DESIGNING A CULTURAL HERITAGE SECTOR BROKER USING SDBC

Boris Shishkov

Abstract: Among the actual cultural-heritage-related problems is the one of effectively managing and globally distributing digitized cultural (and scientific) information. The only feasible way to realize this goal is via the Internet. Hence, a significant issue to be considered is the adequate design of software applications which to realize brokerage tasks within the global space. However, due to the great complexity of this cultural-heritage-related task (compared to other brokerage tasks successfully realized by software systems), the usage of the existing popular modeling instrumentarium seems inadequate. Hence, in this paper, an approach is presented and it is briefly discussed how the approach could be useful for building cultural heritage sector brokers.

Keywords: SDBC; Software broker; Cultural heritage

Introduction

As it is well-known, several types of activities are observed, which concern the digitization of cultural and scientific heritage. Among them are:

- the classification of existing cultural heritage materials;
- the recognition and processing of images from such materials;
- the specification and maintenance of the metadata related to digitized materials;
- the management and distribution of the digitized cultural and scientific materials.

In the paper, this last issue will be addressed. In tune with the current technological possibilities and user demands, such management and distribution should be considered in global respect. It is necessary that the digitized cultural and scientific materials are globally available to the public. Next to that, their accessibility must be regulated. This is not easy because some materials are to be accessible freely by anyone, others should be accessible only by authorized users, still others are to be distributed commercially, and so on. Thus, an advanced brokerage functionality needs to be realized in this regard. As far as the 'global information space' is concerned, this job is to be done by a 'software broker'. By 'software broker' it is meant a software application which realizes a brokerage functionality. It is well-known that software brokers exist and are used for a number of purposes, for instance, flight/accommodation reservations, e-Business, Tele-Work, and so on. However, a global management and distribution of digitized cultural (and scientific) data, which is characterized by a number of restrictions (as above mentioned), makes the brokerage task more complex than what is usually observed in brokerage systems. It is, therefore, essential specifying such a (software) brokerage system based on a sound consideration of the original business system, to be supported by it. This leads to a more general actual research problem, namely the alignment between business process modeling and software specification.

Further in the current section, this issue is addressed and afterwards, the software brokerage systems and their relation to the cultural heritage aspect are considered. Software applications are supposed to have the crucial role of an intermediary between the technology (in particular – Information and Communication Technology – ICT) and the (business) processes supported by it. Hence, an essential issue for current business development is the effective application support. Considering the specification of applications that should support business processes, one frequent cause of software project failure is the mismatch between the users' requirements and the actual functionality of the delivered application. Most of the current software design methods are characterized by a lack of proper alignment between the consideration of users' requirements and the specification of a software conceptual model. Actually, we observe two opposite phenomena [Shishkov, 2002]. On one hand, we observe software being developed without prior adequate investigation of the business processes to be supported by it. This means that the business requirements are poorly determined and the software design model is not rooted in a business process model. Therefore, the developed applications would support the business processes inadequately. Hence, although the applications' quality might be high from a software point of view, the effectiveness of the support they offer to the target business processes would remain low.

On the other hand, although sound business process modeling is often conducted prior to the design of applications, the business process model is only partially used, since it is not straightforwardly transformable into a relevant input for the application design. This does not allow for full employment of the ICT possibilities in solving the particular business problem(s).

Therefore, the two outlined tasks need to be aligned in a better way: the business process modeling and the specification of software applications for the support of the business processes. They both should be considered as one integrated task.

Many researchers have addressed issues related to these problems [Shishkov & Dietz, 2004-2]. Dehnert and Rittgen present a formal representation for describing business processes. This is a promising step and could be especially useful if further related to software design. Olivera, Filho and Lucena have also contributed in this direction, by investigating the design of software on the basis of business requirements analysis. Their suggested approach is a step ahead even though it does not yet offer a straightforward mapping of a business process model into a software design model. Hikita and Matsumoto have studied how the appearance of additional requirements could be reflected in the system's construction, which is also a promising result achieved so far (although not completely solving the problem). Krutchen suggests (based on the existing use case concepts [Jacobson et al, 1992]) a "Business use case" – considered useful in bridging business process modeling and software design [OMG, 2004]. But it is still a question how to consistently identify such use cases. These are only some of the examples of research activities addressing the mentioned problem, having reached no complete and convincing solution. Therefore, it might be concluded that further knowledge is still required in the direction of soundly basing application design on business process modeling.

With regard to the outlined research problem, a promising contemporary approach for application development is the component-based development [Jacobson et al, 1992], founded on the principles of object-orientation (OO). As it is well known, the OO paradigm (characterized by the fundamental concepts of encapsulation, classification, inheritance and polymorphism) is widely considered as a special approach to the construction of models of complex systems, in which a system consists of a large number of objects. According to some researchers, this could be applied not only to software systems but also to business systems [Shishkov & Dietz, 2004-1]. Thus, it seems feasible to expect that software specification and business process modeling could be bridged by basing the specification on software components which are derived from some business components. Such components should fill the gap between the two mentioned tasks. If re-usable components are identified, they could be used many times for designing different applications. Next to that, component-based development seems beneficial for the application design itself. By basing application development on encapsulated, individually definable, re-usable, replaceable, interoperable and testable (software) components, developers could build applications which possess durable configuration and a high degree of flexibility and maintainability. The process of application development would also be improved because building new applications would include using already developed components. This reduces development time and improves reliability. The performance and maintenance of developed applications would be enhanced because changes could occur in the implementation of any component without affecting the entire application. All this makes the component-based application development much more effective than the traditional way of application development.

For all these reasons, in considering the problem of alignment between business process modeling and conceptual software specification, the focus is, in particular, on realizing this on the basis of (re-usable) business components identified from target business processes. By basing the design of applications on such components, it is claimed that the application support to business processes can be improved considerably. A resulting effect from applying a component-based business/software alignment would be improvements in specifying powerful software systems that realize advanced brokerage functionality. Such a functionality would be adequately derived on the basis of the original business requirements. This would make such systems much effective in solving problems in the context of the originally existing business environment.

A software broker, specified in such a way, would be useful for some cultural-heritage-related activities, as mentioned before. They could be significantly facilitated via such an advanced software support. This justifies the current study and inspires the efforts to bring together the cultural heritage domain and latest software design achievements. Additional motivation brings the fact that no information was found about a cultural heritage project or initiative where the brokerage problem has been adequately solved. In this paper, the SDBC (SDBC stands for Software Derived from Business Components) approach is considered as an adequate software specification tool as far as the discussed brokerage problem is concerned. SDBC is a conceptual modeling approach introduced as a way to improve the quality of the software production process. The approach allows for soundly aligning (in a component-based way) business process modeling and software specification, and could be particularly useful in specifying software brokers. These issues are addressed also based on the consideration of SDBC. Besides this, the particular relevance to the cultural heritage issue is discussed.

The outline of the paper is as follows: a consideration of SDBC; an analysis of the applicability of SDBC for building software brokers in the cultural heritage context; conclusions.

The SDBC Approach

The SDBC approach is intended for supporting the alignment between business process modeling and software specification, within the software development context. Theoretical analyses, case studies, and expert opinions [Shishkov & Dietz, 2004-2] support the claim that SDBC provides appropriate mechanisms for the support of the specification task, concerning actual domains such as e-Business, Tele-Work, Virtual Organizations, and so on. The main contribution of the approach is the provision of the essential alignment between a model of the business system to be supported by software and the specification of the (software) system-to-be. Next to that, the derived specification model would be consistent with the current software standards, such as UML [OMG, 2004] and XML [XML, 2004]. Moreover, in such a derivation, SDBC allows for re-use of modeling components. The above-mentioned advantages of SDBC, which distinguish it from the currently used popular software design methods (such as KobrA, Catalysis, Tropos, and so on) [Shishkov & Dietz, 2004-1], relate to a number of issues among which is the adoption of the component-based modeling and specification. The component-based perspective has been addressed in the previous section.

Due to the limited scope of this paper and also because of the availability of relevant sources [Shishkov & Dietz, 2004-1,-2], the SDBC approach will not be introduced. Instead, a brief outline of some essential SDBC issues is presented below, supported by Figure 1.

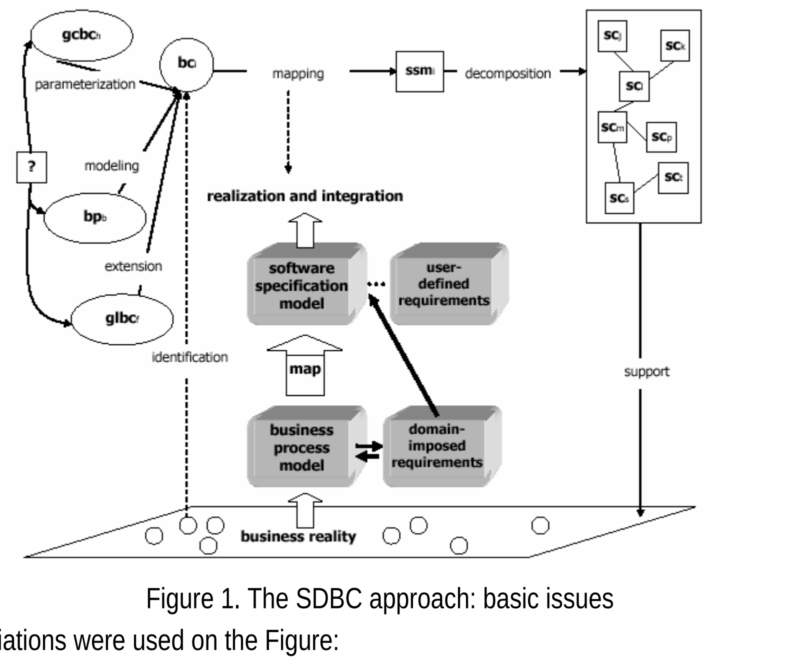


Figure 1. The SDBC approach: basic issues

The following abbreviations were used on the Figure:

bp = business process

bc = business component

glbc = general business component

gcbc = generic business component

ssm = software specification model

sc = software component

Due to the limited scope of this paper, definitions are not presented; interested readers could find them in the SDBC materials which were already mentioned.

As seen from the Figure 1, any business reality is viewed as composed of business components (related to corresponding business processes). Hence, an essential issue is the identification of business components. The identification mechanism is introduced in [Shishkov & Dietz, 2004-1]. According to it, a business component could be identified either by modeling of a business process (the trivial way) or by re-using ‘pre-fabricated’ components. SDBC distinguishes between two types of such re-usable patterns, namely general business components and generic business components. A general business component is a model in which some core functionality is grasped (for example, a general reservation unit). Hence, in order to derive a business component, based on a general one, it is necessary to apply extension (for example, extending a general reservation model into a hotel booking model). As for generic business components – they have in themselves several functionalities captured. Any of them could be selected as an option through parameterization (for example, if there is an accounting system, it might have two options: to work on the basis of the British law or to work on the basis of the French law; therefore, a user could adjust the system and use either of these options). Summarizing, a business component could be identified either by modeling of a business process, or by extending a general business component, or by parameterizing a generic business component. Since within SDBC, a business component is a model of a part of a business system, a crucial question is what are the demands towards such a model. As explained in [Shishkov & Dietz, 2004-1], the model should be soundly elaborated in the following three perspectives: structural perspective (concerning transactions, corresponding actors and their interrelations), dynamic perspective (concerning the transactions workflow representation), and communicative perspective (concerning the inter-actor exchange of communicative act which accompany the actual execution of transactions and are, therefore, of significant importance in specifying a software system if the goal is to adequately integrate such a system in the original business environment). Based on analyses of a number of relevant modeling techniques, DEMO [Dietz, 1999] has been selected for the purpose of facilitating SDBC as far as business components are concerned. The basic reason is the allowance (provided by DEMO) to adequately consider each of the above mentioned three perspectives. Hence, within SDBC, the identified business components are modeled using the notations of DEMO.

After being identified, a business component is to be mapped towards a software specification model. At this stage, the requirements related questions are to be considered. The domain-imposed requirements (those concerning the original business model) are to be derived based on the business component. The user-defined requirements (those especially formulated by the future users of the system-to-be) are added when specifying the software model. As for the specification model itself, it should be fully consistent with the current software design standards, as mentioned before. Hence, it is aimed that SDBC provides a UML-based software specification output. This means that the software specification model derived from a business component should be built with the use case notations; as it is well known, use cases are the modeling constructs serving to link the application domain (the business world) and the software domain, in a UML-based design of software. Thus, taking into account that business components are to be built using DEMO and that the resulting software specification model is to be built with the use case notations, a DEMO – use case mapping is an essential problem within SDBC. This particular problem has been addressed in [Shishkov & Dietz, 2003].

Further on, once derived, the software specification model is to undergo decomposition in order to give way to the identification of software components based on which the software system-to-be should be constructed. The reason for considering components also at this stage of the software creation process is related to the re-use possibilities which result from the component-based way of modeling and designing. Hence, by ‘software component’ is to be meant a (re-usable) part of a software specification model. Thus, the software components in SDBC are not physical components (associated with the current ‘physical’ component technologies, such as CORBA, .NET, J2EE/EJB, and so on), but rather ‘logical’ components representing the logical building blocks of a software system. It should be noted also that SDBC addresses just the business/software alignment and therefore goes as deep as software specification, in considering the software system-to-be.

Software components, specified in this way, could be further integrated and implemented using the above mentioned component technologies. The elaboration level of an SDBC component should correspond to the UML practices. This means that after producing a use case specification of a software component, the use case model needs further detailization (stakeholders analysis, main success scenario and extensions, triggers, and so on) and elaboration (it should include additional modeling activities aiming at complementing the use case diagram with other essential UML ones, such as the UML sequence diagram, the UML class diagram, and so on). Such elaboration is claimed to be sufficient for providing the further integration and implementation phases with an adequate modeling input. After a number of components are integrated and implemented, a resulting software application would appear. According to Figure 1, the essential goal of this application should be the support of the original business system.

For more information on the SDBC approach, interested readers are referred to the SDBC sources, mentioned before.

SDBC, Brokers and Cultural Heritage Aspect

The SDBC approach is claimed to offer useful advantages concerning the specification of software systems that are intended to support complex business systems in different domains. Among the domains where SDBC has successfully been applied are e-Business and Tele-Work, as already mentioned. In both of them, it has been demonstrated how (via SDBC) a software brokerage system could be specified. As it is well known, software brokers are of great interest currently because of their wide applicability resulting from their actual (brokerage) functionality. Software brokers usually facilitate: 1. the match-making of globally available information, 2. the management of digital archives, 3. the globalization of used data networks. Through software brokers, users could have a quick and effective match-making at low costs.

Because of the relevance of software brokers to some cultural heritage issues (as already mentioned) and also because of the appropriateness of SDBC with regard to the specification of software brokers, it is suggested that SDBS is applied in building cultural heritage sector brokers. They could:

– effectively handle the management and global distribution of metadata as well as of digitized cultural/scientific information;

– be usable on a global scale through the Internet.

Therefore, (SDBC-based) software brokers could stimulate the global availability of cultural/scientific data. Below, the role of SDBC in building such brokers will be illustrated. Because of the limited scope of this paper, the illustration will be incomplete. Only some essential issues will be discussed. Starting from the original user information, the first task would be to understand correctly the business problem to be solved by the software system-to-be and to structure somehow the user information which is usually vague and unstructured. A simplified brokerage model has been drawn (Figure 2) around which the particular case information could be added and analyzed.

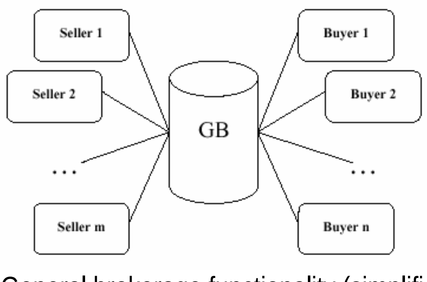


Figure 2. General brokerage functionality (simplified view)

As seen from the Figure, in the general case, there are a number of ‘sellers’ (distributors) of anything and a number of ‘buyers’ interested in it. The ‘General Broker’ (GB) should match appropriate seller and buyer information based on some criteria.

Using this general model, further analysis should follow, considering the particular cultural heritage information (briefly discussed already). In SDBC, the result of such an analysis is reflected in the so called ‘SCI Model’. SCI stands for Structuring Customer's Information. Interested readers could read about the model in the above mentioned SDBC sources.

The general SCI model relevant to this particular situation is depicted (just for illustrative purpose) in Figure 3. The model is incomplete, only some basic issues are there. Among them:

- Units within the General Broker (GB):
  - AU (Acceptance Unit): responsible for accepting and handling submissions from sellers and buyers;
  - FU (Financial Unit): responsible for handling the fee payments done by sellers/buyers as a compensation for the work of the broker;
  - MM (Match-Maker): responsible for performing the match-making concerning the seller and buyer information.
- Actors outside the General Broker:
  - Seller (offering/distributing something, for example, digitized cultural materials);
  - Buyer (being interested in something, for example, in particular digitized cultural materials);
  - Expert: responsible for assisting the broker in some complex situations (in which, for example, real 'human' cultural heritage experience is required);
  - Insurer: responsible for the insurance of relevant issues (for example, insurance against fraud).

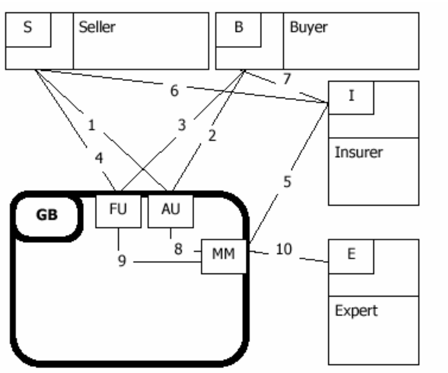
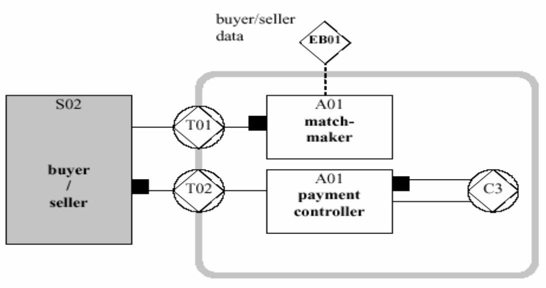


Figure 3: The General Broker: SCI Model

Hence, the GB SCI model facilitates the structuring of the initial case information to be reflected in the identification of a business component. In this particular case, it is suggested that a general business component is identified. The reason relates to the wide usage of software brokers in a number of cases; thus, identifying a general model would allow for re-using it many times. Our (DEMO-based) general business component is depicted in Figure 4. The model is incomplete because of its having just an illustrative purpose.



transaction type	resulting fact type
T01 match-making	F01 match <M> is made
T02 payment	F02 the fee for period <P> by <S/B> is paid

Figure 4: General business component (DEMO-based): The General Broker

As seen from the Figure, two internal GB units are depicted and also two external actors ('seller' and 'buyer': they are modeled as an aggregated actor because of their having the same general attitude towards the broker). As for the internal GB units depicted, they are the match-making unit ('match-maker') and the financial unit ('payment controller'). Two transactions are specified, concerning the actors and relevant GB units: T01 match-making (executed by the match-maker) and T02 payment (executed by seller/buyer). One transaction is specified, taking place within the GB: C3 (it concerns the periodical self-activation of the payment controller in handling all the payments related to a particular period of time). There is also a data bank depicted (EB01), containing the necessary data (concerning both buyers and sellers) that the match-maker should have in order to be able to realize a match-making.

Those readers not familiar with DEMO, are referred to [Dietz, 1999]. Once built, this general business component needs to be extended (as shown on Figure 1) aiming at the identification of a particular business component, in this case: Cultural heritage sector broker (built again with the DEMO notations). However, because of the limited scope of this paper, the transformation from the general business component (The General Broker) to the particular business component (The Cultural heritage sector Broker) is not presented. Information on how such an extension is carried out within SDBC could be found in the mentioned SDBC materials. Hence, a DEMO-based business component (The Cultural heritage sector Broker) should be reflected in a use case software specification model. An example of such a model is depicted in Figure 5. The model is incomplete, containing only some of the use cases characterizing such a broker. The broker is to use a database. It is virtually divided in two parts: one concerning the data submitted by distributors (of digitized cultural heritage materials) and the other one, concerning the data submitted by users. They are represented on the Figure by 'DBD' and 'DBU', respectively ('DB' standing for database; D(U) standing for distributor (user)). On the Figure, it is just illustrated how a software specification model would look like. The use case model will not be explained since it is expected that most of the readers are familiar with UML. As for the DEMO – use case derivation mechanism, information on it could be found in [Shishkov & Dietz, 2003].

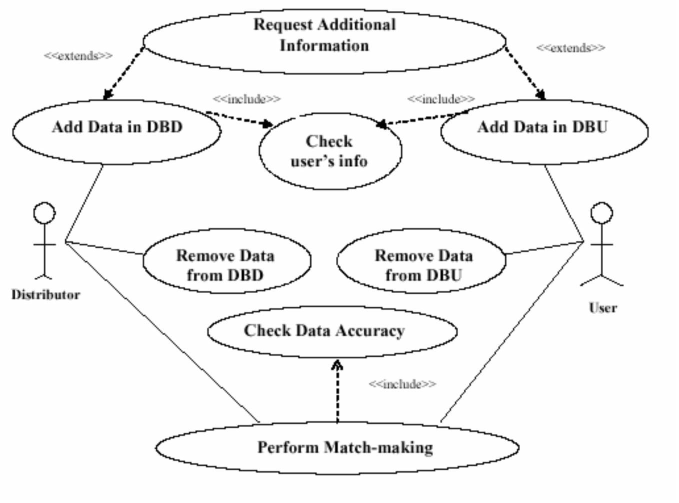


Figure 5. The Cultural Heritage Sector broker: Use case software specification model

Conclusion

Being based on the innovative idea of aligning business process modeling and software specification in a component-based way, SDBC provides an appropriate framework for designing software systems allowing for a sound reflection (in the software model) of the original business requirements and usage of the relevant software standards. Therefore, using SDBC, software specification could be improved (in general) by:

- aligning it to prior business process modeling (based on (re-usable) business components);
- mapping precisely business process models to UML;
- applying powerful modeling techniques and environments, such as DEMO.

Besides its applicability in specifying software brokers in actual domains, such as e-Business and Tele-Work, SDBC proved to be useful in building cultural heritage sector brokers. The great complexity of the original business input makes this situation inadequately solvable by using the existing popular software design methods. The basic reason is that neither of them allows for a sound and complete business/software alignment. Hence, this would result in considering partially the initial business input and relying (as usually) on suitable interpretations from the software designers. However, in such a specific case this could not replace a necessary adequate consideration of the original business issues seen from the specific cultural heritage perspective. Thus, a business/software alignment is crucial and this motivates the selection of SDBC as a suitable approach with regard to the particularly considered (brokerage) problem. Brokers built using SDBC would:

- be rigorously rooted in the (original) business process model;
- be easily re-usable in other cultural heritage projects because of their being built based on components;
- be fully consistent with the current software design standards.

By supporting the specification of cultural heritage sector brokers, SDBC proves to be useful for the cultural heritage domain because such brokers could facilitate the management and global distribution of digitized cultural (and scientific) information.

Bibliography

[Dietz, 1999] J.L.G. Dietz. Understanding and Modeling Business processes with DEMO. In the Proceedings of the International Conference on Conceptual Modeling (ER'99), Paris, France, November, 1999.

[Jacobson et al, 1992] I. Jacobson, M. Christenson, P. Jonsson, G. Overgaard. Object-Oriented Software Engineering: A Use Case Driven Approach, Addison-Wesley, 1992.

[OMG, 2004] http://www.uml.org

[Shishkov & Dietz, 2004-1] B.Shishkov and J.L.G.Dietz. Aligning Business process modeling and Software specification in a Component-based way, the Advantages of SDBC. In the Proceedings of the 6 th International Conference on Enterprise Information Systems (ICEIS'04), Porto, Portugal, April 14-17, 2004.

[Shishkov & Dietz, 2004-2] B.Shishkov and J.L.G.Dietz. Design of Software applications using Generic business components. In the Proceedings of the 37 th Hawaii International Conference on System Sciences (HICSS'04), Big Island, Hawaii, USA, January 5-8, 2004.

[Shishkov & Dietz, 2003] B.Shishkov and J.L.G.Dietz. Deriving Use cases from Business processes, the Advantages of DEMO. In: Enterprise Information Systems V. Ed. O. Camp, J.B.L. Filipe, S. Hammoudi, and M. Piattini. Kluwer Academic Publishers, Dordrecht/Boston/London, 2004.

[Shishkov, 2002] B. Shishkov. Business Engineering Building Blocks. In the Proceedings of the 9 th Doctoral Consortium on Advanced Information Systems Engineering (CAiSE'02), Toronto, Ontario, Canada, May 27-28, 2002.

[XML, 2004] http://www.xml.org

Author Information

Boris Shishkov – Department of Software Technology, Faculty of Electrical Engineering, Mathematics & Computer Science, Delft University of Technology; Mekelweg 4, 2628 CD Delft, The Netherlands;
e-mail: b.b.shishkov@ewi.tudelft.nl