Mihai-Constantin AVORNICULUI
Faculty of Economics and Business Administration,
Babeş–Bolyai University of Cluj-Napoca

CONSIDERATIONS ON OBJECTIVE METHODS FOR DEVELOPING APPLIED EVENT EXTRACTION SYSTEMS

Theoretical articles

Keywords
Event Extraction Systems
Developing Methods
System Implementation
UML

JEL Classification
M15

Abstract

There are several similarities between designing classic systems and systems for event extraction. Similarities derive from the use of common methodology in designing and developing these systems. The system is intended to be a helpful instrument when looking for pieces of informationon different web pages or when in case of a special type of event,information needs to be centralized. Use-case diagrams need to be designed through both methods. The relationships between classes have to be identified and described. Differences induced by the design process are due to the special character of a web based data exchange.

Introduction

Nowadays the amount of information on the Internet reaches high proportions. In finding specific information on the Internet some general instruments, like search engines, have become popular. They automatically run through all the existent web pages, in order to update their databases containing the latest information on the Internet. In most of the cases, the search is done based on a number of strings stored in the database of the search engine. The result of such searches is a large amount of links to different web pages.

To systematize the searching process and to obtain a result in a precise form another useful stage is used, in which the information returned by the search engine is processed and the answer is generated in a more organized form (Han & Kamber, M, 2006, Lin et. Al. 2008).

The centralization of a specific type of event is useful for the development of newsservices. These services must offer real-time updated information about a specific type of event.

Take sport events for example. A user whichis provided with this type of services, might want to find out what sport events will take place in a certain geographical region (a city, a country, etc.) in a well-defined period of time (in that very moment, a day before, or the next week, etc.). All this information has to be obtained from a centralized information source. This way, useful data can be obtained about a specific event from various sources. These sources (the websites from where the information was taken) can complete each other concerning the content of information referring to the given event.

Special aspects that have to be taken into consideration when designing an event extraction system compared to the design of a classic system

In case of a classic system, data and information exchange takes place in the same informational system, due to the fact that data and information are not exceeding the organizational boundaries. Meanwhile in the case of an event extraction, the data system will definitely exceed the system's boundaries (Avornicului, 2010).

The components among which the information stream takes place in a system belong to the same entity and are designed by the same team of developers. In case of an event extraction system these components belong to different organizations and probably have been designed by several teams.

The technology of the developing of event extraction systems presents a high level of homogeneity and ensures compatibility and interoperability. It is less probable that the technology implied in the case of a classic system is natively homogenous and natively compatible and interoperable (Markov & Larose, 2007).

In the case of classic systems redesigning comes only after a certain period of time, after it can no longer face the demands and realities of the domain for which it was made. The systems for event extraction are redesigned as a process. An event extraction system is very dynamic; it evolves in the same time with the web (Lin, 2007).

In case of a classic system the consumers of information are familiar with each other. In case of an event extraction system they are not always known and even when they are known they present a high rate of dynamism.

Considering the opportunity of using UML

Using UML for developing an event extraction system the following aspects are to be taken into account (Avornicului, 2009):

- UML is a powerful instrument for modelling aspects of behaviour. This

way, classes contain both data and the associated processes.

- UML offers a unified vision above grouping different components in the form of packages and also the physical aspect of these packages.
- The UML collaboration diagram is very important for understanding how different objects from a system interact with each other and what messages they exchange.
- Event extraction systems are complex and present a dynamic evolution of a higher level than other types of IT systems. It is necessary to use an instrument of analysing and designing that allows the future extension of the system.
- UML includes mechanisms that allow the easy reuse of the systems components (capsulation, heritage and polymorphism).
- UML allows an integrating point of view for both the data and the process.
- Analysing and designing UML offers a continuous development of the model.
- Development methods that rely on UML are iterative methods guided by use-case diagrams. This strategy permits the elimination of programming errors before the application gets to the client. These facts point to a series of advantages such as: respecting deadlines, reducing both the costs of exploit and maintenance, and the probability that the system does not meet all the client's expectations.
- UML, unlike other methods, allows a continuous transition in both direction between build-design and design-build phases.
- The user is in contact with the team during the whole period of development. In case of other methods, it was only present at the beginning and at the end of the process.
- UML is an expressive language that allows the designer to express shades of the model that would be hard to express in other cases.
- It is an extendible language that allows the continuous adaptation to demands of the domains that it makes models for.
- Despite its flexibility, UML is a precise language with well-defined forms
- UML is easy to use.
- Event extraction systems are systems that work intensely with asynchronous events. UML is a language that provides high level instruments that help at the development of these systems.
- Once a use-case is finalised, the respective subsystem can be launched due to object-oriented programming paradigm.
- Event extraction systems are reactive systems that need detailed modelling of events. From this point of view, I consider that the models proposed by UML for events and messages exchanged by objects are more powerful instruments than other languages can offer.

Aspects of the RAD methods applicability

The RAD method implies developing a solution in a 60–90 day interval. In case we choose a method like this, we have to consider the fact that the development of a complex systemcan be finished only with certain compromises (Zaki, 2000).

The method is based on the following set of assumptions (Sommerville, 2010):

- In many cases, an 80%-functional application can be developed in only 20% of the time allocated for itsdevelopment.
- There are situations when the acceptance of a system is basedon the completion of a minimal set of requirements. The rest of the functionalities get developed after the acceptance.

From the perspective of the previous assumptions, event extraction systems can be accepted with so-called partial functionality. For instance,a basic requirement would be the function of interspecific conversion. The flux engine is a lower-priority requirement; therefore it may be developed later. Reflecting on the first assumption, an event extraction system cannot be interpreted as those which functionality can be developed to 80% only by using 20% of the time available,due to the following (Han & Kamber, 2001):

- We can never state that an event extraction system is “ready”, since it is under continuous development, specifications and message standards being continuously evolved.
- Developing conversion engines is a time-consuming activity due to the complexity of parsing.

Problems raised by the RAD method are:

- By applying classic methods, the client has to wait a lot until the final product can be seen.
- The development when using classic methods can last so long, that in the meantime, the client’s business processes might change radically.
- The product delivery when using classic methods is done by the “all or nothing” principle.

RAD method proposes 5 stages (Sommerville, 2010):

- *Initializing the process* – in this stage the working environment is decided, a general plan of the team is elaborated. Aspects of functionality are analysed and users are identified.
- *Stabilising the users demands* – in this stage everyone from the first stage is involved in identifying the user’s exact demands. Often in this stage the user’s demands are specified under the UML use-cases.
- *Designing the system* – it proposes elaborating a beta version of the system in specific terms of modelling. Because of iterative development, the possibility of a later extension of the system has to be ensured. For this, object oriented methodology is needed.
- *The actual development*– implies writing the code. The development team has to use the same programming style and the same patterns. Homogeneity can be realized by the use of CASE instruments. The stage is iterative and it has in permanent attention the delivering of functional versions. Potential extensions of the application can be made in this stage.
- *Finalizing the project* – proposes shutting down the iteration chain and delivering a final form of the application to the user.

Characteristics of RAD methodology are (Avornicului, 2010):

- Teams used in RAD are mixed teams. Among their components are polyvalent persons: developers, analysts, architects. Beside themthere are the final users, as well as other persons that have the necessary knowledge for developing the respective system. An event extraction system needs a similar structure. An event extraction system often has to create contact between pages with content of different domains. In this case a mixed team would only contribute to the success of the system. The complexity of these kinds of systems makes the use of polyvalent developers necessary.
- The method uses instruments that allow: visual development,prototypes creating, multiple languages, CASE instruments. These instruments are specific to developing modern systems like event extraction systems.
- Functionalities are added iteratively as they are needed. The event extraction system can be operational in the first stage with a few core functionalities to which later new functions can be added.
- IT uses iterative development. Every iteration ends with a working version of the system. With the exception of the first iteration that might take longer, all

iterations take between 1 and 21 days. Iteration starts with a common period of development in which there is a close relationship between the developers and the users. Following this stage there are more iterations that take the system closer to the users vision.

- Gives credit to the users demands. The method starts from the idea that the users know clearly which functionalities are required. In case of an event extraction system they try not to involve the user in the operation of data exchange, unless it is necessary.
- Introduces a temporal dimension to demands that insures the evolution towards the final version of the application.
- Accentuates programming. Unlike classic methods, RAD puts the accent on intensive programming (still not as much as the XP method).

Aspects of the RUP methods applicability

RUP represents a methodology that allows the developing team a disciplined approach in allocating tasks and responsibilities, within time and budget limits.

RUP is a product-oriented process and it is sustained by several instruments provided by Rational Corp. RUP increases productivity of the team by giving access to a base of information with guidelines, patterns and instruments that allow the monitoring of all activities that appear during the process of development. RUP is a special concept made for optimizing the use of UML, it is a configurable process and it is supported by the biggest part of instruments that automates the process. The method is both for small and large teams of developers (Sommerville, 2010).

RUP is an iterative method. Every iteration has one or more objectives. In RUP the objective is to produce working software that gives value to the client. Iterations are limited by deadlines. This involves that each facility has to be carried out in a certain period of time.

According to Kruchten (2001), RUP has the following characteristics (Kruchten, 1998):

- *Interactive developing of a software product* – proposes developing consecutive iterations in short increments. This insures a real-time detection of risks.
- *Processing demands* – it is a continuous process of identification of demands of a system that evolves within time.
- Processing these demands requires a disciplined manner of evaluating, associating and monitoring priorities. Communication has better chances when it has a well-defined set of demands at base.
- *Uses architecture based on components* – it is more flexible,it is easier to extend the respective application. Components can be redesigned or extended without compromising the evolution of the whole system.
- *Uses visual instruments of modelling contributing to the understanding of extremely complicated systems*– using UML models the complexity of a system can be effectively processed among more developers.
- *Permanently verifies the quality of the produced software*– this is done in each iteration. Because of this, errors are discovered in time and revising costs are reduced.
- *Controls changes brought to the developed software*– dealing with these changes represents the key to success in development.In case one of the team members brings a change to the system, everyone whose work is affected have to be warned.

Controlling these changes is an additional opportunity of this method (Chen & Wei, 2002). From analysing RUP methods characteristics the following conclusions are brought having in scope

event extraction systems (Avornicului, 2009):

- RUP is a method that allows developing systems with a flexible and extendible architecture. Event extraction systems are these kinds of systems, therefore they comply with these demands.
- RUP puts accent on dealing with aspects of potential risk in time. This characteristic is a plus for every system.
- It does not imply a fix set of requirements in the initial stage; in this meaning they can be refined as the project evolves. From the point of view of an event extraction system tasks cannot be specified in the first stages, so this characteristic is in favour of event extraction systems.
- RUP puts accent on the final product and on the conformity of this with the final user's demands. This is clearly an advantage for EES.
- RUP leaves the evolution of the system entirely on the users will. From the point of view of developing an event extraction system, this character can turn to an important disadvantage. It is possible that the user does not have any knowledge of EDI or XML message formats so he could jeopardise the flexibility of the system.
- RUP shows the advantages offered by UML. For a lot of IT systems this represents added value.
- RUP makes possible to control the quality of the developed system. The systems quality is in close relationship with its reliability. The better quality of the code means a more reliable system. This characteristic brings an advantage to developing an EES;
- The time in which a system is developed using RUP is shorter than in case of other methods, thereforethis can also be considered an advantage.
- When RUP is applied, it becomes a repetitive and predictable process for the developing team. This leads to a high efficiency in case of developing large and reliable software.

It is considered that RUP represents a method which can be used with success at the developing of an EESif some disadvantages of the method are avoided. Among these is the fact that RUP does not contribute directly to the development of implementing instructions, since it is a perspective method compared to the more agile XP. The opponents of RUP method ask questions like: what to do in the initial and transitional stages? While RUP pleads for the specialization of people involved in the development, agile methods plead for generalists that can meet as many requirements as possible from different stages and phases. RUP is not the follower of aggressive rebuilding as far as agile methods have this as their primary characteristic. Due to its characteristics RUP is adapted to developing EES for provisioning chains and workflows.

Aspects of the XP methods applicability

Extreme Programming (XP) has evolved due to problems caused by long cycles of developing traditional methods. Initially the method started as a way of obtaining resultsfast by using tools that proved their success along the time (Cutting et al., 1992). After a number of successful attempts in practice, the XP method was put in theory based on principles and practices. It has to be mentioned that practices presented in the XP method are not new, but have been taken from existing methods that already proved their effectiveness.

The "extreme" term suggests that these principles are taken to an extreme practise (Raffai, 2005, Sommerville, 2010).

The XP method promotes 4 values to the rank of principles:

- *Communication* – represents essential criteria of a project's success. Problems often appear at the development of a system when communication is missing among the teams' members.
- *Simplicity* – it is wanted but it cannot be expected from beginners. To be

simple,you have to have the experience of some other projects. Simple solutions do not appear at first try, but after lots of different solutions that lead to a simpler view of the problem. For something simple it is hard to generate errors or even if it does it is easy to rectify. A simple solution does not have to be confused with a simplistic one. A simple solution means the maximum compromise between complexity and flexibility. A solution becomes simple by fine-tuning. If a similar approach appearsat the beginning, it means that the person who proposes has got experience.

- *Feedback* – is the team's fuel. It motivates the members professionally to go further and to resolve problems in a way that the client would be more and more satisfied. Feedback can be conceived from the temporal perspective, the client's perspective or the programmer'sperspective. From the temporal perspective we can have feedback in minutes, days in the development phase or weeks, and months in the stage of implementation and maintenance. From the client's perspective, feedback represents his ability to modify the way the system deals with certain problems. From the programmers' perspective, feedback is the only way to see if his work met the client's expectations. Feedback evolves simultaneously with the three values: communication, simplicity and courage.
- *Courage*– comes with your will and engagement to your client, for dealing with problems faster than others.

Another principle of the XP method is quality software development. This principle implies that members of a team wish to develop software that makes their job easy at the maintenance stage (Jing & McKeown, 1999).

XP methods bring into discussion different roles that participants can have when developing (Sommerville, 2010):

- *Programmers* – write source code and test the applications. The key of XP's success stands in making the programmers communicate and collaborate with other members of the team.
- *Client* – the person or organization that will use the certain system. He decides on the requirements, whether they are met and their priorities. He also describes the functions and brings test scenarios to verify/approve them.
- *Tester* – the person who verifies that the test results comply with the client's former demands. They periodically run tests on the system to make sure that it has not been altered after interventions.
- *Monitor* – the person who evaluates the progress made by the team and whether it can fit in time and budget. He also gives feedback.
- *Coach* – is responsible for the whole evolution of the project. The coach becomes the coordinating factor of the entire process. He has to have the neededexperience to offer support to the whole team, as well as solutions to more complicated problems. Just like a project manager.
- *Consultant* – is the external person that delivers technical information about the domain in which the team works. Often the consultant is chosen by the client and is a person with experience in the domain.
- *Manager* – the person who makes decisions. He evaluates the stages of the project and based on this he makes decisions. This should lead to solving the difficulties and deficiencies.

Analysing XP's applicability in the development of an EES, the method promotes the following practices:

- *Planning*– involves a tight interaction between the client and the programmers. Programmers estimate the effort necessary and the client decides on the purpose and the frequency of launching prototypes. This practice advantages smaller event

extraction systems like electronic market or provisioning. It disadvantages types like administrating the flow of activities.

- *Delivering versions in short time intervals*– a system is delivered quicklyand then updated to better and better versions. This practice is very popular among software companies; a proof for this is the number of patches launched to resolve dysfunctions. When dealing with an event extraction system,a strategy like this one can be applied for all presented architectures.
- *Developing systems is madebased on a single description that explains how the system works*– a practice like this is hard to apply on event extraction systems. Only one description is feasible in the case of solutions specially developed for a certain client having a higher level of personalization or prototypes with a demonstrative role; but not a general solution that is about to provide a flexible instrument which would facilitate the modeling of interorganizational data exchange. This practice advantages smaller event extraction systems like electronic market or provisioning but disadvantages types like administrating the flow of activities.
- *The system should be designed to be simple* – each time complexity is discovered, it needs to be eliminated. From the point of view of event extraction systems, complexities are hard to be eliminated; in most of the cases they are only isolated from the architecture.
- *Continuous testing of the product under development* **–** representsan advantage for every system, still some aspects can make this approach difficult. For instance, in case of event extraction systems the lack of communication between two developers due to undefined message structure makes both coding and testing hardly achievable.
- *Restructuring the system every time it is updated, simplified or made more flexible if this does not change its behaviour* – can be used only at small or medium-sized systems. In case of event extraction systems it should not be an issue if the architecture is proper.
- *Programming in teams of two* – it is considered productive unless it is used for educational purposes. Otherwise it can be a waste of resources. Responsibility of coding could not be assigned to only one member; the lack of responsibility would lead to the lack of motivation.
- *Anyone from the team can change the code*– this might be a shortcut to chaos, even easier than the lack of communication. Even if communication is working, in case of large systems these changes can lead to an unexpected behaviour of the system. It is extremely unpleasant to realise that a component that did its job earlier, does something else later.
- *Continuously adding functions to the initial prototype* – a natural process, beneficial for every system that hascomponent-oriented architecture.
- *It is recommended that a working week should not exceed 40 hours of work or if it does it should not happen twice in a row* – the impact cannot be evaluated on an event extraction system.
- *Permanent contact with the client insures that there is always someone to answer the developers' questions*.
- *Unified coding technique*– considered the real key to the success of a project. It is essential for the team to use the same coding conventions, explanation patterns and uniform codes.

Conclusions

Designing a system is influenced by a series of factors. It often depends on the experience of the designers. There is a saying: every second system designed by the same designer is better than the first one. The affirmation has full support because the designer does not make the same mistakes twice.

Another aspect is that some problems have to be solved with special methodology. Using these, the designers get better results with fewer resources. Objective methods are used with success for designing event extraction systems.

References

[1] Avornicului M., (2009).Data mining în Internet, *Editura RISOPRINT*, Cluj-Napoca

[2] Avornicului M., (2010). Informatikai rendszerek tervezése és menedzsmentje, Kolozsvár, *ÁBEL Kiadó*, Kolozsvár

[3] Chen G., Wei Q., (2002). Fuzzy association rules ant the extended mining algorithms, *Information Sciences*, pp. 201–228.

[4] Cutting D. R., Pedersen J. O., Karger D., Turkey J., (1992). A cluster-based approach to browsing large document collections. In *Proc. of SIGIR-92, the 15th ACM Int. Conf. on Research and Development in Information Retrieval*, Copenhagen, pp. 318–329

[5] Han J., Kamber M., (2001). Data Mining: Concepts and Techniques. *Morgan Kaufman Publishers*

[6] Han J., Kamber M., (2006). Data Mining: Second Edition Concepts and Techniques. Morgan Kaufman Publishers

[7] Jing H., McKeown K.R., (1999). The decomposition of human-written summary sentences. In *Proc. of the 22nd Int. ACM SIGIR Conference on Research and Development in Information Retrieval*, , Berkeley, California, pp. 129–136

[8] Kruchten, P. B., (1998). The Rational Unified Process*:* An Introduction – *IEEE Software*

[9] Lin B., (2007). Web Data Mining – Exploring Hyperlinks, Contents an Usage Data, *Springer*

[10] Lin T. Y., Xie Y., Wasilewska A. – Lian C. J., (2008). Data mining: Foundations and Practice, *Springer*,Berlin

[11] Markov, Z., Larose D.T., (2007). Data Mining the Web, *John Wiley & Sons*

[12] Raffai M., (2005). UML 2 modellező nyelv, *Palatia Nyomda és Kiadó*

[13] Sommerville I., (2010).*Software Engineering, 9th Edition*, Addison-Wesley

[14] Zaki M. J., (2000). Parallel and distributed data mining: An introduction. In Zaki and Ho(eds.), *Large-Scale Parallel Data Mining*, LNAI 1759, Springer-Verlag