Abstract

In modern software development, a multi-level architectural design and the use of Web- and component-based technologies are key factors for standardization (enhancing reusability), quality assurance and cost effectiveness. Clearly, this is not only true for software development in “trendy e-domains” like E-Commerce, E-Learning etc., but also in conventional industrial application areas, including information systems for technological process control (TPC-IS). This paper focuses at the latter ones. It outlines the most important common features of TPC-IS, discusses the impact these features have on a Web-based architectural TPC-IS design, and proposes a multi-level TPC-IS cross architecture considering these impacts. The appropriateness of that cross architecture is illustrated by reference to some projects which have been carried out for the Ukrainian gas-transport industry. Adequate technologies like MS COM/DCOM and ActiveX, XML and RDF are choosen, tested, and discussed with respect to TPC-IS implementation issues.

Section 1. Introduction

Nowadays the automated control of complex technological processes in real-time mode became an important technical problem, because of the time and cost constraints, which have to be fulfilled in this case. Appropriate computerized systems are supposed to be able to capture necessary data getting from technical devices, to send the appropriate control commands in order to manage them, to visualize the technological processes, and to support decision making activities by operating stuff etc. From our point of view, and accordingly to some international studies (see e.g. [1,2]) the main features of information systems for technological process control (further abbreviated as TPC-IS) could be listed as the following:

- distributed system architecture
- large-scale and multi-dimensional relationships in the system topology
- usage of heterogeneous data resources
- incorporation of some legacy software solutions
- continuous working cycle (“24x365” operating mode) including the real-time specific features
- diverse human actor groups involved in the process control framework (users, dispatchers, decision makers etc.)
- necessity of actor’s support in some extreme system operating cases (error massaging, exception handling, system error analyzing etc.).

All these facets impact the system architecture design tasks, which have to be done by TPC-IS development, especially taking into account the advent of modern Web-technologies in several technical domains. Accordingly to this the main aim of our paper is to elaborate some possible vision about the architectural framework and appropriate software solutions for Web-
based TP-IS with respect to: 1) from one side taking into account the special system features listed above, 2) from another side focusing attention on some general modern trends in SW&IT domain. This is done by investigating of the concrete TPC-IS application domain, namely the fuel and energy complex, where attention also is paid at the international level, especially in European Gas Research Group recommendations [3], and in the Ukrainian publications (see e.g. [4]). Our paper is structured as following: Section 2 focuses on some modern trends in TPC-IS domain and gives a briefly overviews of “state-of-the-art” on these issues especially in the Ukraine. In Section 3 we discuss the impact of TPC-IS features on system architectural design, and we present our first vision about one possible version of its cross system architecture. Some modern information technologies and software tools needed for its implementation are considered in Section 4. In Section 5 we present our feasibility study based on the real TPC-IS projects, which we performed and perform currently in the gas-transport and gas-field industry branches. In the Section 6 we make some conclusions, and we define briefly our future works have to be done in this R&D domain.

Section 2. “State-of-the-art” in TPC-IS Development Domain, and the Situation in the Ukraine

The main international trend in the domain of TPC-IS development is the usage of SCADA (Supervisory Control And Data Acquisition) system concept [5-9]. This class of computerized control systems could be considered as some extension of CAD/CAM systems family with respect to hard-ware devices level, which they are interfacing with. Typical SCADA system is a special software solution facilitating a lot of so-called programmable logical controllers (PLCs). The main goal of any SCADA system is to collect the technological process data, to represent them in proper time and form, and to accomplish the necessary control actions. SCADA systems are used not only in industrial processes: e.g. gas-extracting, steel making, power generation (conventional and nuclear) and distribution, chemistry, but also in some experimental facilities such as nuclear fusion [6], and for similar mission critical industrial processes where reliability and performance have the vital importance.

Nowadays on the Ukrainian market of SCADA systems, along with the systems provided by well-known abroad developers, like TraceMode (provided by AdAstra and its Ukrainian distributor [7]), and MOSCAD (Motorola SCADa system, which was installed on the plants of UkrTransGaz departments [8]), only one SCADA system developed by Ukrainian company is presented. The matter concerns the “Contour” SCADA system, and its 1st version was developed at the beginning of 2001. Accordingly to materials published on the Ukrainian SCADA-Web-resources [5], we tried to analyze the advantages and drawbacks of “Contour”system.

As “Contour” positive aspects, we can indicate the object-oriented design approach used; the certain flexibility, which is based on usage of ActiveX technology; the OPC (OLE for Process Control) standard support, which gives a possibility to work with different types of LPCs, and makes their installation easy. However, we should emphasize a number of drawbacks, among them: the 2-tier system architecture; impossibility to simulate non-regular situations (“Contour” can only represent them); lack of powerful retrospective database subsystem (the system only stores the technological data in Paradox format). And, last but not least, the “Contour” currently does not support any Web-based clients.


The specific TCP-IS features listed above definitely impact an architectural system design, which has to be done for real technological process engineering. If we have to elaborate some
TPC-IS solutions for an appropriate technological process to be controlled, generally there are three possible ways for this purpose available:

- new TPC-IS design and implementation
- usage (installation and customizing) of some industrial SCADA system as the COTS (Commercial Off-The-Shelf) solutions
- legacy software system reengineering

We believe that the best choice from the approaches listed above definitely could be determined by the given information infrastructure of the appropriate technical object to be automated. This choice is related to technological process features, and has to be done taking into account existing time and cost constraints to be fulfilled. The following general problem could be formulated and should be solved in this domain: how to elaborate the matrix for cost/efforts estimation which are needed in different project cases, and how we could be able to provide decision making support for technological process managers and software developers, who are facing these problems with?

But we are deeply convinced, that in any case the target TPC-IS should have the system architecture, which

1) is based on the well-known 3-tier paradigm of distributed system design, including
   - data presentation (user interface) tier
   - business logic tier
   - data access tier

2) and furthermore, this architecture is supposed:
   a) to provide an advanced (multi-dimensional, multi-media) technological process data visualizing at the data representation tier. There must be 2 options of such visualization available
      - “is” - mode for regular system operation
      - “could be” – mode for simulation of several non-regular (worst-case) control situations
   b) to have the following services at the business logic tier:
      - a set of unified components for data exchange with external devices of several kinds
      - a special problem-oriented control facility for business components monitoring (in our concept it is called as P-Monitor)
      - a forecasting sub-system (e.g. based on neural-network methods etc.)
   c) to support a process-oriented system information model at the data access tier including:
      - Operational database, which captures the process data for visualizing issues, it should be elaborated as a typical OLAP-database facility
      - Retrospective database, which collects the process data in some aggregated formats, and which has be designed as some sophisticated multi-dimensional data-mart solution
      - Meta-data model for diverse data resources fusion and integration, which could be examined as a domain knowledge-based control engine leveraging up the whole system information architecture.

This architecture is presented in UML notation on Fig.1. Further we will illustrate this our first architectural vision with some software solutions elaborated in our real projects performed in 1999-2001 (due to limited volume of this paper we did not present them in all details). But at first we have to make some remarks about technologies and tools, which could be used for our system design and development issues.
Section 4. Adequate Information Technologies and Tools: Choosing and Discussion

4.1. Some issues of the component-based software development

Modern component-based technologies give us a possibility to “atomize” the TCP-IT architecture into the subsets of reusable software artifacts for each of technical devices to be controlled (e.g. such types of devices as pumps, gas-tanks or separators in gas-transport and gas-field industry). MS COM/DCOM component model [10] with ActiveX, DirectX or OLE technologies, and CORBA/EJB architecture elaborated by OMG (Object Modeling Group) [11] can be used for this purpose. The usage of these approaches is well enough described in the appropriate literature, and we would like only to present some our own suggestions concerning component-based implementation of TPC-IS architecture introduced above (s. Section 3), especially for the subset of application components, which are responsible for the technological process data visualization. Furthermore, we have to consider such side conditions in the Ukraine as:

- dominated usage of Microsoft IDEs and appropriate software solutions
- relative slow communication channels (mostly not more as 33600 bps)

Taking into account these points we could make the following suggestions:

- visualizing components have to be implemented as MS ActiveX controls, which can be easily inserted into HTML – pages and can be viewed from a web site all over the world
- since ActiveX controls are basically just COM objects implemented a specific set of interfaces, it makes sense to use ATL (Active Template Library) instead of MFC
(Microsoft Foundation Classes) for this purpose. Firstly, ATL is small, and secondly, controls written with ATL do not require the availability of large run-time DLLs as MFC-based controls do. ATL gives us a tremendous winning in number of transferred data. Component based on ATL do not need any preinstalled libraries for their execution, and their size is much smaller than size of component based on MFC. As our experience has shown, DLL - compiled library that contains two components based on ATL is about 40 Kb, for compare: one component based on MFC is about 50 Kb and also it needs 900 Kb of MFC-libraries additionally! Also, ATL-based components are essentially more efficient, thus results in tight code, well-suited to handheld devices featuring small memories, and less powerful processors etc.

- using ATL we can create a ActiveX control with Dynamic HTML (DHTML) capability.

As a result, we can state that ATL-based components will be faster and smaller, nearly at 10 times, in comparison with similar components developed based on the standard MFC libraries.

4.2. XML and RDF as a tools for system meta-data modeling

One of the main problems which have to be resolved by any Web-based non-trivial application is the heterogeneous data representation and processing. Especially this problem is urgent for TPC-IS development, because of a lot of diverse data resources to be integrated and to be managed in its environment. We introduce the Meta-data Model (MDM) for this purpose in the TPC-IS architecture proposed above. Now we have to choice an appropriate tools, which could be adequate for its implementation. We believe that RDF (Recourse Description Framework) [12] and XML (Extensible Markup Language) [12] should be useful in order to met these challenges.

We propose to consider our MDM as a collection of XML/RDF – based documents which are supposed: 1) to provide data interoperability between different TPC-IS components exchanging machine-understandable information on the Web; 2) to support advanced data processing, taking into account the topology and operating mode of technological processes, which have to be controlled by the appropriate TPC-IS. We would like to demonstrate our approach for the MDM-building using XML/RDF-features with the example of the following problem solving: how we have to force our TPC-IS to generate HTML-pages, on which some visualizing components should be placed properly?

At first, assume we are working with the component, which provides the process data visualization for the certain TPC- device, e.g. Pump. This component could be placed on the HTML page using object tag like (see Fig. 2):

```
........

  <OBJECT classid=clsid:1C7B2B55-8F13-11D4-9573-204C4F4F5020
   id= “Pump” style= "HEIGHT: 312px; WIDTH: 891px">
</OBJECT>

............
```

Fig. 2 HTML script for one visualizing component

This HTML declaration just tells to Web-browser where to get code of component, and how it should be registered on the client PC. As far as we are dealing only with one separate instance of the only one device class Pump included in an appropriate TPC-IS, this declaration would be enough. But for the real-life TPC-IS we have to assume additionally:

- there are different device classes, e.g. Pump, Gas-Tank, Separator…etc.
• even inside one device class its several instances \{pump1, pump2,….. pumpN\} could be aggregated into some blocks or subsystems.

Further, assume that our system has to generate a lot of HTML – pages for the dynamical process data visualizing. In this case HTML declaration given above (on Fig. 2) doesn’t support placing of visualizing component only, which actually are needed for several representations. Using XML features we can describe what classes of components are valid to be included into target HTML. To formulate the rules for the components placing, we construct a number of DTD (Document Type Definition) scripts for each class of the system devices. All visualizing objects can be identified by their CLASSID that is unique. Therefore, the DTD-script for the aggregate Pumps Block could be written as (see Fig. 3):

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE RootDocumentName [
<!ELEMENT PumpsBlock (Pump+)>
<!ELEMENT Pump (CDATA)>
<!ATTLIST Pump
    id CDATA #REQUIRED
    Param_1 CDATA #IMPLIED
    Param_2 CDATA #IMPLIED
    ...
    Param_N CDATA #IMPLIED>]
  <Pumps Block>
    <Pump id="1"
      Param_1="Value1"
      ...
      Param_N="ValueN">
      <![CDATA[<OBJECT classid="clsid:D4F9CBE5-84AF-11D4-955F-204C4F4F5020"
          id="Pump1" style="HEIGHT: 192px; WIDTH: 515px"></OBJECT>]]></Pump>
  </Pumps Block>
```

Fig. 3. DTD script for aggregated visualizing components

This DTD restricts the placing of any other kinds of visualizing components except the instances of the device class Pump into the appropriate HTML document, which has to be formed and to be transferred to client Web-browser. Moreover, in order to structure our metadata in such manner for the different device classes we could use the RDF-based script like presented on Fig. 4 (the 8 pumps are supposed to be aggregated in one block):

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<?xml namespace href="http://www.w3c.org/RDF" as "RDF"?>
<?xml namespace href="http://main_server.com/RootSchemaName-shema#" as "LOCAL"?>
<RDF:RDF>
  <LOCAL:PumpsBlockDescription>
    <LOCAL:BlockType> PumpsBlock </LOCAL:BlockType>
    <LOCAL:NumberOfPumps> 8 </LOCAL:NumberOfPumps>
  </LOCAL:PumpsBlockDescription>
```

Fig. 4. RDF script for aggregated visualizing components
4.3. DBMS motivated choosing for Web-based TPC-applications
Besides the generally accepted requirements to DBMS (reliability, data security, high productivity, integration with development tools, reasonable price and others), we can name the ones which are caused by the features of TCP systems and Web applications:

- ability to capture data with high frequency
- large volumes of information storing and processing
- integration with other subsystems of TPC
- XML data exchange format, and multimedia data support

Let us compare 2 widespread DBMS - MS SQL Server 2000 and Oracle 9i – with respect to their correspondence to requirements mentioned. Here we will notice the results of comparison only on a few (most important) criteria, more fully such analysis can be found, e.g., in [13]. MS SQL Server can be preferable with comparison to Oracle if the criteria “cost” has vital importance (this situation is typical for the Ukrainian customers) and the size of data is not large (it is possible, when TPC-IS should store with high frequency only operational data, and retrospective section could contain some aggregated data).

However, most standard DBMS solutions (including MS SQL Server and Oracle) do not allow to store the data with high frequency (about milliseconds), and their integration with the existing TPC - subsystems can not be easily achievable. Consequently, it can be necessary to use some new type of DBMS, which are specially oriented on working with such requirements. Among these systems we could name the IndustrialSQL Server provided by Wonderware company [14], which was created on the base of MS SQL Server. This solution represents the special kind of relational real-time database management system, reaches such results by using special algorithms of data storing and compression, and it is able to save the disc space for data storing by a factor of 50-100.

Section 5. Feasibility Study: TPC-IS in the Ukrainian Gas-Transport and Gas-Field Industry

5.1. Universe of Discuss (UoD) description. Some side conditions
As already mentioned above, our experience collected and results presented below are based on the “real-life” TPC-IS projects performed in 1999-2001 in the Ukraine. We have been working for both new developments and reengineering activities on two kinds (types) of such systems, namely:

(a) on typical gas compressor station (GCS) as an essential part of Ukrainian main pipe-line gas transport infra-structure, this project was carried out on the GCS in the Romny region (there are more then 70 such GCS total in the whole gas-transport Ukrainian system)

(b) on small plant for complex gas refinement (PCGR) which is a “backbone node” of regional gas-field industry (especially in Kharkiv region); there are the 9 such PCGR at present, and some new one should be implemented in the near-term outlook.

In the case (a) we worked in the environment of already existing software, taking into account the necessity to integrate our solutions with legacy TPC-IS components. In the case (b) we started to develop a new information system for existing TPC-facilities, but also with respect to existing “information environment”, e.g. dedicated data channels, some legacy IT solutions running on the others PCGS, etc. In both case at first we were supposed to perform some kind of TPC system architecture reverse engineering: we elaborated the object-oriented
vision of system functionality using UML, in order to get some start points for target TPC-IS design requirements (the appropriate UML diagrams are presented in [15,16,17]).

5.2 Some TPC-IS architectural software solutions elaborated

Based on our TPC-IS cross architecture proposed in Section 3, and using information technologies and software tools discussed in Section 4, we have elaborated:

(a) 2 unified software components: TechXObject and XProtocol for data exchange between LPCs of several types (s. more detailed about them in [16]). Both these components facilitate as OLE-application data server, and both are implemented as Windows NT – services.

(b) the special customized database solution for process data storing and analyzing, presented in [18]. It is based on MS SQL Server 7.0, and is connected with application data server via appropriate component SQL Connect Service.

(c) a set of visualizing components, which are domain-oriented, and which represent appropriate mnemonic diagrams of the technological processes to be controlled by our TPC-ISs. They are implemented as ActiveX controls [17].

The common deployment diagram for these components is presented on the Fig. 5, and shows also the technologies used for components interaction: COM/DCOM, HTTP, ADO and ASP (Active Server Pages) for server-side scripting.

As the result for end-users operating within TPC-IS (for internal operators) and outside of it (for remote users) the HTML-based interface is provided, some its examples in form of screen snapshots are shown on the Fig. 6, (a) and (b).
Section 6. Conclusions and Future Work

As our projects experience has shown, besides the general design problem formulated in Section 3, there are a lot of both conceptual and technological tasks, which have to be done in future, some of them are: to elaborate effective reengineering approaches for legacy TPC-IS with respect to Web-platform transition; to develop concept and IT-support for Web-based stuff’s training; to find out more efficient server-side software solutions taking into account
growing number of system users, etc. We currently are working on these issues in the framework of the concept elaborating for the n-level WWW-based (abbreviated as N-W3) distributed TPC-IS, which is supposed to be implemented in Kharkiv region.

Acknowledgments

We would like to express our thanks to Mr. S.V Ovasapov and Mr. A.I. Kuzmin (design firm «PromAvtomatika», Kharkiv, Ukraine), and also to Mr. Y.M. Chrapach (“KharkivGas-Vidobuvannay” company) for their fruitful project cooperation.

References (all the URL given below were accessed in December 2001)

2. SCADA-??????? ????????????????????????. ??? "?? ?? ". 2001.- 234 ?.
3. URL: http://www.eurogas.org
4. ????????????? ? ?., ?????? ??. ? ? ????????? ??????????????- ????????? ????????????
5. URL: www.scada.com.ua/article2.html
6. URL: http://www.hut.fi/Yksikot/Auttieto/rap2/kanninen/kanninen.htm
7. URL: www.tracemode.ru
8. URL: http://www.motorola.com/moscad.htm
10. URL: http://www.microsoft.com/com/
11. URL: http://www.corba.org/
12. URL: http://www.w3.org/
13. URL: http://www.interface.ru/
14. URL: http://www.wonderware.ru/indsqlser_prod.htm
18. ????????? ? ?.. ????? ? ?.. ????????????????? ????? ????? ? ?? ??. ????????? ?
19. ??????????????. // ?????? ? ?????, ?????????????? ????????? ??????????????. ? ??.