Managing Software as a Discipline on 6th Gen MODU’s

Software Management of Change (SMOC)

Stephen Hadley
Kingston Systems Ltd
April 2012
Executive Summary

Modern 6th generation drilling rigs are run by software and computer systems and proper management of these systems has become a key requirement for safe operations.

Software Management of Change (SMOC) is a set of procedures and information designed to control, track, and understand changes to software systems for the purpose of increasing predictability, disaster recovery, auditability, and accountability. Unfortunately, many operators are learning about SMOC the hard way through increases in accidents, data loss, NPT, and unpredictable system behavior.

This document covers the nature of a good SMOC program, consequences of a weak SMOC program, common pitfalls, and training recommendations. The latter half of the report gives case studies to help demonstrate how SMOC policies can effect operations, and an appendix detailing the level of care and testing practiced in software management industries.

A good SMOC program consists of the following:

- A single point of accountability for ensuring that Software Change Management processes are in place and requirements are being met, typically the Chief Engineer, Electrical or Maintenance Supervisor.

- A set of procedures and forms to control and track changes to software and its configuration. This allows management to make educated decisions about what program changes are being made, why, and what they will affect. It also enforces a testing and back out procedure for changes and an audit trail of previous changes made.

- A set of procedures to ensure that software changes are implemented in a safe and predictable manner and properly tested following installation.

- A set of procedures to ensure that systems are recoverable and that equipment failure will not result in serious lost time. The nature of these procedures will depend on the type of system. Some backups will be through automated systems and other will require a periodic equipment visit by an ET. In some cases, responsibility for backup and recovery will be entirely on the vendor. In each case, appropriate instructions should be written up and cataloged appropriately.

- A registry of installed software and a history logging version and configuration changes. This allows management to know what software has been in place at any given point in time.

- Formalized processes and documentation to ensure effective handovers are made.

The benefits of a good SMOC program are:

- Increased system uptime via stability, predictability, and accountability

- Ensures only tested well understood software changes are installed

- Ensures any changes made can be backed out in a timely manner

- Software changes are monitored and authorized by appropriate personnel

- Improved disaster recovery

- Better control and communication with vendors

- Lower cost of software ownership

- Easy tracking of software maintenance contracts and licenses

- Ensures good information portability and transfer between crew changes
The risks of a weak SMOC program:

- More NPT due to system down time.
- Increased reliance on the vendor to provide information and testing voluntarily
- Unexplainable or incidents with indeterminate causes
- Undocumented and uncontrolled changes by vendors
- Unexpected behavior or unintended consequences when interacting with other machines because of inadequate testing
- Regression of previously fixes issues. Software upgrades are particularly prone to this as they often overwrite locally made fixes.
- Ambiguous liability and accountability for problems introduced by changes by third parties
- Possible data loss after equipment failures

Common Problems

SMOC is not just about filling out forms. The filing of the paperwork provides accountability, but unless the details of the proposed changes are well understood, many problems are still likely to occur. It takes time for the crew to gain the experience needed to understand the pitfalls involved in managing computer systems.

Considering the risk of injury and damage that can be caused by even minor software changes, it is essential that the testing of the software systems be thorough and complete. All the machines controlled by the program modified should be function tested as well as each system that it interacts with. Special attention should be paid to emergency stop, zone management, position limits and other safety interlocks.

In a typical software company, a software release will be tested by a minimum of 3 people (Developer, Quality Assurance Department, and Client). The software release will be deployed to 5 systems before being released to production and retested at each deployment (Development, QA, Client Test, Parallel test, and Production) before it is released for use. The standard procedure is detailed in Appendix A.

Currently, the offshore drilling industry does not have anywhere near this level of quality checking but is dealing with a far more hazardous work environment. Again, this puts the burden on the contractor to insist on a thorough retest after every software change.

Common Problems

- **Bad Releases:** Because of the nature of developing HMI software, it is difficult to test the software on shore. Sometimes it seems as if the software was never tested at all, as it is “Dead on Arrival” and a local technician must fix it to make it work.
- **Technician Error:** Some fixes are created on the spot by a local technician/software engineer. It is generally considered very bad practice in the software industry to have the same person both make a change and test it. Developers test things to see if they work, not to see if they are broken.
- **Error Regression:** Often an upgrade patch from another rig is converted to be used on the rig. The upgrade patch can then overwrite any changes and fixes made on the local rig.
- **Error Porting:** When a patch is installed on a rig, all rig specific parameters must be manually updated to match the new rig. This is an extremely error prone process.
- **Installation Error:** Sometimes an update has multiple interdependent parts or a specific installation order. If a part is missing the error may not be easily detected but can cause problems later.
Side Effect Error: The new software may function perfectly but be incompatible with other software already in use.

Network Error: Installations often require the computer to be rebooted or the server restarted. If good restart procedures are not followed, this can cause communication problems on the network until the system is restarted properly.

Note: It is a mistake to believe that small changes in software do not require the same level of documentation as large changes. A small change to a value can easily cause a collision or other serious problem. As the amount of work is small, human nature often causes the equipment to not be tested as thoroughly afterwards as it should be. This can present more risk than a large change where a complete examination and retest is more likely to be performed. It is very important that the system be locked down and all future changes be well understood, recorded, and properly tested afterwards. There is often great resistance from technicians and vendors as to the need for further testing. Technicians usually want to test the changed functionality and not the entire system. This drastically underestimates the amount of testing needed to ensure the changes made do not have unexpected side effects. This puts the burden on the contractor to insist on a retest of the software.

SMOC (Software Management of Change) Training

One of Kingston Systems objectives is to ensure each segment of the responsible parties on board understand their role in the SMOC process. Informal conversations and real world examples listed in the case studies section of this document are used to demonstrate key principles.

- **OIM and Company Man:** Enforcement of SMOC policies, Approval and sign off. Basic understanding of SMOC processes and pitfalls is recommended, but the majority of the work falls on the chief engineer.

- **Chief Engineer:** The CE is the usually key person in the approval process. This can place a significant amount of new paperwork on his desk and often requires learning about networks and computer system management. This area is often the greatest obstacle to getting good SMOC processes in place.

- **ET staff:** ET staff often becomes a network security line of defense to prevent unauthorized vendor access to machinery. During start up phases (and afterwards) many vendor technicians access the system and make changes to software and parameters without any notification, approval or testing process at all. Record keeping and software registry tasks are often added to the ET task list.

- **Vendors:** Vendors need to be more responsible and provide proper documentation and testing procedures. The contractor should have Software Change Request (SCR) forms prepared for vendors to fill out to ensuring proper preparation for any changes made. Changes must be tested properly before distribution and proper installation and post installation testing procedures are required. Responsible vendors usually have this information prepared for the onboard technician.

- **Machine Operators:** The machine operators are the final and often the most critical link in the process, yet frequently the most neglected. Ultimately, it is the machine operators and their crew who are exposed to risk of injury, death when things go wrong. Machine operators need to know how software changes can affect them and their crew and how to mitigate associated risks by ensuring all functionality and safety systems are rechecked after each update.

Because the operators are never in the position to know exactly what has been changed, it is critical that each time after a technician accesses the system, physically or remotely, that they run a set of checks on logical interlocks and safety systems. Depending on the magnitude of the change a formal retest may be required, but at a minimum, the next run through on the equipment should be slowed down and a pause at each step to a test of...
relevant interlocks should be performed. This simple step can prevent a serious accident.

SMOC Case study 1 (observed on another rig)

Crash of the sack room terminal. A HMI terminal in the sack room was discovered with an unresponsive terminal after the close out of the Permit to Work for installing software updates on the DrillView system. This was determined to be caused by the restarting of servers during the software upgrade without properly reinitializing all the terminals afterwards.

Emphasizes the importance of:

- **Post Installation Testing**: Technicians are prone to only checking the update for items they were told have been changed. Rechecking of functionality across the entire system is critical as there are often unexpected side effects. This is a big job and requires effort from the crew. The Chief Engineer should make this a priority in all installations and insist on a detailed and complete written installation procedure. Special attention should be given to safety systems including emergency stops, anti-collision systems, and equipment interlocks.

- **Understanding the Interconnectivity and Interdependence**: System complexity is often underestimated. Technicians are often specialists in a very narrow subset and are unaware of possible consequences to other systems.

- **Good Restart Procedures**: Servers and terminals sometimes need to be restarted in certain orders. Restarting a server without reinitializing the connected terminals may cause the terminal to hang up after having lost its connection. Unless a very careful study is made, it is generally recommended to restart all connected servers and terminals according to OEM instructions after a software upgrade.

SMOC Case study 2 (observed on another rig)

Software Upgrade Installation failure. A SCR (Software Change Request) was filed and approved. Time was allocated under the Permit to Work process and other users were locked out of the network and from access to effected machinery. Unfortunately, the technician was unknowingly provided with a bad release package. To further complicate the situation, no offsite support was available to the technician. When contact with the home office was reestablished, the missing files were sent but blocked by antivirus software. Eventually, an alternate route for software delivery was found. After the installation was completed, it was found to be incorrectly programmed and was of no use.

The end result was that several hours of system lock out time doing tasks that should have been done offline. Because of low software quality and poor vendor testing, the end result was a completely preventable waste of time.

Emphasizes the importance of:

- **Good installation procedures**: Because down time is expensive, the complete installation package should be in hand locally, not downloaded over a slow connection after the lockdown has been started. The update package should be checked to ensure all parts have arrived before filing for the Permit to Work. When possible, onshore support should be available to help resolve any problems. The Chief Engineer should require offshore support to be online during installation time periods. Where possible offline testing should be done so that the installation package can be confirmed as good before locking out the equipment.

- **Post installation testing**: It is important to never assume that the installed package actually does what it is supposed to do.

- **Good back out procedures**: After a failed update, it is important to be able to restore the system to its original state.
Vendor Responsibility: Good SMOC procedures allow management to control installation processes and hold vendors accountable for wasted time and low quality product. Because the expected installation time and procedure was documented, it was easy to prove a failure and promote the issue up the chain of command to prevent future occurrences.

SMOC Case study 3 (observed on another rig)

The Drawworks control server was taken off line and rebooted for backup. This backup was approved by the Chief Engineer. During the backup process, the Driller was in the chair attempting to move the Drawworks. The Driller and the Technician were in informal communication but no specific isolation or work stop was in place.

The driller’s chair left touch screen had a menu open which stopped responding after and during the reboot. The Drawworks came to an unexpected halt and the driller was unable to exit the screen or operate the chair menu using the “close button”. The chair had to be rebooted before full control was restored. There was no risk of damage or injury as a non-critical function was being performed and the equipment automatically stops motion when control is lost.

The following actions were not taken but would be required to satisfy SMOC concerns

- A scheduled time in which the action is to take place. This would prevent management confusion as to why alarms are being generated and why equipment is off line.
- A formal notification of supervisors and operators that the action was about to be performed. This, Job Safety Analysis (JSA) would have prevented the miscommunication and avoided any potential equipment or personnel harm.
- A written set of steps. This would have allowed a supervisor to recognize that a reboot was to occur and better understand possible side effects.
- Equipment Isolation would prevent any changes from causing sudden unexpected movement or inability to move.
- Complete understanding of side effects so that the procedure included resetting of other effected machinery. This requires the approvers to take additional responsibility in understanding how the equipment functions and is interconnected.

Lesson: Equipment should be locked out and isolated during control system work and under a permit to work system. Restart sequences for equipment need to be followed and well understood.

SMOC Case study 4 (observed on another rig)

Software upgrade is installed leading to a collision between the top drive and the top of the drill pipe because the update was designed for a rig with a shorter derrick. The pipe was bent out of position and was in danger of popping out of the vertical pipe handler gripper arm. The upper stop limit set point had been unknowingly changed by the software upgrade.

Lesson: Vendor test scripts are not infallible. Software upgrades sometimes change parameters unexpectedly and without the knowledge of the technician performing the upgrade. Sometimes hidden logic changes have unintended consequences that are not discovered until after installation. Following every change, all limits and interlocks must be checked and tested.

SMOC Case study 5 (observed on another rig)
A software change was made to zone management settings and was retested between two machines. The interaction with a third machine was not tested and caused a collision resulting in injury risk and 2 months of critical machinery down time.

Lesson: It is human nature it to check to see if changes made are working, and checking to see what new errors have been introduced is often skipped. Following every change, all limits and interlocks must be checked and tested.

SMOC Case study 6 (observed in a shipyard)

Conversion rate set point change on a rail crane leads to the machine running off the end of the rail and serious equipment damage.

Lesson: very small changes can have very large effects.
Appendix A

Software Industry Standard Software testing and release procedure.

The following is an example of a typical software release procedure that would be used by a software development company producing a client server or web-based product. This industry considers 99.95% uptime to be a minimum standard for performance. Releasing an erroneous release to production would be treated as a very serious event. This level of testing is standard in the software industry and the Oil and Gas Industry is far behind minimum standards.

Key Features:

• Changes are never made directly on the production, parallel, test, or QA machines.
• Each installation is reproducible and repeatable and the system configuration at any point in time is recoverable.
• A change made is tested by a minimum of 3 people, the developer, the QA department, and the client. In practice usually there would be at least two more, the project manager and at least one other client tester.
• A change is tested approximately 5 times before it is released to be used in production.
• Every previous change made in all previous releases is retested in every new release. This is often done using an automated testing program.

Release Procedure

1. Software change is made by a developer on a development machine and tested locally. This change can be as simple as a number format change or a spelling mistake. Once the change has been tested locally by the developer an issue fix release package is created.

2. The release package is installed on a QA (Quality Assurance) Machine along with all the other changes made by other developers

3. The internal QA department tests the releases and ensures that none of the changes interfere with each other and the product is functioning correctly. These tests will test all issues to ensure they are fixed, retest every previous fix made in all previous releases, and a complete functionality test to ensure nothing new is broken. A release is assembled from all the changes and put into a package. The package contains a set of release notes that detail all the changes made and a test plan.

4. The package is then released to a client test machine where the client will be allowed access to test and approve the software. Usually the installation on the Client test machine is pretested by the QA department before the client is given access. These tests will be as complete as the tests in step 3.

5. Once the new release is approved it is put into a parallel test machine where it receives the same data feed as the production machine. The output of the parallel test machine is compared to the production machine to ensure fixes are in place and no unexpected output differences are produced.

6. Once the client is happy with the changes, the client signs off and schedules the installation of the release. The installation is planned to coincide with low usage time periods. The change over to the new release follows a specific written procedure of shutting down the software systems, backing up the previous installation, running the installation, restarting the system, functionality check, returning to online status.

7. An error at any point in this sequence usually results in a failed release. Each system will be reverted back to its original configuration and the process will restart at step 1, 2, or 3 depending on the nature of the problem.