Standing on the Shoulders of Giants: A Vision for Data Standards for Hydroelectric Project Management

Alan Hodgkinson	**Joseph J Kaelin**	**Martin Pike**
SoftXS GmbH	Pöyry Infra Ltd.	Qoqnis Ltd.
Alpensrasse 14	Hardturmstr. 161	7 Crown Glass Place
CH-6300 Zug	CH-8037 Zurich	Nailsea, BS48 1RD
Switzerland	Switzerland	United Kingdom

1. Introduction

As an industry we are not addressing the challenges brought about by the widespread adoption of Information Technologies (IT) for managing Hydroelectric Project documentation and data. The current use of IT systems on hydroelectric projects is eroding project controls, and in many cases is the cause of inefficiencies, which lead to increased project costs and risks. In terms of the IT system we use, we are currently in a transitional period where IT systems for managing project documentation and data are essential, but are not fully matured and are unable to fulfill the basic industry requirements.

Other industries have faced similar challenges and their experiences can provide guidance. In order to apply their lessons we must understand the different basic characteristics of our respective industries and working practices.

The authors propose developing industry standards to ensure that critical project knowledge can be properly managed and maintained, and that electronic systems be defined that are able exchange documents and data, based on these standards. These standards should be based on the engineering and construction practices that have evolved over centuries of experience, rather than imposed from outside by software systems designed by outsiders.

2. Long-Lifetime Assets vs. Short-Lifetime Systems

The hydroelectric plants being built today will outlive the people and systems that create them. A hydroelectric plant typically has an expected lifetime of a century. This exceeds most people's lifespan and far surpasses the expected lifespan of today's computer systems and software.

The knowledge produced during the design and construction phases of a hydroelectric facility is essential for its long-term operation and maintenance. Considering that the first substantial maintenance might be performed decades after construction is completed, it is important to ensure that the knowledge from the construction is preserved in a form that can be understood by the next generation of engineers and systems.

In the previous age of paper drawings, while there was the inconvenience and tedium of searching physical archives, there was a certainty that once you found a document, you would actually be able to read it. In today's digital world this is no longer guaranteed. In the world of electronic documents, document searching has been vastly improved, but there is no longer a guarantee your software will be able to understand the documents and data you find.

Software systems and file formats are constantly changing and we are frequently confronted with new versions of software applications that are unable to properly read files created by older versions. This is already occurring during the lifetime of typical projects and we can only expect the problem to get worse as time progresses. Given the long lifetime of hydroelectric plants, this is a serious issue.

A further challenge for hydroelectric projects is collecting the complete design and construction knowledge in the first place. The globalization of projects has compelled us to embrace electronic communication. Virtually all project documents are created and stored in electronic form, usually by the document's author, who often sends them directly to the recipients. This convenience has allowed the secretarial and clerical staff of projects to be substantially reduced. Unfortunately, this has been accompanied by a decrease in the capabilities available to create and maintain a complete master project archive. The project managers and engineers have taken up some of the clerical duties, such as word processing and sending emails, but not the duties required to create and maintain a comprehensive project archive.

In response, electronic Document Management Systems (DMS) are being deployed on most projects. In fact, many projects have multiple systems. These systems are often incompatible and ill suited to the basic needs of our industry. There are a variety of reasons why this is so, which are described in the following sections.

3. Hydroelectric Projects are Unique

Large infrastructure projects, specifically hydroelectric projects, are a unique industry that is very different from most other commercial enterprises:

1. Projects are run by collections of companies. Overall responsibility is divided amongst the participating companies according to their roles and the overall contract type. Often there is no single company that exercises complete control. Conflicting loyalties to the company or the project further complicate the situation.
2. The coordination between the multiple organizations, teams and locations is complex and is typically communicated via documentation and data subject to well-defined (and strict) industry procedures.
3. No two projects are the same. Each project has different topology, geology and a different collection of companies.
4. Due to the uncertainties of geology, projects must frequently revise their design and program as the project progresses. The multi-company nature of projects means that the resulting effort of coordinating changes is a significant management challenge.
5. Projects have a limited lifespan and the participating companies typically do not have a permanent commitment beyond the handover to the owner.
6. Projects have a long enough lifespan, often a decade or more, such that there is a real danger of knowledge and expertise being lost during the project. This can be due to partner companies that are only associated with the project for a short period or normal personnel changes during the project.

4. Impedance Mismatch: IT Vendors and Hydroelectric Projects

Today's document management systems evolved from content management systems (CMS) intended for deployment within a single company. CMSs were originally intended for managing correspondence, sales literature, product descriptions, catalogs, price lists and such like. The main requirement was the ability to save and later search for documents. Search facilities were usually limited to keywords, title and text search. Workflow abilities were primitive (compared to construction), reflecting non-rigorous practices commonly found in commercial enterprises. Often a CMS was deployed for a single department and operated autonomously from other CMSs.

Content management systems then evolved into *enterprise* content management systems, which were capable of sharing to a larger audience, spread over multiple locations, but still with all users belonging to the same organization. Enterprise CMSs generally have improved search capabilities.

Various adaptations have since been made to make CMSs attractive for use on construction projects. Software vendors are now attempting to sell to engineering and construction companies. The point of entry is usually the IT department, and the intent is that the systems are sold and deployed as an *enterprise* CMS available to all employees. This fits well with corporate purchasing practices, but fails to address the needs of large multi-company projects.

Unfortunately, most software vendors do not understand how large construction projects really work. There are two main misunderstandings:

1. There is a well-established (and relatively complex) set of industry practices, mainly expressed as workflows, for managing engineering and construction documentation and data
2. Large construction projects are temporary entities composed of many enterprises. This has implications for:
    a. The sharing of project data to outside companies, while not sharing internal company information
    b. The willingness of employees of external project partner companies to install new PC applications in order to access the CMS (which may even be contrary to company policy)
    c. The willingness of employees of external project partner companies to learn project-specific software systems

The current state of the art is that we must deploy multiple document and data management systems, many of which are ill suited to the task they perform and unable to communicate with other systems. Access to some systems is restricted to the company in which they are deployed, making it difficult to share documents and data with other project participants.

5. Brief Overview of Computerization in Construction

The first use of computers in construction (excluding early accounting applications), started in the sixties, and was the use of FORTRAN programs for performing engineering calculations. These programs were often created by engineers themselves, or in the engineering departments of universities and research groups. The development of engineering calculation programs continued through the seventies and eighties, by which time extensive libraries of programs had been created, many of which (sometimes translated into more modern computer languages) are still in use today.

While engineering calculation programs represented a major step forward for creating and confirming the design of complex structures, they had little effect on the existing engineering and construction management processes. Engineering and construction practices remained in place, more or less unchanged. It is of note that the development of engineering calculation programs was driven by the construction industry itself.

Starting in the late seventies and eighties the first Computer Aided Design (CAD) software appeared. By the nineties the use of CAD systems for producing engineering drawings was widespread and the equipment used for producing paper drawings was rapidly disappearing from design offices. The activities of a draftsman changed completely and the job title '*CAD operator*' was created. The storage of drawings also changed. Previously all drawings were stored in archives of physical files. With the introduction CAD systems, digital files could be stored instead. In most design offices, electronic drawings were stored in file servers on local area networks, which had recently become available. The digital files produced by CAD systems were all in proprietary formats and there were only limited capabilities for exchanging drawing information with other construction software systems.

In spite of the ability to create digital drawings, their delivery largely remained in paper form. This was for a number of reasons:

1. Electronic delivery was difficult due to the limitations of the networks then available
2. Most recipients (contractors at job sites), lacked the technical knowhow and infrastructure required to receive and print electronic drawings
3. Many project partners preferred deliveries of paper documentation. E.g. they were unwilling to accept changes to existing practices

CAD systems significantly changed drawing production, but did not have a major effect on other project areas. Drawing checking, approval and distribution systems survived virtually unchanged. Even today, a title block and revision list found on a drawing would easily be understood engineers from previous generations. CAD drawing production, like engineering calculation programs, did not disrupt construction practices.

Project scheduling software was developed on mainframe computers starting in the seventies for the aerospace industry, with use spreading to large construction projects in the eighties. Early use of scheduling software was quite rigorous, making use of CPM (Critical Path Method) and PERT (Program Evaluation & Review Technique) techniques together with other industry specific procedures. This has since remained a specialty activity performed by qualified scheduling engineers. Scheduling software uses proprietary data formats that make data exchange with other construction software difficult. With the arrival of the PC, spreadsheet and other desktop software have however disrupted professional scheduling and scheduling is now often performed without adherence to professional scheduling standards.

Calculation programs, CAD systems and scheduling systems co-existed with the current largely paper-based control systems, which were generally considered authoritative. This caused very little change to existing project practices.

In the late nineties a number of factors came together, which caused a major disruption in the construction industry:

1. Globalization with many project partners in distant locations
2. Widespread availability of inexpensive Personal Computers (PCs) and word processing programs
3. Widespread availability of networking, email and the World Wide Web

The convenience offered by widespread networking, meaning that anyone could instantly send a document to someone anywhere in the world, quickly became essential in the rapidly globalizing world. In fact, digital communications accelerated globalization. Construction professionals were now able to write and distribute documents themselves, without having to depend on centralized support staff. Communication and documentation about project issues moved from paper correspondence to email, increasing the speed and convenience in which issues could be discussed and resolved.

This convenience had a cost. It became nearly impossible to find a project still maintaining a unified and comprehensive project archive, containing all project documentation, emails and data.

In response, projects turned to DMSs adapted from other industries, as described earlier. The result is generally sub-optimal, causing either inefficiencies due to their poor 'fit' or a relaxation of project controls, or more typically, both.

The previous generation imposed rigorous procedures for managing and filing paper documentation. This rigor is now disappearing because the production and filing systems for emails and electronic documents currently in use are incapable of imposing the strict controls that were previously universal. The rapid evolution of software systems (driven in part by the electronics and software industries) has exacerbated the situation. In particular:

1. Project controls have been weakened
2. It is difficult to determine which data and documents are authoritative
3. Project partners to not have easy access to all the information required to perform their duties

The problem is that the cart is leading the horse. We are, in some cases unconsciously, adapting our working methods to fit the limited set tools available from software vendors. Rather than proactively defining what we will accept, we are embracing practices imposed by software systems created by outsiders from other industries.

The result is a radical departure from the traditional paper-based processes that were well-known and accepted.

- ***The computer and digital communication revolution has caused a major disruption to the construction industry. This disruption has occurred due to failures to adequately respond to pressures from outside the industry.***

6. Experiences from Other Industries

Other industries have faced and overcome the challenges of adopting computerized systems and can provide guidance for our own industry. The following sections describe how the banking, aerospace and the high-energy physics research community have adopted IT systems.

6.1 Banking

Banking was one of the earliest adopters of IT systems and is also one of the largest employers of software professionals. Banks originally created internal IT systems on mainframe computers to manage their bank accounts. Later systems managing *bank-to-bank* transactions (payments and currency exchanges) and *bank-to-third party* transactions (stock exchanges, credit card companies and intermediaries) were created. These systems evolved over many decades, often at great cost.

The success of the baking industry IT systems is based on the following principals:

1. Business processes not software vendors drive software system definition (all early banking systems were implemented by the banks themselves)
2. Build or buy small mission specific systems that perform limited but well-defined tasks
3. Recognize that **all** IT systems must exchange data with other systems
4. Establish standards for data formats and protocols for exchanging data. Examples include *SWIFT* for interbank transfers and *FIXX* for exchange trading
5. Develop the IT infrastructure incrementally, based on components, which can later be replaced individually as requirements change
6. Reduce risk by introducing system components gradually, which also allows the people-based processes to be gradually adapted to the improved systems

Initially effort was spent getting internal IT systems to exchange data. The resulting savings were enough to overcome inter-company rivalries and allowed industry standards to evolve, enabling automated data exchange between banks and other financial institutions. This is now advanced to a state where software vendors are forced to produce systems that conform to banking standards.

A key point is the step-wise approach to developing standards and componentized systems. In the short-term this causes minimal disturbance to existing processes, increasing the acceptance by the personnel using the systems. In the longer-term it has led to a major cultural change within the banks and to a revolution in the banking industry.

6.2 Aerospace

Modern aircraft (and spaceships) are extremely complex and are literally composed of millions of components, all of which must adhere to strict specifications and function together. One of aerospace's main IT challenges was the capture of all knowledge about an aircraft's design, all the components necessary for its construction and all information associated with its servicing.

The aircraft industry is different from construction:

1. Multiple similar aircraft are produced
2. Aircraft manufacturers are the dominate partner in business relationships and are able to dictate their business processes and data exchange requirements to their suppliers and partner companies

Like banking, the aerospace industry has created a complete IT ecosystem, defined mainly via standards, which strictly define the data formats used for exchanging data. Given the less structured nature of aerospace data (as compared to financial records), most aerospace data is stored in using *Structured Data Formats*, which allow data to be componentized, and then later aggregated into different formats. *SGML* and *XML* are examples the most important structured data formats. The main advantage of this approach is that it separates the storage of the information from its presentation, allowing the same source information to be used for different purposes.

The main features of Aerospace IT include:

1. Aerospace was an early adopter of structured data formats and took an active role in defining how aircraft knowledge should be represented using *SGML* and *XML*
2. Using structured data formats as the *lingua franca* for communication between data systems allowed development of individual software components specialized for narrowly defined tasks
3. Many data specifications were originally (US) military specifications adapted for civil use. One important example is S1000D, which defines procurement and production of technical documentation
4. Individual data components are typically stored in specially customized content management systems, which are capable of extracting particular subsets of the data components for different publishing purposes
5. A series of specifications were developed to enable the automatic production of documentation from the extracted data components

An example of automatic document generation from individual components is *Interactive Electronic Technical Manuals* (IETMs). These are electronic documents stored online or in portable devices, often used by servicing personnel where it is impractical or impossible to carry the physical paper versions.

Like banking, the aerospace industry's development of standards has forced software vendors to produce systems that conform to those standards. For example, there are over ten different software companies producing IETM systems.

6.3 High-Energy Physics

In the mid-eighties Tim Berners-Lee of CERN, a particle physics research facility in Geneva, proposed a system for interchanging scientific information between research institutions located all over the world. Most academic and research institutions had access to wide-area networks, but the mechanics of transferring research results were inconvenient and frequently required specialist knowledge.

Berners-Lee's vision was to change from a *'file transfer'* methodology, where specific information was sent like a letter to a specific location, to a *'publish'* methodology, where large amounts of information would be made publicly available for anyone to access.

Unable to get funding or find software vendors willing to build systems to realize his vision, Berners-Lee defined the protocols and data standards himself and together with a small group of colleagues implemented an early implementation of what became today's World Wide Web. The result was popular among scientists. By word of mouth, use of the Web slowly spread to larger and larger audiences.

By the mid-nineties many individuals and informal groups started producing 'open source' implementations of the new 'Web standards' that were available to anyone free of change. This caused software companies, notably Netscape, to take interest and start producing commercial web browsers and web servers. By the end of the nineties the public was actively 'surfing the Net'. By the turn of the millennium web addresses were found on all business cards and advertisements.

Why did the Internet succeed?

1. The *World-Wide-Web* was essentially a research project, which was designed and implemented simultaneously. Prototypes were created, tested, and corrected in very short cycles. The resulting standards emerged as a result.
2. The early world *World-Wide-Web* depended on only two *de-facto* standards:
    a. A transfer protocol: **Hyper-Text Transfer Protocol (*HTTP)***

    b. A data standard: **Hyper-Text-Markup-Language (*HTML)***. HTML is a simplification of *Standard Generalized Markup Language (SGML)*, an overly complex and theoretical standard defined by an ISO committee, which had languished for years due to the technical difficulty of implementing it
3. HTTP and HTML were extremely simple to implement. In fact, they allowed useful systems to be built using only sub-sets of the standards
4. Anyone who writes software that implements the *HTTP* and *HTML* standards can exchange data with all other programs. This allows for multiple implementations, optimized for different needs, all of which are able exchange information
5. The standards were simple enough yet complete enough to allow programs to communicate with other programs. This allowed complete web-based software eco-systems to be developed
6. The definitions of HTTP and HTML were available to the public free of change. There were no trade secrets, copyrights or intellectual property issues. This meant anyone and everyone could take part, which was key to the Internet's early growth

It is important to note that the Internet revolution was *started by individuals representing end-users* and by not software companies. Microsoft, the world's largest software vendor, was largely unaware of the Internet until it nearly overwhelmed them.

Left to software vendors, the Internet would have remained a collection of 'closed gardens' (Microsoft's *MSN Network* and *America Online* were the largest networks at the time), with only limited abilities to communicate between them. As a comparison, image a world in which you could only telephone people who use the same provider as you. Having to prevent such limitations demonstrates the importance of having industry needs dictate software capabilities.

7. Recommendations

The most import lesson from the other industries is:

- ***We must take an active role in the definition of the software systems we use***

Document and data management is not an end in itself. It exists only to support efficient and accurate project work. The immediate goal is to make document and data management fit our needs in the most painless way possible. Improved IT systems can make processes efficient and accurate, but it is up to us to explain exactly what we need.

The construction and documentation management procedures, that we are all familiar with, define our industry's best practices and consequently must define the requirements of the software systems intended to support them.

The construction industries basic requirements for IT systems that manage data and documents are relatively simple:

1. Quick and efficient access to all information by anyone, located anywhere
2. Authoritative documents and data must unambiguously identified
3. Software systems must implement procedures (workflows) defined by industry best practices and the project handbook
4. Software systems must be adapted to the way industry practitioners actually work

Engineers are by nature tool builders and thanks to their analytical abilities are well positioned to participate in the definition and development of the next generation software tools. Indeed this has frequently happened in the past, often causing headaches because little thought was given to sharing the resulting data with others. This is not the fault of the individuals involved, but of the industry, which to date has not provided any guidelines for defining data formats for sharing data, nor explained the importance of doing so.

It is essential that industry experts (e.g. engineers and project managers) and not software companies perform the requirements analysis, simply because the requirements and standards have more to do with industry practice than software systems. Once defined, requirements can be given to software professionals for implementation.

Unfortunately, software development is a notoriously unpredictable activity. Unlike construction, no one can easily see and evaluate partially built software systems. To prevent unpleasant surprises it is absolutely necessary to keep software practitioners on a short leash. Use short (iterative) definition and development cycles, and make use of quick feedback from the end-users, to prevent software projects from going astray.

Further details:

1. Use engineering vocabulary and terminology and not IT jargon to define software capabilities
2. Build a component-based software infrastructure, using components that are able to exchange information
3. Break engineering and construction processes into discrete steps and build a series of individual components, which can be '*chained*' together, to perform them
4. Focus on the '*what*' not the '*how*'. Define the intended capabilities of the components and the information they consume and produce, rather than their internal processing details. Unfortunately this can be difficult because it is contrary to the way most engineers think
5. Define protocols and data formats for exchanging information between the components. This can initially be done as guidelines and then later via formal standards as experience and consensus evolves
6. Build small components where possible. Smaller components can be built more quickly, allowing for experimentation and for a *best of breed* to evolve
7. Design and build systems as you go. Multiple quick experiments are frequently faster and less risky than long design and implementation phases
8. Add components incrementally. This causes less disturbance to existing working practices

8. Conclusion

Project management ultimately depends on efficient access to accurate documents and data for timely and accurate decision-making, just as all project members depend on it for carrying out their own work. IT systems can be used to collect, store, organize and distribute this information, provided that we are able to coherently describe our needs to those that create the software systems.

Our industry has a rare opportunity to take advantage of being late-adopters of Internet technologies. We can learn from other's experience. By applying the lessons learned and recognizing the unique characteristics of our own industry, we will be able to create IT systems that fulfill our requirements and 'fit' our projects. The project-based nature of hydroelectric projects will make this a challenge, but it will also provide opportunities because it allows for cycles of experimentation and fine-tuning, accelerating the convergence to an optimal solution. However we must take an active role to ensure the process really delivers what we need.

As engineers, we have the abilities required to analyze and express our requirements. The combination of our industry's best practices and project handbooks should already contain all the required inputs. We should carefully review our project handbooks to ensure that all authoritative data is unambiguously defined, along with the standards and processes associated ensuring documentation and data quality. This should be the baseline from which we build our IT systems.

The existing state of our documentation and data systems, a mix of paper and electronic systems that have limited inter-communication abilities, will make the transition a challenge. The key is to proceed on a component-by-component basis, fixing one area at a time and disrupting the current 'people-processes' a little as possible.

We are not yet burdened with enormous legacy systems and still have a green-field in which to develop new standards. However, as companies continue to deploy computerized systems, the difficulty of getting broad industry acceptance will increase.

The time to act is now.

The Authors

Alan Hodgkinson graduated with ScB degrees in Electrical Engineering and Computer Science from the Massachusetts Institute of Technology. He is the general manager of SoftXS GmbH, a company specializing in document and data management for large construction projects. Email: alan@softxs.ch. Tel: +41-41-511-9776.

Joseph J Kaelin graduated with an ScM degree in Civil Engineering from the Massachusetts Institute of Technology. He is a senior tunnel engineer and senior project manager at Pöyry Infra Ltd, and has worked on hydroelectric projects for over three decades. Email: joseph.kaelin@poyry.com. Tel: +41-76-356-2050.

Martin Pike graduated from King's College London with a BSc in Biology and Physics, and has nearly 30 years experience building large software systems in many industries. He has been involved in the development of a number of standards initiatives, including MathML, XML and OWL. He is CEO of Qognis Ltd, which develops large-scale distributed database technology. Qognis Ltd, 7 Crown Glass Place, Nailsea, BS48 1RD, UK. Email: qognis@gmail.com. Tel: +44-1275-866-384**.**