Strategy for the amendment of plant information models by means of OPC UA

Eugen Reiswich
Software Engineering Group
University of Hamburg
Hamburg, Germany
reiswich@informatik.uni-hamburg.de

Alexander Fay
Automation Technology Institute
Helmut-Schmidt-University
Hamburg, Germany
alexander.fay@hsu-hh.de

***Abstract*— Companies with established and reliable control systems are often confronted with the question of how to benefit from upcoming innovations in science and practice without removing or extensively changing their systems. One of the currently often discussed innovations is the new standard OPC Unified Architecture (OPC UA). OPC UA provides an improved data transport and a semantically rich information model with the aim to improve operators supervisory and control tasks.**

OPC UA is a promising candidate to provide the next generation platform for control system development that better meets today's technological and business requirements. However, during our research activities over the last three years in two projects with industry partners, we encountered difficulties in introducing OPC UA concepts into existing control systems as many currently proposed integration solutions either replace existing systems or provide technology-driven integration solutions which are unable to reveal the innovational strength to operators and engineers. Companies wishing to benefit from upcoming OPC UA concepts and adjust their control systems towards the state of the art also wish to protect their investments in existing tools, data sources and employees' training. In this paper we approach OPC UA from a user-centered design view. We first derive requirements on how to improve the operators' and engineers' work situation. Subsequently we show which OPC UA information model properties can cope with these requirements in a standardized way. Finally we present our observations on the challenges faced by companies when introducing new technologies into existing control systems and we propose an evolutionary strategy on how to combine existing tools and models with new concepts.

***Keywords*—OPC UA, information model, user-centered design approach, evolutionary integration.**

I. INTRODUCTION

Many companies run control systems which have been developed and steadily improved over years, but eventually risk to expire as soon as new technologies emerge. When companies encounter new technologies they can usually decide to either redevelop existing systems or to integrate new concepts. While a redevelopment requires massive system changes and is thus virtually impossible, the integration success mainly depends on the chosen strategy. Rather than introducing new concepts in one big step, nowadays software engineers prefer an iterative approach that incrementally introduces many minor changes [1]. Current evolutionary integration approaches in the realm of OPC UA try to retain existing systems using software technical concepts like wrapper, proxies, gateways and adapters [2]. These approaches however limit new solutions, leaving operators and engineers - despite all the integration effort - with yesterday's problems.

OPC UA is still in an early stage of adoption and current publications and discussions almost exclusively focus on technological aspects like web technology, security, platform independence and data transport [3], [4]. On the other hand the OPC UA information model has the opportunity to improve operator's and engineer's work situation supplying semantically rich plant information like device types, hierarchical plant structures and corresponding views. However, during our work with industry partner we repeatedly noticed that we need to provide further benefits beyond the semantically rich plant information to justify the integration effort. Operators and engineers especially requested scenarios that demonstrate how information models can significantly improve supervisory and process control situations. Moreover companies requested integration solutions that reuse most of the existing control system's tools and models and thus reduce the integration effort and risks.

Based on 3 years of experience in two projects on control systems for particle accelerators we had the chance to develop applications based on new information models. In this article we will provide an overview of current solutions for the fundamental control system tasks as Data Access (DA), Alarms & Events (A&E) and Historical Data Access (HDA) at HMI-layer. Then we will show how these proposals can be addressed by OPC UA information models in a standardized way. Finally we will demonstrate how OPC UA based concepts can be evolutionary integrated in existing control systems without entirely replacing them.

After a short overview of OPC UA we summarize problems we have encountered while introducing new technologies into existing control systems. We then present user-centered scenarios on how information models can improve supervisory and control tasks, propose appropriate solutions and an evolutionary integration approach. Finally we present a set of problems we have not found solutions for yet and discuss ideas how to deal with them.

978-1-4673-0311-8/12/$31.00 ©2012 IEEE

II. Overview over the challenges integrating new technologies impose

After the successful adoption of OPC in thousands of applications follow-up the standard OPC UA has been introduced to satisfy new technological and business requirements. To better set former OPC apart from OPC UA it is now called *Classic OPC*. The fundamentals of OPC UA are data transport and information modeling [3, p. 19]:

- **Data transport**: the major improvements in data transport concern the renunciation of COM/DCOM technology towards a platform-independent, secure and reliable technology.
- **Information modeling**: In OPC the semantics of provided data is usually limited to the tag name and some additional properties [3, p. 19]. OPC UA information models provide semantically enriched plant information like device types, hierarchical plant structures and corresponding views that can help to create subsets of large plant structures.

Although these concepts sound fairly reasonable, integrating them on top of existing control systems imposes a host of challenges. Some of them stem from mainly technology-centered discussions, others stem from versatile but insufficient integration solutions in the very early adoption phase where existing control systems have to be combined with new concepts.

A. User-centered vs. technology-centered design

When new technologies need to be integrated into existing systems, one can distinguish between user-centered and technology-centered design approaches [5], [6]. While the technology-centered design approach primarily focus on the technical product and its performance aspects, the user centered-design approach considers a product from the end user perspective with the aim to improve and better support human work. Current publications and discussions in the realm of OPC UA almost exclusively focus on technological aspects with a focus on platform-independency, security, data transport and web technology. Whereas user-centered improvements are primarily motivated by semantically rich plant information models. During our work with industry partner we repeatedly noticed that existing control systems already provide many of the plant information OPC UA promotes, although they are often spread over many resources. While the information model benefits are not very prominent at first glance, the integration effort with wide-ranging changes to existing systems in contrast is. In this situation companies seek for answers to the following questions:

- **How to protect investments in tools and training?** Many control systems have been developed and steadily improved for years. Hence companies usually intend to protect their investments in tools and training and requested solutions that upgrade existing systems.
- **Which benefits can users gain from information models beyond semantically rich plant data?** During our workshops and interviews, operators and engineers especially asked for practice-oriented scenarios that demonstrate how information models can improve supervisory and process control tasks.

B. Evolutionary integration

Control system software is according to Lehman's program classification an E-type system [7] and thus evolves in a changing environment. These changes are caused by users and upcoming technological innovations. Rather than replacing existing systems by upcoming innovations in a big bang, evolutionary strategies that favor minor changes in an iterative process proved to be more promising [1]. When existing systems face new technologies companies need to clarify the following questions:

- **How to migrate available information and new models?** Existing control systems already incorporate many models e.g. within CAE-tools, engineering databases and in piping and instrumentation diagrams. Reusing these models would be most helpful.
- **How to integrate existing software and new concepts iteratively?** OPC UA proposes many improvements in data transport and information modeling. An evolutionary integration strategy should allow operators and engineers to start incrementally with the obvious improvements where most benefits are expected. This practice also helps them to understand and handle the sometimes overwhelming technical details.

Many integration aspects have already been discussed to some extent and first evolutionary integration solutions appear [3, p. 293ff], [8], [9] however with some conflicting outcomes:

- **Wrappers:** wrappers enable new OPC UA clients to access classic OPC servers through mappings between OPC UA and COM technology [2], [3]. While this approach leaves existing servers mainly untouched, it is unable to expose new functionality to clients as classic servers are unaware of new OPC UA concepts. OPC UA clients are therefore limited, despite the integration effort, to the functionality of classic OPC servers [2]. Moreover it is not clear how new OPC UA and classic clients should interact.
- **Proxies**: proxies enable classic OPC clients to access new OPC UA servers [2]. The advantage of this approach is that classic clients remain nearly untouched. Classic clients are however unaware of new OPC UA features and thus don't' use them by default.
- **Gateways and adapters**: Gateways provide real-time, alarm and history based information from a single source rather than contacting three different classic OPC servers [3]. Adapters are special gateways individually developed for control systems to better expose new OPC UA features and improve interoperability [2]. Neither gateways nor adapters provide upgrade solutions for existing tools to facilitate new concepts and improve user workflows.

Although first integration solutions exist, they also introduce new questions leaving control system developers in an uncertain situation on how to avoid looming disadvantages and gain most of the upcoming advantages.

C. Our approach in a nutshell

Approaching new technologies from a user-centered design view we identified the information model to be the most obvious candidate to improve operator's and engineer's work situation. In several user interviews and workshops we identified scenarios and state of the art solutions on how models can improve supervisory and process control situations. Consequently we have implemented these scenarios in a prototype and studied which model characteristics can improve and unify existing solutions in a standardized way. Thereby we present an evolutionary, client-side integration solution which doesn't influence existing servers and at the same time reveals the power of information models to operators and engineers. Thus we begin with the integration in small steps within a bounded context in order to reduce the overall integration risks.

We further reduce the integration effort by reusing information provided by existing models and upgrade existing tools where possible. As models change over time they are threatened by data erosion. In order to reduce this threat we provide engineers with synchronization tools and concepts that are able to detect and resolve deviations between existing and new models.

III. Information Models for Data Access

A. Display Design

Operators supervise control processes and interact with the automation system using schematic displays. In many control system developing displays is still a manual task predominated by copy & paste operations. This is expensive, time-consuming and hampers later reconfigurations [10]. There are several ongoing research activities trying to automate display development by reusing engineering data from previous planning phases [10], [11], [12]. In our approach, we pursue a semi-automatic display development based on the *ObjectType* concept as proposed in [13]. ObjectTypes enable to assign similar devices e.g. temperature sensors to a common device type like PT 100. Instead of configuring hundreds or thousands of individual device instances, display designer rather configure types and transfer these settings to associated device instances [13]. This configuration can encompass e.g. the widget used in HMIs, alarm behavior and corresponding faceplates for detailed control.

B. Evolutionary integration

To implement the above mentioned concept we have used several concepts in combination:

- **OPC UA information model**: contains all device instances of interest and corresponding device types. In our approach we do not interfere with existing servers and corresponding data transport for real-time data. Thus we integrate the information model client-side by providing additional interfaces that enable new DA-client features e.g. browsing the information model's address space (see Figure).

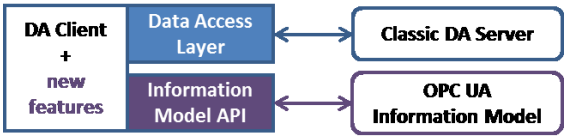

Figure 1: Information model integration

- **OPC UA Type Library***:* contains all device types available from the *InformationModel*.
- **Widget library**: most control systems already provide a *WidgetLibrary* containing templates with commonly used widgets. We reuse these libryries.
- **TypeConfigurationService**: this service is used to link OPC UA device types with existing widgets and additional configuration information. As these configurations are related to display design and thus to UI, the *TypeConfigurationService* makes it possible to split UI and non-UI related code and keep the information model small and manageable.

Display designers configure types by navigating through a *TypeLibraryTool*, selecting the type of interest and linking it to a corresponding widget, alarm behavior or faceplate. Once the type-based configuration is finished, display designer browse the *InformationModelTool* to find concrete device instances of interest and drag and drop them into displays (see Figure 2). During the drop process the type information of the selected device is used to retrieve all HMI relevant information.

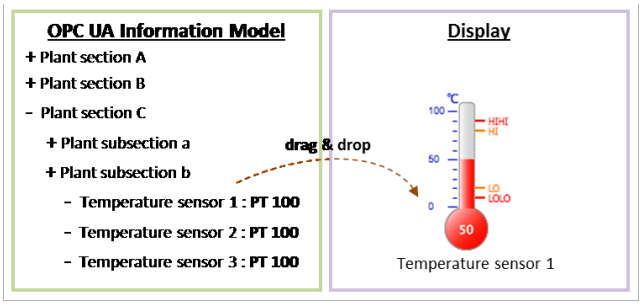

Figure 2: Semi-automatic display design

The type library and information model tools incorporate many new OPC UA based concepts and thus have usually to be newly developed. According to [13] creating displays based on a type concept allows:

- **Reconfiguration**: *InformationModel* and *TypeLibrary* properties change over time e.g. server connection properties are added later on or modified, widget colors and alarm behaviors are changed. Type-based configuration tools enable operators to perform these changes in one single place.

- **Validation concepts**: Widgets in displays are loosely coupled with corresponding devices within the *InformationModel* through a DeviceID. We used this connection to detect changes within the *InformationModel* e.g. delete operations, broadcasted these events to displays and marked corresponding widgets as deleted. Hence operators are able to faster indicate dangling widgets and improve the display's quality. Types can expose properties with boundary information like MIN and MAX values. When widgets are linked to types validation concepts can help to detect whether a type can satisfy a widget's interface e.g. a temperature widget is improper for a magnetic valve type. Many such validations can be done at display design-time and thus reduce errors during run-time.
- **Connecting devices to displays:** in some cases e.g. when alarm situation occur, operators need to know in which displays a specific device is represented. Maintaining these relationships is often a manual process, which is error-prone and expensive. Using the DeviceID stored in widgets, we developed algorithms that automatically scan all displays and find those containing the devices of interest.

The display design example we have chosen deals with rather simple devices. However, the device type concept has the ability to also support more complex devices like turbines and engines although more configuration effort is required.

IV. Information Models for Alarms & Events

A. Alarm challanges

In alarm situations, operators often need to handle many alarms within a short period or react to dispensable alarms caused by devices in maintenance. Well established and proven alarm philosophies help to cope with this situation and reduce alarms at their source by providing better alarm configurations [14], [15]. Nonetheless most experts agree that even the best alarm philosophy and configuration will not avoid alarm floods or alarms caused by devices in maintenance [15, p. 22], [16]. In addition to that, many alarm lists are unstructured and thus hamper operators alarm awareness [17].

Instead of trying to optimize the alarm configuration we rather focus on providing operators with better tool support that can help to cope with the mentioned problems. As stated in [15, p. 22] the overall alarm situation can be improved by providing operators with extra details on the plant area affected, associated schematics and trends. We thus used the information model's hierarchical plant structure and developed tools that are able to assign alarms to corresponding devices within a hierarchy and such improve operator's assessment in critical situations. The hierarchical structure also helps to mute entirely plant sections and thus filter e.g. maintenance alarms (see Figure 3).

In large automation systems operators have to deal with thousands of devices. Information models providing a view concept helped us to develop tools which enable operators to

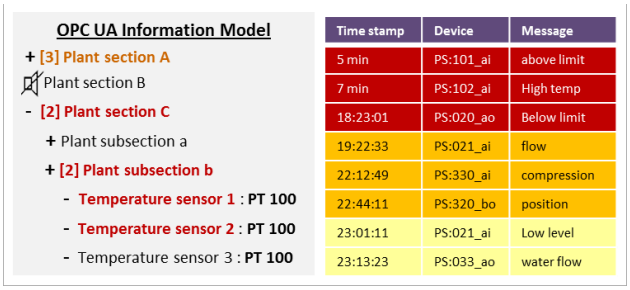

Figure 3: Information model based alarm handling

create named filters as proposed in [15, p. 114] with a hierarchical subset of a plant with devices of interest and thus help to concentrate on relevant plant sections.

In control systems we observed, operators often reconstruct historical alarm situations by studying flat excel sheets or log files. We help operators to reconstruct past alarm situations in a more structured way by switching the alarm tool's input from real-time events to historical alarm events.

B. Evolutionary integration

In most control systems, alarm handling plays a major role. Changes to alarm systems can affect the system stability or even cause damage within the plant. To avoid revolutionary changes in existing alarm systems we integrate our alarm approach client-side. For that reason we developed a new *AlarmService* that connects to the classic A&E DAL and subscribes to alarms (see Figure 4).

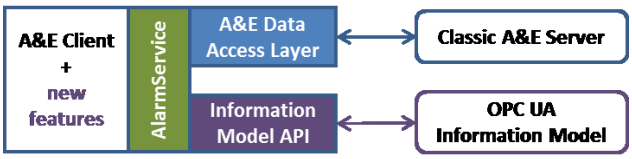

Figure 4: Alarm service integration

When alarms arise, the AlarmService uses tag names or better DeviceIDs (in case classic server and InformationModel share the same IDs) to find corresponding device nodes within the InformationModel. We use nodes to traverse child-parent relationships in order to assign alarms to corresponding devices within the alarm tree. Finally we create an OPC UA-based alarm containing alarm and hierarchy information and store it in an *AlarmModel* within the *AlarmService*. This model reflects the current alarm situation and sends updates to the Alarm-UI whenever the model is changed.

The alarm UI we worked with is comprised on the one hand of the existing alarm list and on the other hand of a new developed OPC UA-based plant representation. Unfortunately the existing alarm list we worked with was unaware of new alarm concepts. We encountered two solution approaches to support new alarm concepts reusing existing alarm lists:

- **Enable AlarmService to provide both legacy and new alarm concepts**: this solution doesn't require any changes to the alarm list. While less intrusive, this solution increases the maintenance effort as legacy and new concepts have to be maintained at the same time.

- **Change alarm list to comply with new concepts**: this solution is more intrusive as the alarm list has to be adapted to new concepts. In the long run, however, this solution will reduce development effort and improve the software quality as only one concept has to be understood and maintained.

To facilitate a reconstruction of historical alarm situations we developed a *HistoryAlarmService* that connects to a corresponding database. A complementary alarm tool enables operators to switch between current and historical alarms and thus switch between alarm services reusing the same alarm tools (see Figure 5). Furthermore we provide additional UI-tools that allow operators to step through historical alarms.

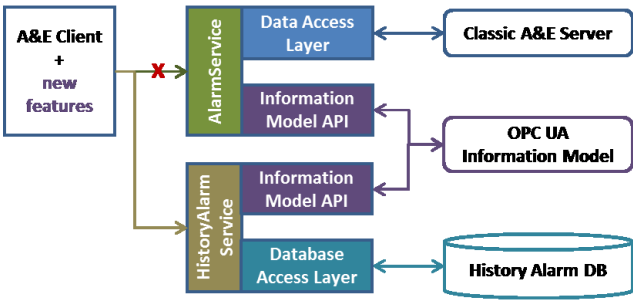

Figure 5: Redirection of alarm services

V. INFORMATION MODELS FOR HISTORY DATA ACCESS

A. *Evaluation of historical data*

Historical data helps to reconstruct a plant's situation for a point in time or period. To improve operator's understanding of a historical situation, dedicated servers supply data compression and analysis capabilities. These are used to create trends, calculate mean, maximum or minimum values [18]. During our workshops with engineers we noticed that trends help to evaluate selective historical data. In order to understand the comprehensive plant situation of interest, engineers have to amalgamate many individual trends into a mental model. This is time-consuming and hampers the understanding of historical situations.

To complement trend based analyses we intended to reuse schematic displays and redirect their input from real-time to historical data. The control systems we observed, however, provided row data without any boundaries or alarm information that displays require to be useful. Fortunately the information model is able to complement these pieces of information. We are currently working on a solution that supplies displays with a combination of row and information model data. As we do not plant to change displays and the way they subscribe to real-time data, we are currently working on a switch that is able to change the data source from real-time data to historical row data and enrich these with boundaries and alarm information provided by the information model (see Figure 6). By the time this article emerges, first tentative steps seem to be promising, however we have not been able to finish our development yet.

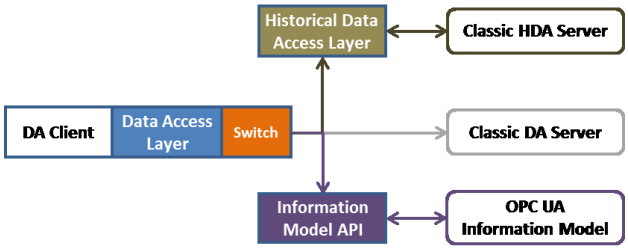

Figure 6: Historical data evaluation in schematic displays

VI. INFORMATION MODEL INTEGRATION ON TOP OF EXISTING MODELS

In the previous chapters we described which benefits operators and engineers can gain using new information models. What we haven't answered yet are important questions regarding information reuse from existing data sources:

- **Where do information models get their input from?**
- **How can information models reuse data incorporated in existing data sources?**
- **How do new information models cooperate with existing data sources?**

A. *Reuse of existing data*

During our work we encountered several promising candidates for data reuse:

- **CAE-tools**: the information we have been able to extract from engineering tools encompassed hierarchical plant structures, device types as well as I/O descriptions. However, according to engineers these were only about 60% of the required information needed to run a control system and thus had to be complemented. To reuse the engineering tool's data we used the build in Excel-export capabilities and developed a corresponding import adapter using the Computer Aided Engineering Exchange (CAEX) standard. The aim of CAEX is to foster vendor independent engineering data exchange [19].
- **Engineering databases**: these are used to configure classic server. Hence these databases contain the latest I/O descriptions. Some engineering databases we examined also provided to some extend type information and hierarchical plant structures. Other databases were rather flat lists with I/O configurations. To reuse the data we created SQL-adapters for the databases we worked with.
- **Legacy configuration tools**: in one control system we found a legacy configuration tool containing most of the information needed, stored using a company specific XML schema. As the source code of this tool was open we developed CAEX-based export and import adapters to reuse this data.

Reusing information and data from different models is usually a serious challenge in software engineering as nearly 60-80% of the integration effort has to be spent on reconciling semantic heterogeneity [20]. This task encompasses primarily finding semantic equivalences between model elements [21].

These equivalences have finally to be shifted from one source to the other using integration patterns like translator and adapter [22]. Fortunately we found OPC UA information models to have a high similarity of structure compared to existing models which substantially improved the information reuse. This stems from the fact that OPC UA rather distills and unifies many existing standards instead of introducing a totally new one [23].

During our work we found models derived from CAE tools rather useful for greenfield projects, whereas engineering databases and legacy configuration tools proved to be more useful for data integration. These sources reflected best the underlying plant structure and provided the latest and most comprehensive information. However, in all cases we were not able to completely avoid manual extra work as all existing models were initially created without the intention to be later reused in OPC UA information models. This task encompassed data type conversions and completing OPC UA relevant information not provided by the selected data source.

B. Coping with data erosion

Engineers change I/O configurations over time using reliable and proven configuration tools. As we do not substitute these tools, we have been very soon confronted with data erosion problems and needed appropriate synchronization concepts. First we had to detect deviations between the information model and the chosen data source. We encountered several strategies to recognize deviations (see Figure 7):

- **Modify existing configuration tools**: if the source code of existing configuration tools is available, they can be modified to send out a ping whenever the configuration changes. Once our synchronization tool receives a ping it starts a synchronization task. This concept ensures a quick detection of model deviations and facilitates minor and manageable changes.
- **Cyclic or manual synchronizations**: in many cases companies use third-party configuration tools which cannot be modified easily. In these cases cyclic synchronization concepts known from Windows or Mac OS operating systems or even manual synchronizations are possible solutions.

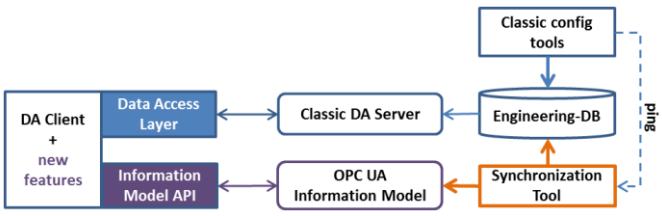

Figure 7: Information model synchronization

Synchronizing data between multiple sources with usually different semantics requires appropriate matching capabilities. These must be able to identify and map inter-model relationships. Rahm et. al. [24] differentiate element-level and structure-level matching.

- **Element-level matching:** considers in the simplest case only elements at the finest level of granularity such as attributes in an XML schema or columns in a relational schema.
- **Structure-level matching**: refers to matching combinations of elements that appear together in a structure such as relating two hierarchical structures to a single structure.

Over time engineers usually add, delete and modify process variables (PVs), their attributes, higher level devices like turbines and rearrange hierarchical structures. These changes can be synchronized fully automatically, using ontologies or even manually [20], [24]. Rahm et. al. state that in general it is not possible to fully automatically match two sources as many needed matching criteria are either not formally expressed or even documented. They propose concepts that are able to determine match candidates, which the user can accept, reject or change. During our work we developed in a first stage an algorithm based on element-level matching that is able to detect add, delete and field modifications concerning attributes, PVs and high level devices. We also developed corresponding tools that present deviations to engineers and help them to resolve these deviations manually.

VII. CONCLUSION AND OUTLOOK

In this article we presented our experiences made over the last three years in two projects, integrating information models in a user-centered approach. We described in typical scenarios which model properties are able to improve existing tools and workflows and thus help companies to approach state of the art solutions. Afterwards we have shown how to integrate these solutions into existing control systems in an evolutionary manner. Finally we demonstrated how existing data sources can be reused as input for information models and how these different sources can be synchronized. We thus complement current predominantly technology-based discussions with a user-centered approach and provide companies with a solution that reuses most of the existing tools and data. Although we pursued a solution that retains existing control systems and enriches them with new concepts, the integration task is still sophisticated. There are many reasons for that:

- **Evolutionary integration**: as we have chosen an evolutionary integration, we often needed to support existing and new concepts at the same time. This hampers the development process and causes additional maintenance costs.
- **Steep learning curve**: although first publications in the realm of OPC UA information models arise, there are many complex concepts that have to be understood by each developer in a team. Moreover operators and engineers have also to get used to new concepts and sometimes change their workflows.
- **Manual synchronization effort**: our primary goal was to upgrade existing tools where possible and reuse data they incorporate. Thus we have not substituted existing configuration tools and data sources but rather provided

predominantly manual synchronization concepts between new and existing models. This, however, can be a tedious task especially for large sized models [20].

We are constantly refining our approach to further reduce manual work as well as time spent on integration. According to feedback from operators and engineers we successfully demonstrated that information models are a good basis for a user-centered integration approach. The tools we have developed adjusted existing control systems towards state of the art and related well to operators' and engineers' needs in their daily routine.

REFERENCES

[1] M. Riebisch und S. Bode, „Software-evolvability," *Informatik-Spektrum, vol. 32,* p. 339–343, 2009.

[2] T. Hannelius, M. Salmenperä und S. Kuikka, „Roadmap to adopting OPC UA," *IEEE International Conference on Industrial Informatics (INDIN),* 2008.

[3] W. Mahnke, S.-H. Leitner und M. Damm, OPC Unified Architecture, Springer, 2009.

[4] J. Lange, F. Iwanitz und T. J. Burke, OPC - Von Data Access bis Unified Architecture, Berlin: VDE Verlag GmbH, 2010.

[5] H. Züllighoven, Object-oriented construction handbook - developing application-oriented software with the tools and materials approach, dpunkt.verlag, 2005.

[6] M. R. Endsley und E. S. Connors, „Situation Awareness: State of the Art," *Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century,* 2008.

[7] M. M. Lehman, „Programs, life cycles, and laws of software evolution," *Proceedings of the IEEE Special Issue on Software Engineering, vol. 68 (9),* pp. 1060-1076, 1980.

[8] V. V. Tan und M.-J. Yi, „Development of an OPC Client-Server Framework for Monitoring and Control Systems," *Journal of Information Processing Systems, Vol.7, No.2,* 2011.

[9] A. Gırbea, S. Nechifor, F. Sisak und L. Perniu, „Design and implementation of an OLE for process control unified architecture aggregating server for a group of flexible manufacturing systems," *IET Software vol. 5 (4),* pp. 406 - 414, 2011.

[10] L. Urbas, S. Hennig, H. Hager, F. Doherr und A. Braune, „Towards context adaptive HMIs in process industries," *9th IEEE International Conference on Industrial Informatics,* pp. 244 - 249, 2011.

[11] S. Schmitz und U. Epple, „Automatisierte Projektierung von HMI-Oberflächen," *VDI-Berichte Nr. 1980,* pp. 1-12, 2007.

[12] M. Schleipen, „OPC UA supporting the automated engineering of production monitoring and control systems," *IEEE International Conference on Emerging Technologies and Factory Automation,* 2008.

[13] R. van der Geest, H. Hofmann, E. Jellum, Z. Korendo, R. Martinez, M. Orkisz, C. Zeidler, J. S. Andersson und L. G. Bratthall, „Integrating Hundred's of Products through One Architecture- The Industrial IT architecture," *Proceedings of the 24th International Conference on Software Engineering (ICSE),* 2002.

[14] B. R. Hollifield und E. Habibi, Alarm Management: Seven Effective Methods for Optimum Performance, ISA, 2007.

[15] „Alarm Systems A Guide to Design, Management and Procurement," *THE ENGINEERING EQUIPMENT AND MATERIALS USERS ASSOCIATION, PUBLICATION No 191, Edition 2,* 2007.

[16] K. F. Emigholz, „Improving the operator's capabilities during abnormal operations; observations from the control house," *Process Safety Progress,* Bd. 3, 1996.

[17] F. Nachreiner, P. Nickel und I. Meyer, „Human factors in process control systems: The design of human–machine interfaces," *Safety Science 44,* pp. 5-26, 2006.

[18] M. H. Schwarz und J. Börcsök, „An Investigation and Survey on the Future Direction of OLE for Process Control (OPC)," *INTERNATIONAL JOURNAL OF COMPUTERS Issue 4, Volume 1,* 2007.

[19] M. Schleipen, R. Drath und O. Sauer, „The system-independent data exchange format CAEX for supporting an automatic configuration of a production monitoring and control system," *IEEE International Symposium on Industrial Electronics,* 2008.

[20] A. Doan, N. F. Noy und A. Y. Halevy, „Introduction to the Special Issue on Semantic Integration," *SIGMOD Record,* 2004.

[21] D. Karagiannis, H.-G. Fill, P. Höfferer und M. Nemetz, „Metamodeling: Some Application Areas in Information Systems," *Information Systems and e-Business Technologies,* pp. 175-188, 2008.

[22] E. J. Evans, Domain-Driven Design: Tackling Complexity in the Heart of Software, Amsterdam: Addison-Wesley Longman, 2003.

[23] W. Mahnke, A. Gössling, M. Graube, und L. Urbas, „Information Modeling for Middleware in Automation," *IEEE 16th Conference on Emerging Technologies & Factory Automation,* 2011.

[24] E. Rahm und P. A. Bernstein, „A survey of approaches to automatic schema matching," *The VLDB Journal 10,* p. 334–350, 2001.