On the Intelligent and Secure Scheduling of Web Services in Service-oriented Architectures – SOAs

Katalin Szenes

John von Neumann Faculty of Informatics
Budapest Tech Polytechnical Institution
Bécsi út 96/B, H-1034 Budapest, Hungary
szenes.katalin@nik.bmf.hu

Abstract: The ready availability of the internet lead to the development of such business applications that the users can reach through web sites. In order to keep track with the rapidly changing business requirements some of these applications – the SOAs – are built of loosely coupled and thus individually modifiable components – so-called web services – that might keep their connections with each other also on the internet. These components communicate each other mostly by the means of XML-based languages often through the internet. This increase the vulnerability of sensitive corporate and personal data. The paper proposes to increase the information security by the means of such a scheduling methodology that is based on some former results of an artificial intelligency-based scheduling of parallel and concurrent process systems.

Keywords: services oriented architectures, SOA, web service, intelligent orchestration of web services, artificial intelligence, parallel and concurrent process systems, time critical applications, information security

1 Service-oriented Architectures

The concept of the service oriented architectures – SOAs – comes from the middle nineties when the improvement of the application development tools permitted different ways for sharing the business logic between the different applications. Another issue was to permit a kind of multi-threaded execution of these different aplications even if they operated on the same database. It is possible that the boom of client-server computing is also due partly to these ideas.

A bit extended requirement set that a SOA has to comply with might be formulated as the availability of such a *unified* surface that the user can reach at one point and that *contains* such *services* that

- are developed by different companies, suppliers,
- using different methodologies,
- running on various hardware-, operating system- and database platforms.

With the improvement of the different portal development tools the next requirement is that this set of different, but from the users' point of view unified services should be available simply by clicking onto a homepage. This access mode resulted in the choice of XML as the communication language for these different services. Based on XML different languages were created WSDL (Web Service definition Language) and the like. All of them are dependent on their XML base.

The idea is to support the business systems developers in such a way that they can design a business process without taking any care of implementation problems. Just as in the good old days of Artificial Intelligence when the goal was to support the user ’in the how’ to such an extent that he/she has to deal only with the ’what’ of the problem, that is with the specification level only.

The experts who are well-versed in the business processes design the processes and the developers take these business process definitions and link the steps of the processes to the appropriate services of the SOA.

Another ambitious goal of the service oriented architectures is to wrap the old legacy systems to the other services in such a way that the user can call them using the single entry point to the architecture. Of course there are problems with this joining the old systems to the new ones. Usually nobody would be able to modify them as no information about their inside is available anymore. However, if their communication can be grabbed somehow and forwarded to the common cooperation point then there is hope that these old systems can somehow be integrated to the new architecture at least from the viewpoint of a commonly used entry point.

Getting back to the possibility of calling the architecture by a simple click that is from the corporate intranet or from a remote point that is connected to the corporate network only by the internet besides an enhanced support of the business sytems planning this is the promise that makes SOA desirable. Nowadays when the employees tend to work practically anywhere outside the premises of the company the availability of an application system without a wired connection is a very important point.

2 Information Security Threats Attached to the Different Levels of the Architecture

Az important source of danger is the user authentication, especially throughout such corporate networks that spread through country borders. The service oriented architectures are to be used from any outside point, from anywhere the user might click onto the web portal that is entry point of the SOA application. This entails the use of a kind of syndicated identity that is valid in different companies.

For this the so-called federated identity management is the solution that supports the checking of the identity through different corporates by additional methods besides the passwords. The tools stating themselves to be compliant to the accepted standards are checked by an alliance of the suppliers [13].

The base level of a service oriented arhitecture comprise the web services themselves that are programmed to serve mostly only one function, one menu point of the user menu of a whole service oriented architecture. These services communicate partly through the internet by the means of the XML-based messages mentioned above. The internet as a communcations media is dangerous, hackers, crackers and the like try to break and read the messages either for fun or, what is worse, as these are the professional threats, for money.

This communication opens up the inner structure of the applications towards this untrustworthy outside world.

Because of the XML this communication is threatened by every vulnerability of the HTTP protocol and the more so as the physical format of the web service comm. is usually the SOAP protocol which is a physical extension of HTTP actually by adding some bytes. (SOAP earlier abbreviated the term Simple Object Access Protocol but from 2003 on the XML Protocol Working Group of W3C – World Wide Web Consortium – decided to delete this reference.) Among the SOAP dangers are the eavesdropping and faking of messages, or message redirection just as well.

There are lots of other threats that originate partly from the strong reliance of the communication of the processes that takes place mostly the internet and the vulnerability of the infrastructure implementing the web page onto which the user clicks requires lots of defensive considerations but to enumerate all this and their possible defense is outside of our present subject but is detailed in [13].

What is related to it is the issues concerning programming - programming the web services, their communication and their cooperation with each other. To solve at least some of the problems arising here is the artificially intelligent-based scheduler suggested.

A frequently used XML language for the description of the communication and interfaces of the web services is WSDL (Web Service Definition Language). It

describes how to call a web service and from where it is to be called. The programming errors committed in this language are so common that if anybody asks the Google to present tools for WSDL scanning that is tools that are able to look for WSDL vulnerabilities then he gets hundreds of hits within a tenth of a second. This shows that the programming of and around web services should be disciplined. The object-oriented programming is a promising candidate for writing programs running in an exposed environment as this discipline, supporting the creation of thoroughly planned systems, helps avoiding the vulnerabilites caused by programming mistakes [14]. In building the proposed AI-based tool object-orientation was also exploited [8].

The information security problems of the communication of the web services are widely discussed in the XML Protocol Working Group of the W3C (World Wide Web Consortium) and the OASIS WSS (the Web Service Security Technical Committee of the Organization for the Advancement of Structured Information Standards) but the way of distributing the tasks required to be performed by the end-user and the management of the interaction of the components are issues that seem to be outside the focus of interest. What we propose here is an AI-based implementation of the orchestration of the web services.

3 The Intelligent Orchestration of Web Services

3.1 Semaphores Revisited – AI Tools Come Back

The problem of using the same data or applications by different processes is not a novel one in computer technics. The implementation of the handling necessary read / write permissions was usually based on a kind of semaphore that works just like at the railway. It is set to ‘set’ if the data is already in use – there might be processes that read it simultaneously while one is reading it – this depends on the characteristics of the data. The semaphore is set to ‘free’ if the data is accessible. Just now this idea became fashionable again for the SOAs [5].

Unfortunately, the capabilities of a semaphore are not enough for time critical applications. For example, the electronic card systems of banks require that the time elapsed from the point of time that the ATM software forwards the request on the customer's bank account to his own card system and then this asks about the balance the customer's bank has to be very short otherwise the ATM refuses the transaction.

Thus for the processes belonging to a banking application have to ‘know’ about time and have to be able to finish certain tasks within a given time interval.

These time and resource handling problems are very similar to the production scheduling problems we have solved by the means of artificial intelligence-based process modelling systems, first by a PROLOG-based one [6] then by a new one, PCUBE, that was more effective then T-PROLOG as first FORTH then C++ was its base code [7]. This suggests that the scheduler of system PCUBE where the name stands for the *P*lanning of *P*arallel and concurrent *P*rocess systems might be tried to manage the interaction and the scheduling of the web services of a service oriented architecture.

3.2 The Scheduling Requirements of the SOA Applications

The basic requirements to be fulfilled by the secure information systems are:

- the availability of the systems and their data,
- the confidentiality of their functions and data,
- the integrity of their data.

Availability means that the users can reach the services of the systems during a predictable percent of the system uptime and the downtime doesn't exceed a given percent of the whole working time of the users.

The confidentiality refers to the possibility of accessing the functions of a system or that of the data. An application is secure if these access rights reflect exactly the role of the users in the company. Everybody is able to access those data and functions that are needed to the work but only those.

Data integrity means that throughout the processing the data remain the same as they had entered the system. E.g. the family name of a partner will not be changed by a mistake for the name of the street where he lives.

Confidentiality and integrity can be ensured by the means of information security methods [11]. The level of their fulfillment is to be checked by periodic risk management cycles [10].

The proposed method for the orchestration of the web services is such a scheduling of the services and planning their joint execution that avoids the deadlocks of the system of web services by the means of a preliminary planning of their execution. This supports the *availability* of the service oriented architecture consisting of these web services by helping them to fulfill the specifications of the application and contributes to the *confidentiality* of their functions and data by the means of decreasing the probability of the occurence of program errors that lead to systems vulnerability by the means of offering the transparency of the cooperation of the web services as concurrent processes.

To create the environment in which this method can be employed to the service oriented architecture such a layer, the so-called enwrapping layer was defined, that

'packs together' the web services into one system of concurrent processes [13]. The resources for which these processes compete are the data base records that they 'have to' handle in order to fulfill that particular user requirement for which they had been programmed. The main component of this layer is a scheduler that provides for the orchestration of the web services.

Clicking onto the entry of a SOA-based application is actually clicking onto a link on a web page. This link gives the control straight to this enwrapping layer.

The first tasks are the identification and the authentication of the user. If this is successful then, according to the user's position and role in the corporate hierarchy he gets authorization to use certain web services of the SOA. This point arises numerous information security problems that can be solved or at least mitigated by the means of software, hardware, network security and organizational methods [12, 13].

Further important issues are

- the *choice* of the appropriate web services that execute those tasks that serve the menu points the user chose invoking the service oriented application,
- the determination of an *optimal order* of assigning these web services to the user's disposal,
- the assigning of the *resources* - applications or data to the disposal of the web services.

The goal of the intelligent scheduler that is introduced here is to manage this choice of the needed web services and controlling their steps of execution in such a way that the problem is solved, and, if possible, in an optimal way.

All these have to be done allocating the necessary resources to the tasks in case of need. The proposed method handles the web services as *parallel processes.* The *resources* for which they compete are the elements of the corporate databases. This is the concurent system of processes that has to be scheduled. To build such a scheduler that the enwrapping program has to contain some methods belonging to toolkits of the artificial intelligence are proposed.

3.3 The Intelligent Scheduler

3.3.1 PCUBE, the AI-based Planning, Simulation and Modelling Tool for Parallel and Concurrent Process Systems

The long history of the method began in 1980 when we extended the Artificial Intelligence (AI)-based PROLOG with simulation and resource manipulation. The model of the implemented instructions were taken from SIMULA 67 and

Concurrent Pascal. The result we named as T-PROLOG (where T stands for time) and it ran on the ICL 1900 mainframes [1]. T-PROLOG had been written in PROLOG thus it was big and slow. Even in this way the language was a good illustration for the use of AI in solving real-life parallel and concurrent problems [6].

When the Z80-based and other 64 kByte microcomputers became available the implementation had to be changed because of the size and speed facilities of these microprocessors. The special way of PROLOG program execution arose the idea that the simulation and time/resource maintenance instructions can be described by the means of tree - traversing.

The description of the process sytems is a set of clauses, it is similar to a T-PROLOG program. A clause is either a simple statement or a deductive rule. The clauses give the conditions of the execution of the processes, they are the description of the environment, the knowledge base of PCUBE, in which the processes 'have to achieve' their given goals.

Below this surface such a system system was developed that has a completely different architecture from those of PROLOG or T-PROLOG. However, as the model of the processes are tree traversings, the finding of a deadlock free solution of the set of process goals is still possible.

This way the advantages of the AI problem - solving could be preserved. It means that *if there is a solution* of the given problem under the given conditions the system finds it. The deadlock situations are resolved by the means of the so-called backtracking that is known from PROLOG: the system steps back to such a previous point of deduction where an other alternative could have been chosen and tries to use this one.

The tree - traversing was implemented in a list processing language developed for this purpose. Otherwise this is a general list processing tool that supports both the forward and backward list traversing. The task of microcomputer implementation was reduced this way to the realization of the list processing language.

Thus the levels PCUBE consist of are the tree - traversing, the list processing, and the base level coding tool.

For base level tool first programming language and system FORTH was chosen [9]. At that time we had to compromise with a memory of 64 kByte and in FORTH very economic programs could be written. FORTH supported the programmer in defining his / her own instruction set which – in this case – consisted of the list processing instructions.

With the development of the microprocessor FORTH went out of fashion and the base elevel of PCUBE could easily be changed to C++ [4].

3.3.2 The Key to the Implementation of PCUBE

The base of PCUBE is the one-to-one correspondence between the process and the backtracking traversing of a tree. The nodes of this tree correspond to the conditions that the process ‘can use’ to reach its given goal.

The trees come from the way that the middle level of PCUBE, the list processing language interprets the clauses. It is an inference machine, just as PROLOG had been. Trying to reach a process goal the system takes the clauses one by one and these clauses are the nodes of the trees that correspond to the processes. This is how the set of clauses ‘describe’ the system of processes. This way the process system to be handled by PCUBE correspond to the traversing of the non-distinct union of the trees corresponding to the individual processes. Due to the tree traversing PCUBE has a built-in backtracking feature. Thus principally if there is a solution of the given problem then PCUBE will find it. If it reaches a deadlock situation it backtracks and tries to reach the goals in an other way if possible.

For the resources, the communication, and the time handling instructions special clauses are defined, such as

time handling:

AFTER (point_of_time)

AT (point_of_time)

BEFORE (point_of_time)

HOLD (time_interval)

TIME (what_is_the_time)

TIME (point_of_time)

resources:

RESOURCE (resource_name , number_of_the_available_ones_of_this_type)

RELEASE (resource_name)

TAKE (resource_name)

direct communication - messages:

SEND (message)

WAIT (message)

Those self-explanatory arguments that contain the term ‘time’ refer to the system time of PCUBE. This is a common ‘variabl’ to every process in a given user's process system and is maintained by the scheduler. The scheduler is necessary to handle all these facilities describing parallelism, concurrency and communication. The processes execute actually in a quasi-parallel way when the system of

processes is simulated to find a common solution for the goals of every process comprising the process system.

For the service oriented architectures this scheduler can handle not only the locking of records but there is hope that it can model the cooperating of the web services in such a way that every one of them reaches its goal within a given time interval.

Conclusions

The *set* of web services comprising a service oriented architecture can be handled as a *system* of parallel and concurrent processes. The resources for which the services compete are the data and the applications of the institutions. Availability, one of the most important information security requirements, can be supported by the means of scheduling the web services using PCUBE, the modelling, simulation and *P*lanning tool for *P*arallel and consurrent *P*rocess systems. Its scheduling algorithm is built using a tree traversing method derived from the PROLOG way of program execution.

Acknowledgement

The author is greatly indebted to Professor Tibor Vámos for his more than 30 years support of the research work with discussions and encouragement.

References

[1] I. Futó, J. Szeredi, K. Szenes: A Modelling Tool Based on Mathematical logic - T-PROLOG, in Journal Acta Cybernetica, Tom. 5, Fasc. 3, 1981, Szeged, (JATE) Hungary, pp. 363-375

[2] IBM - Chris Nelson, Jeff Miller, Willy Farrell, Rachel Reinitz, Kyle Brown: Implementing a Service-oriented Architecture Version 1.0, February 8, 2005, Copyright © 2005 IBM Corporation

[3] R. C. Norris: Virtual Private Networking: Confidentiality on Public Networks, Information Systems Control Journal, Vol. 3, 2001, pp. 23-26, editor: Information Systems Audit and Control Association, Rolling Meadows, Illinois, USA

[4] Pálossy László, Tempfli László: A PCUBE szakértői rendszer egyes elemeinek implementációja C nyelven IBM PC-re, Szakdolgozat, Eötvös Loránd Tudományegyetem, Budapest, 1993

[5] David Perelman-Hal: AJAX and Record Locking, Dr. Dobb's Journal, CMP Media LLC., 600 Harrison Street, San Francisco, CA 94107, USA pp. 45-51

[6] Katalin Szenes: An Application of a Parallel Systems Planning Language in Decision Support - Production Scheduling, Proc. of the IFIP W.G. 5.7 Working Conf. – 'APMS (Advances in Production Management Systems)

82', Bordeaux, France, Aug. 24-27, 1982, ed.: G. Doumeingts & W. A. Carter, North Holland, l984, pp. 24l-249

[7] Katalin Szenes: Planning the Activity Schedule of Process Systems by the Means of an AI-based System, Proc. of the 27th International MATADOR Conf., Apr. 20-21, 1988, Manchester, ed.: B. J. Davies, UMIST, MACMILLAN Education Ltd., 1988, pp. 139-144

[8] K. Szenes, P. Forró: Implementing the Base Level of a Process Maintenance System in FORTH, Proc. of the 6th Symp. on Microcomputer and Microprocessor Applications, Budapest, Hungary, Oct. 17-19, 1989, ed.: Scientific Society for Telecommunication, Budapest, Hungary, pp. 65-74

[9] K. Szenes, P. Forró: System PCUBE Short Reference Manual, PCUBE Version 1.0 (IBM PC, PC DOS), 1989, Version 1.1, February 1990

[10] Katalin Szenes: Building a Corporate Risk Management Methodology and Practice, Proceedings EuroCACS 2002 - Conference for IS Audit, Control and Security, March 24-27, 2002, Budapest, Hungary

[11] Katalin Szenes: Prevention of Fraud in Financial Institutions and in Other Corporations in Hungary, panel, ISSE (Independent European ICT Security Conference and Exhibition), Budapest, Hungary, September 27-29, 2005

[12] Szenes Katalin: Az ISACA auditálási alapelvei, és a COBIT® módszertan bemutatása, Az informatikai biztonság kézikönyve, 2006, Verlag Dashöfer Szakkiadó Kft. & T. Bt., 1062 Budapest, Andrássy út 126., pp. 6.11.1-6.11.83

[13] Szenes Katalin: A szolgáltatás-orientált architektúrák biztonsági kérdései, megjelenés alatt, Az informatikai biztonság kézikönyve, 2006., Verlag Dashöfer Szakkiadó Kft. & T. Bt., 1062 Budapest, Andrássy út 126.

[14] J. Tick: Software User Interface Modelling with UML Support, in Proceedings of the IEEE 3rd International Conference on Computational Cybernetics, ICCC 2005, Hotel Le Victoria, Mauritius, April 13-16, 2005, pp. 325-328