WM2008 Conference, February 24 -28, 2008, Phoenix, AZ
Abstract #8145

Advancing Information Technology in the Waste Management World

B. Slater, G. Smylie, S. Thompson, H. Bruemmer
Insei Software Engineering Services
2235 E. 25th St., Idaho Falls, ID 83404

ABSTRACT

The development and utilization of software for the waste management world is critical, yet complex. Numerous and sometimes conflicting regulations, coupled with demands for streamlined efficiency and high standards of safety, require innovative information technology solutions and closely-managed processes. The primary goal of this paper is to demonstrate how this challenge can be met by applying software engineering best practices to the waste management domain. This paper presents two case studies highlighting how IEEE (Institute of Electrical and Electronics Engineers) software engineering standards have proven to be effective within the CH-TRU and RH-TRU waste management arena. These examples show how adherence to best practices has enabled software to meet institutional expectations for usability, consistency, reusability, documentation, quality assurance, and adherence to regulations. Specific techniques, such as the use of customizable software lifecycle management software, and the integration of subject matter experts and the information technology specialists through the change control board, will be presented in detail. With an eye to the future, we will show the software resulting from a best practices approach can be further enhanced with the use of artificial intelligence techniques to tackle problems such as accounting for unexpected user inputs, analyzing the relationship between data fields, and recognizing aberrant patterns in the data.

INTRODUCTION

The process for supervising, validating, and controlling data related to waste management processes has become increasingly more important as regulatory requirements have become more prominent and complex at the same time. The main stream thought to handling the vast data requirements has been to transition to Information Technology (IT) based solutions to facilitate these efforts. IT solutions represent a logical choice for effectively streamlining, consolidating, and validating waste management processes. While the use of IT has greatly benefited the waste management world, as with any business solution, improper development and usage of new business tools does not always adequately address the specific needs nor address the expected cost savings and functionality that were originally intended with the efforts. For IT solutions to have the greatest level of benefit, a well defined and stringent process must be followed that brings together the necessary resources and expertise to guarantee that all the functional requirements meet the approval of relevant stakeholders. One main goal of this paper is to give detailed examples of how IT solutions developed under “best practice” methodologies

have been able to successfully automate waste management processes. In addition, attention will be given to existing technologies that could further drive and enhance the waste management industry.

LIST OF ACRONYMS

ASME	American Society of Mechanical Engineers
CH-TRU	Contact Handled Transuranic
CCB	Change Control Board
CTMA	CH-TRUCON Management Application
IEEE	Institute of Electrical and Electronics Engineers
RH- TRU	Remote Handled Transuranic
RTMA	RH-TRUCON Management Application
TRAMPAC	TRUPACT Authorized Methods for Payload Control
WIPP	Waste Isolation Pilot Plant
WWIS	WIPP Waste Information System

BEST PRACTICES

Best practices can best be defined as the most efficient and effective way of accomplishing a given task using a repeatable process that has proven successful for one or more groups of individuals. Software engineering best practices are subjective, and if defining these practices is left to disparate groups of software professionals, process inconsistencies in design, development, verification and validation of software products from team-to-team or project-to-project abound.

Research conducted by the Standish Group, a firm that focuses on assessing risk, cost, return and value for Information Technology (IT) investments, has proven that most successful software projects – projects completed on time, on budget, and with all the features and functions originally specified – follow standard processes and utilize tools that monitor and control progress.

The de-facto standards for general best practices as related to software application development are compiled by the IEEE Computer Society Software Engineering Standards Committee (SESC). One of the SESC's primary missions is to develop an integrated family of standards that respond effectively to user needs. Adoption of the SESC practices removes inconsistencies across development teams and projects that would otherwise be introduced.

The majority of the SESC standards are practice standards rather than product standards, focused on the conduct of software engineering practices instead of specific standards for the developed end-product software. These standards address fundamental software engineering principles such as:

- Invest in the understanding of the problem.
- Since change is inherent to software, plan for it and manage it.
- Uncertainty is unavoidable in software engineering. Identify and manage it.
- Since tradeoffs are inherent to software engineering, make them explicit and document them.
- To improve design, study previous solutions to similar problems.
- Define software artifacts rigorously.
- Establish a software process that provides flexibility.
- Manage quality throughout the life cycle as formally as possible.
- Implement a disciplined approach and improve it continuously.
- Build with and for reuse.

The following flow chart provides a high level overview of the software development process utilizing IEEE standards. This process addresses the principles previously outlined and provides for full accountability and control of change, error and enhancement identification and resolution, and complete traceability of development and verification activities.

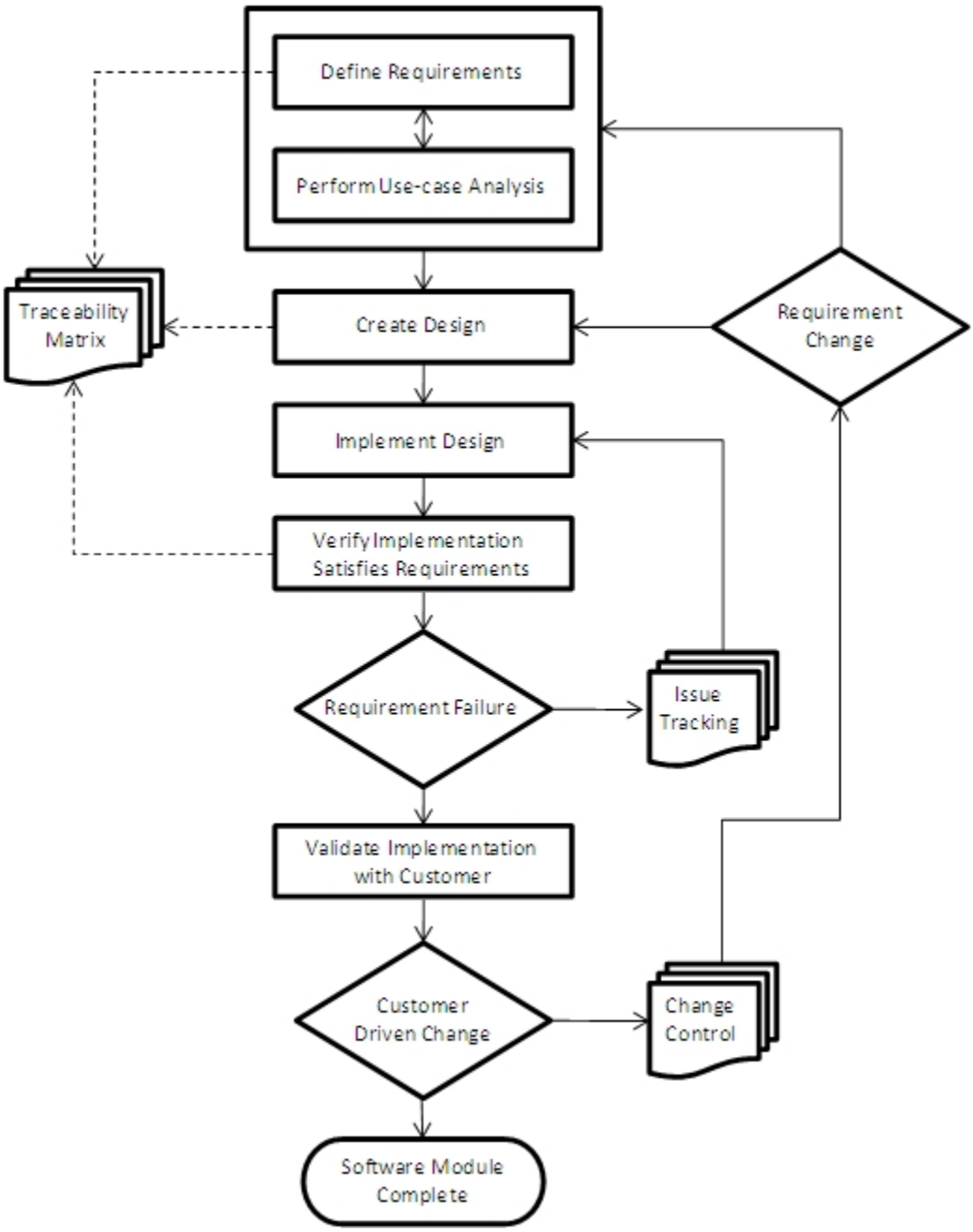

Fig. 1 – Best Practices Software Development Process

In the Department of Energy's (DOE) waste management environment where contracting organizations often provide services at multiple sites and specific regulations apply complex-wide, adhering to proven best practices defined by industry recognized standards can result in significant cost savings for software projects and help prevent software delivery schedule overruns.

CASE STUDIES

TRAMPAC Evaluation Software

In the spring of 2004 the Waste Isolation Pilot Plant (WIPP) commissioned a new effort to enact widespread updates to the existing eTRAMPAC software. The existing eTRAMPAC enforced the rules and regulations contained with the CH-TRAMPAC documentation. Many factors and objectives drove the need to perform the required updates to the software. The objectives for the project included the need to conform to the pending revision updates to the CH TRAMPAC, facilitate greater ease in maintaining the software package, add database driven checks, and implement usage of object-oriented software architecture.

After careful consideration and a thorough review and approval process, Insei was selected to perform the required upgrades to the existing software package. After a full evaluation of the existing code base, Insei recommended that a complete rewrite be performed in order to best reach the objectives of the project. With that decision, the full software development life cycle process began to create an updated version of the eTRAMPAC software now known as the CH-TRAMPAC Evaluation Software (CHTES).

The CHTES software would require a full team of experts with a wide range of skills and knowledge to adequately address the full development of the software. Included on the project team were project managers, quality assurance reviewers, software engineers, database administrators, software analysts, mathematicians, CH TRAMPAC subject matter experts, and Transportation Certification Officers (TCOs). In a best practice scenario, input and validation would be required from relevant stakeholders to produce software that would meet their stringent needs and expectations and likewise conform to ASME NQA 2 standards.

The first step in the development cycle was to create an IEEE compliant Software Requirements Specification (SRS). The SRS was initially developed based on face to face requirement gathering meetings with CH TRAMPAC SMEs and other WIPP Waste Information System (WWIS) WWIS experts. Requirements also were directly extracted from other SMEs and collaborating regulatory and technical documentation. All software requirements were directly traceable to a source in the form of a specific document citation, person, or Change Control Board (CCB) decision. The Change Control Board is a steering committee comprised of relevant project stakeholders and given the charter to maintain the correctness of the project. The final SRS underwent internal review and subsequently was approved by the full CCB. Through this process the project was able to validate in the early stages that the eventually developed software would meet all the necessary functional, transportation, and regulatory requirements. Modifications to the SRS from that point forward were only made through a structured change control process with final approval by the CCB after formal review performed by SMEs, technical leadership, project management, and other stakeholders.

An important aspect of the SRS is that each requirement contained within met all applicable IEEE standards for software requirement specification where each individual requirement was clear, concise, complete, correct, traceable, and testable. Each requirement was individually reviewed to ensure that the above criteria were successfully met prior to submittal to the CCB.

Following development of the requirements, a full design phase was performed to address the technology needs of the project. The design phase produced a data definition model (DDM) that ensures that the software engineers have adequately addressed and understand the data requirements for performing transportation validations. Next, an IEEE compliant Software Design Description (SDD) that defined each system component, its intended usage, integration points, and interfaces was created. As a final step of the SDD process a mapping was created between the requirements and design elements to validate the system design fully addressed all specified requirements. The SDD and its accompanying system architecture was reviewed by a team of WWIS developers and database administrators to guarantee that the intended software would integrate seamlessly into its parent system and that the functional interfaces between the two sets of software were correctly indentified and understood.

The final phase of the project was to perform the implementation and validation of the software. Each software engineer was assigned a specific set of design elements and associated requirements to create as part of an individual software unit. Once the software unit was completed and passed developer testing, it was assigned to an analyst for functional validation against the approved software requirements. After individual validation of software units, all software units underwent integration testing to authenticate their usage as a single application. Finally, the CHTES software underwent another set of integration tests as it was used in conjunction with the WWIS application. A key step in the validation of the software was the mandated acceptance testing performed by project stakeholders. During the acceptance testing, project stakeholders in the form of SMEs were able to test the usage of the software and certify that it met the specified requirements and would perform the necessary functions for properly authorized TRU waste shipments.

The CHTES project is great example of how, through well defined collaboration and documented efforts, a wide range of stakeholders were able to work seamlessly together to create a fully functional system for enforcing a complex and prominent set of software requirements, and simultaneously facilitate the ability for TRU waste shipments to be received by the Waste Isolation Pilot Plant.

In subsequent years the above process was reproduced for creating the RH TRAMPAC Evaluation Software (RHTES) for the approval of RH based TRU waste shipments to the WIPP facility. Based on the success of the CHTES software and it reproducible knowledge and processes, the RHTES was created in fraction of the time and cost in comparison to the CHTES software.

RH TRAMPAC Management Application

With the increasing complexity of TRAMPAC regulations and the associated data required by validation software such as the CHTES and RHTES, the need arose for streamlined and automated methods for properly maintaining the data sets that are used by the software. Vital to the need was the concept of retaining the integrity and correctness of data sets that are used to approve or reject waste shipments, while at the same time retaining the ability to modify and add to the data sets as required by revisions to the TRAMPAC software and the individual and changing needs of the shipper/generator sites.

As a potential solution to this problem, it was proposed that instead of retaining a manual process for recording TRAMPAC evaluation data, software could be developed that would facilitate the data maintenance process. As a result, the compliance evaluation data relating to RH waste evaluations was targeted for software automation. As the concept of using software tools for data maintenance was being developed, a secondary usage for the software quickly evolved. The software, through an identical interface, could provide packaging engineers and TCOs with the ability to perform "trial and error" calculations relating to RH TRUCON codes. The software would not only have the ability to record RH TRUCON validation criteria but would also have the ability to directly calculate decay heat and flammable gas generation rate (FGGR) limits based on user inputs. The resulting software would offer the capability for users to quickly generate decay heat and FGGR limits using proven, repeatable methods that utilized waste composition and payload configuration criteria supplied by users. Instead of using multiple sources and resources for the generation of new RH TRUCON codes, the software would offer a one stop solution for generating and testing compliance data specifications.

As a result to the identified needs, and in conjunction with the development of the RHTES software, the RH TRAMPAC Management Application (RTMA) was developed. Using the same business and development processes under "best practice" standards, the RTMA quickly became a successful solution for users to view, create, review, and test RH TRUCON compliance data. The RTMA application provides a web-based interface that offers the ability for TRU waste shipper/generators, packaging and transportation engineers, and data administrators to collaborate and review RH TRUCON related data.

Once again the above process was reproduced to create a counterpart to the RTMA application that managed the regulatory data related the CHTES software. As a result, the CH-TRAMPAC Maintenance Application (CTMA) was created. Due to the success in using the RTMA and CTMA application for maintaining the data referenced in the accompanying TRUCON documents the applications are now being targeted for designation as the source control for all TRUCON related data. In this case, it was seen that best practice enabled software became the

clear choice for streamlining and dictating the business process pertaining to the maintenance and approval of both CH and RH TRUCON codes and its associated documentation.

WHAT BEST PRACTICES HOLD FOR THE FUTURE

The case studies described previously have benefitted greatly from the application of "best-practices" to their associated software lifecycles. The primary result of this is high quality software and a maintenance cycle that responds with precision, and if necessary, rapidity, to the changing needs of the users and owners. While this has so far contributed to a cohesive team with great confidence in the product, a best practice methodology also holds promise for improving the processes that are external to the software, and for discovering new questions, and possibly answers, hidden within the data being collected.

Since application software is simply the computerization of what would otherwise be a manual process, similar best practice techniques could also be used to improve those processes that are, perhaps for the time being, still manually performed. If best practice techniques can lead to a highly reliable and reproducible system such as the software described in the case studies, they are also suitable for documenting and testing manual processes that require a high degree of reliability.

The Input End - Improving Data Collection

An example of how "best practices" could be used comes from processes that are external to the CHTES, but generate data which are ultimately transmitted to that system. Many of these processes are waste characterization procedures, such as headspace sampling for flammable volatile organic compounds. Other processes test the characterization equipment, and yet others validate the measured results of both test runs and actual sampling events. The complete set of these related processes could be modeled similarly to a software project. Each individual process would be analyzed as a subsystem of the project, with its own requirements, test cases, and error-reporting documentation. A governing body similar to the CCB would help steer the direction of the overall project, as well as reach consensus on the decisions affecting the individual processes. The availability of documentation to all participants helps make it an open system. This gives a greater number of knowledgeable people the ability to review the entire system and its components for consistency and accuracy. Questions or problems encountered would be documented so their solutions could be referenced and consistently applied in the future.

Since the output from a software application is only as good as its input, it makes sense to apply standards to the data-generating processes that are at least as rigorous as those to which the

software itself is held. A set of practices very similar to those used in the original software projects described in the case studies above could be used to improve the quality and institutional knowledge of the data collection program.

The Output End – Data Analysis and Problem Prevention

Many software database applications collect and store data, presumably for the lifetime of the program under which it is collected. Interestingly, much of the data that is collected is never viewed again, excepting periodic summaries, unless some error later arises that necessitates the review of this data. But if the answer to an undiscovered problem, or even a portion thereof, lies in the data awaiting only the discovery of the problem, a proactive analysis of the data ahead of time can help prevent problems in the first place.

Again using the CHTES case study as an example, many of the waste containers from a particular waste stream were generated by a single process, and therefore should have similar properties. While the existing CHTES software evaluates each incoming data record for compliance with the transportation regulation, catching data errors that do not exceed the regulatory limits is completely outside of the CHTES scope. Once a representative inventory of containers from a particular waste stream is present in the database, semi-intelligent software could generate a profile of the properties for that waste stream. Incoming data records, after approval by the CHTES, would be compared to the representative profile, and a warning issued if the new data record is significantly inconsistent with that profile. If subsequent review of the record finds an error, a potentially critical error might have been averted. If, however, the review finds that the data is correct, further inquiry could help determine whether the profile should be adjusted to accommodate the statistical outlier, or whether it was a one-off case that should still trigger a warning if it happens again.

Being a software project, it would naturally benefit from the best-practice techniques for software engineering described earlier. Additionally, however, a project such as this would also benefit greatly from the institutional knowledge that arises in a steering committee like the CCB. An individual does not typically understand the entire system well enough to know what kind of data profiles to create, or what kind of outliers to look for. The combined experiences of the diverse personnel that make up the CCB form a kind of collective intelligence. Experiments of different waste-profiling algorithms can be documented alongside other aspects of the system. Those algorithms that aren't found to be successful may later be improved with the inclusion of new data. By documenting them permanently, those unsuccessful avenues won't need to be travelled down unless something new is included to possibly improve them.

CONCLUSION

In the future the waste management industry will face many continued challenges to comply with requirements under increasingly higher levels of scrutiny and litigation while dealing with

pressure from various sources to increase efficiency and accuracy. IT solutions developed under best practices conditions offer a high return on investment and can certify to stakeholders in waste management processes that results are complete and accurate. By applying new techniques and technologies to current IT solutions, stakeholders have the opportunity to increase automation, ensure accuracy, and improved overall efficiencies leading to cost savings and increased performance overall.