From the beginning, high level goals of OMG have included reusability, interoperability, extensibility, and dynamic evolution. OMG customers have wanted the ability to assemble applications quickly from standard parts, and OMA framework users were seeking consistent semantics for how shared services work when using the full suite of OMG specifications. We have made good progress on separable specifications but have said too little about composition of services and facilities. Current specifications do not provide enough information to realize the desirable the goals of being able to mix-and-match and plug-and-play with alternate implementations of services or facilities." [1]

"With IIOP interoperability for CORBA, and a diverse set of distributed object services, the OMG Object Management Architecture has come of age as a tool for expert programmers to generate complex multi-tier systems. It is now time for OMA and CORBA to take the next step, by providing a component framework enabling CORBA objects, representing both client-side and non-visual server-side functionality, to be assembled into systems, using visual development tools and scripting languages, by unsophisticated programmers." [2]

"If CORBA distributed objects are to dominate the software industry in the near future, the Object Management Architecture must include the infrastructure that allows unsophisticated programmers to assemble encapsulated and self-describing distributed objects ... Only by this measure can CORBA and the OMA drive the mass market for distributed multi-tier computing." [2]

"Current OMG service specifications assume that the application developer understands all the relevant services and invokes them when appropriate." [1]

"Principled selection and decomposition into subcomponents can enable substitution and customization by developers other than the original designers. But there are reasons to hide containment information as well: A puzzle for OMG is how to represent the list of behaviors (before and after methods, a schedule, context variable, and so forth) to insure proper control." [1]

"CORBA has the potential to dominate business application building, but will only realize that potential if the OMG defines a standard component infrastructure ..." [2]

What are the solutions proposed to improve the technology ? We give examples of just few of them.

"JavaBeans has rapidly emerged as an important standard for component objects. The existing synergy between CORBA and Java would be greatly enhanced by defining a CORBA component model compatible with Java Beans." [2]

"Properties, in the JavaBeans sense, are part of the externally -accessible view of the object, and are accessed by get- and set- methods. JavaBeans properties also encompass the notion of generating change events on modification of particular properties - termed bound and constrained properties. We need to supplement CORBA to support a similar notion... There are two possible ways to make additional meta-data for properties and events visible in the object declaration. The most straightforward way to do so is to add these concepts as fundamental elements of IDL. An alternative would be to supplement IDL with a separate component definition language." [2]

"If OMG is to produce a viable, widely-used component technology, there is great benefit in specifying a standard scripting language for gluing together components. Javascript is one possible scripting language that could be endorsed by OMG." [2]

"Given that most of these new component-based applications will be deployed on the Web, there is a need to name CORBA objects using mechanisms and conventions popular there. A URL specification for naming CORBA objects that allows clients to name objects on any host in the internet is needed." [2]

"In order to support a powerful component model the concepts of a component supporting multiple independent interfaces and of a component aggregating a set of sub-components, possibly exposing their interfaces, are very powerful. This is something that many component models support and JavaBeans is evolving to do so as well. The OMG is currently entertaining submissions to support multiple interfaces in CORBA to support these concepts." [2]

"Semantics: there is already activity in OMG to address the need to say more about specifications than just their interface and an English explanation of their intended semantics." [1]

"Object Model Extensions:...

* CDL or JDL - the JBOF CDL extended IDL with ODMG ODL which includes ODMG relationships (which differ from OMG Relationships Service), extensions for collections of instances of a class and keys, and some other additions. Some modelers have found these extensions useful..." [1]

The perspectives for the CORBA Component Roadmap are inferred of these and other proposals, like the following:

"Update the "Technical Objectives" of the OMA Guide chapter to modularly include:

- technical objectives on architectural principals and ilities (component model and packaging, scalability by composition and federation, evolution, real-time and QoS.)
- technical objectives for ORB, services and facilities like trader, language mappings, scripting language
- technical objective(s) on domain interfaces interoperability with industry standards including the Web, Java and COM/ActiveX". [1]

The proposals like the above actually are important and are good to follow. Many arguments for further development are based on comparison with competing componentware architectures (OMG is already behind) to get good ideas out of them leading to partial improvements for OMG. Probably, this is also a good compromise between a desire to make serious architecture reconsiderations and conservative nature of the standardization consortium.

But still this is a continuation of 'first act - then think' approach instead of developing of more well grounded strategy for survival in more durable perspective.

To switch to the latter in the componentware infrastructure it is required to get finally to sound understanding of the essence of the component-based development and reusability problem leading to proper component-oriented modeling, compositions and design in the CORBA world instead of taking of ad hoc solutions without deep understanding of the role of various infrastructure components and their interdependencies (UML is just one example of recent ad hoc solutions by OMG).

We should clarify why we are looking for more complete and precise semantics intended for service specification decomposition/composition in a course of design with reuse, what shall we do with such semantic definitions when we get them, why do we need component definition model distinct of other modeling facilities, how we shall use it, how object analysis and design modeling as well as IDL will be correlated with the component definition models, and so forth.

Otherwise, OMG will have to continue the fluctuative evolution following the arguments like: 'JavaBeans seems to be doing well', 'Javascript is great' or 'some modelers found CDL extensions useful', let's try these directions immediately (because OMG does not know better ones).

To show that situation with the proper understanding of the essence of the component-based development is not hopeless, we shall briefly refer to some results of just one research project devoted to the interoperable component-based information systems development. This is the SYNTHESIS project that is underway at the Russian Academy of Sciences [3, 4, 5, 6].

2 SYNTHESIS project approach

The project systematically investigates various component-based development issues, including:

1. what is the required level of completeness of specifications for the component-based development
2. what are the required canonical modeling facilities to uniformly specify heterogeneous pre-existing components with the required completeness
3. what is the required model to specify the information system requirements (information system analysis model) with the completeness needed and how this model is to be related to the canonical component specifications (e.g., is this one and the same model or are these models different ?)
4. what are the required modeling facilities for the component-based design (design of an analysis model reusing pre-existing components) - is this an additional model or one of the above
5. how standard object models (like IDL, ODL) should be correlated to the above
6. what are suitable representational facilities for the models above (graphical, diagrammatic, etc.)
7. how to identify component *capabilities* that can provide a support of a specific application (specification of requirements)
8. how to correctly decompose specifications of pre-existing components into smaller fragments (capabilities) potentially relevant for specific requirements
9. how to correctly compose relevant component capabilities to get a *refinement* of a respective part of a specification of requirements
10. what is the minimal set of required formal facilities and methods to justify different steps of the component-based design process
11. how the required formal facilities should be related to the canonical and requirements modeling facilities above

Investigations showed that the following decisions are appropriate (respectively to the issues above):

1. specifications should define contextual information of the application in form of ontologies as well as provide complete structural and behavioral information for the information system entities including complete specifications of type operations and workflows
2. canonical modeling facilities should provide for complete abstract data type specifications - of an object and non-object kinds (including complete specification of operations and workflow types), subtyping (with rigorous semantics) and classification (metatype based) hierarchies, support of the spectrum covering unstructured, semi-structured and structured data, treating type attributes as type definitions in their turn, a complete type specification algebra producing new type specifications from the existing ones, object calculus support with formulae leading to a possibility of creating collections of values of new types according to a type specification algebra

3. modeling facilities for all phases of the component-based development (including ontological specifications) are those of the canonical component specification model
4. the design is based on the canonical model facilities having rich possibilities for type specification compositions
5. there should exist a mapping of IDL, ODL and other standard specifications into the canonic specification model
6. any standard representation facilities can be suitable (e.g., a UML subset with some extensions may be quite good): but the approach is different from now, it is the canonical model that has primary importance providing proper interpretation and meaning for graphical representation
7. the identification of component capabilities is based on the ontological relevance of component constituents to the constituents of an analysis model and on an ability of such component constituents to rigorously *refine* the respective constituents of the analysis model
8. the ability to decompose is based on properties of the type specification algebra of the canonical specification model
9. the ability to compose is based on properties of the type specification algebra of the canonical specification model and on proper methods leading to rigorous formation of refinements of analysis model specifications
10. the required formal facilities and respective design procedures ahould provide for justification of: (i) consistency of specifications of requirements and of component specifications, (ii) correctness of abstraction of original component specifications by the canonical ones, (iii) identification of truly relevant component capabilities for a given specification of requirements, (iv) correctness of capabilities compositions, (v) validity of refinement of a specification of requirements (or its part) by a composition of pre-existing capabilities
11. there should exist a mapping of the canonical modeling facilties to the formal ones.

The canonical modeling facilities possessing the features above were defined, principles of mapping of the canonical specifications to a formal notation were elaborated, methods of component-based design relying on the canonical model, its type algebra (making possible proper type specifications decomposition and compositions) and mapping to the formal notation have been developed, the process of the component-based design using graphical representation facilities for the developers has been developed and partially prototyped, methods for abstracting of canonical component specifications from the original heterogeneous ones have been defined, a mapping of ODL to the canonical model has been specified.

3 Conclusion

Component-based information systems development is a complicated technology that requires serious investigations and systematic approach. Ad hoc solutions on separate constituents of the technology do not lead to a suitable practice. The constituents of the technoloy are so highly interdependent that only a systematic approach for their elaboration can give proper results.

The SYNTHESIS project shows that the systematic approach for the component-based information system development is feasible.

References

[1] Thomson C., Linden T., Filman B. The next generation object management architecture. http://www.mcc.com/projects/oip/

[2] CORBA Component Imperatives. ORBOS/97-05-25. IBM Corporation, Netscape Communications Corporation, Oracle Corporation, Sunsoft, Inc.

[3] Kalinichenko L.A. Emerging semantic-based interoperable information system technology. In *Proceedings of the International Conference Computers as our better partners*, Tokyo, March 1994. World Scientific.

[4] Berg K. and Kalinichenko L.A. Modeling facilities for the component-based software development method. In *Proceedings of the Third International Workshop ADBIS'96*, Moscow, September 1996.

[5] Kalinichenko L.A. Method for data models integration in the common paradigm. In *Proceedings of the First East European Workshop 'Advances in Databases and Information Systems'*, St. Petersburg, September 1997.

[6] Kalinichenko L.A. Workflow Reuse and Semantic Interoperation Issues. In *Advances in workflow management systems and interoperability.* A.Dogac, L.Kalinichenko, M.T. Ozsu, A.Sheth (Eds.). NATO Advanced Study Institute, Istanbul, August 1997