# Measure

## Addressing the Problem

The lack of software guidance set forth by both LM and the government’s military forces from the initial design phases and throughout the product lifecycle caused vague and lax building and testing standards, drove the project over-budget and over-schedule, and foreshadowed a lack of accountability by both parties. The lack of software cohesion and training set the program back a minimum months if not years. Due to the government’s failure to suggest easily integrated software products and develop a more efficient software training plan the program spent significant amounts of time and effort improving the legacy system instead of adding in design to an improved current system.

If software programs were embedded into the initial negotiations with the representing government bodies and LM, the TBMCS would have achieved greater levels of traceability and accountability. The TBMCS did not need software programs convenient for one side of the negotiations, the program needed jointly agreed software programs, even if commercial, to assess system performance and leave little open to interpretation. The directionless software choices, set much later on in the product lifecycle, forced LM to accommodate what the government thought would be easily integrated software solutions. The deployment of the software nor the interaction among operation centers and the system never even reached fruition. Over 20 organizations were involved at some capacity in the TBMCS software selections and training in the year before TBMCS underwent its first operational test and evaluation, with a complicated architecture for developing the system’s software and network design in the TBMCS from the top down. This lead to a chaotic clash between the organizations. In addition, there was a lack of formal concept of employment, tactics, techniques, and procedures (TTPs) which were inconsistent and varied, and there was never an agreed on formalized measures of performance leaving vague stipulations for what kind of software selection would be needed for optimal performance. LM and the government failed to establish a distributed learning environment inducive to adequate funding, the best training materials, and delivery of quality products on time to all users.

From the start, the TBMCS was supposed to provide a common and shared operations and intelligent database with a common suite of tools to plan, manage, and execute the ABP, and was to include a common operational picture for shared situational awareness. Ultimately, the decision flow generating difficult-to-manage software expectations affected not only the system as whole but at a system-of-systems level. The difficulty in determining the impact of software selections was directly seen during operational testing and schedule delays. Due to the massive failure of the software selection and prohibitive training for the TBMCS system, the performance baseline was volatile up to system acceptance, long after operational tests and evaluation were performed.

## Measurement Validation

A number of measurements could be implemented and tracked to improve the problems arising from a lack of definitive requirements early on in the TBMCS program.

Since the software selections drive early development and design decisions, project performance measures may have included cost performance index (CPI), schedule performance index (SPI), defects per single line of code (SLOC), customer complaints, corrective action requests, inquiry response time, and defect containment.

The process of their performance could be guided by defects (deficiencies) per unit (DPU), defects (deficiencies) per million opportunities (DPMO), rolled throughput yield (RTY), sigma levels, and process capability indices. This group of measurements would help determine the progress and consistent quality of the end-product. Since a legacy system is already in place, assuming the data is available, LM members could compare their relative progress and success with the legacy system to get a rough idea of their progress. Defects can be relative to the difficulty of the integrated technology and schedule pressures but overall the number of defects should be very similar to the legacy system, barring any major issues during development. For example, if 50 software bug defects were identified per million opportunities in the legacy system, if the TBMCS system was showing 500+ software bug defects per million opportunities this would signal major issues arising during the process flow. DPUs in the TMBCS would be a more difficult measure just because the units of the TBMCS would most likely only break down to theaters which have a higher variance than more simplistic sub-systems.

The communication of the software decisions and training could use a top-down flow or downward flow approach, a bottom-up flow or upward flow, and horizontal communication. In the top-down flow of information, upper management in the TBMCS would directly communicate to all the workers below them in the organizational chart. While the message may come as more inspiring and invoke a sense of connection, it could also risk coming off as condescending to middle-managers. In the downward flow approach, middle-management would take more control of the communications by receiving instructions to relay to those below them. This would risk the trustworthiness due to lack of transparency with upper management and those low on the organizational chart. The other approaches basically show how information would flow up to upper management but the key takeaway is transparency by upper management leading to the lower-level organizational chart members feeling free and trusted to voice their dissenting opinions without repurcussions.

The TBMCS program lacked communication between LM and the government. The government would be paying for the features of the TBMCS and LM appeared to dance around demanding wants from the government and asking questions they would need to make the TBMCS a more definitive design early on. The government for the most part would take for granted software demands because they most likely wouldn’t know the direct technical impact introducing new software would have as LM could if they worked with them more regularly. The government would just name their functionalities vaguely and hope LM would produce the proper paper trails and technical progress to monitor progress on the feasibility of each feature. Since a legacy system was already in place, the government just stated what they would like improved over the legacy system without a firm grasp about what it would take to accomplish their desired features. This lead to additional confusion because the government could then demand a certain functionality they didn’t explicitly request but actually wanted be incorporated into their system. By going through an in-depth discussion of software functionality, LM could reveal features to the government that wouldn’t be obvious by just talking about top-level functionality of the system.

International Standards Organization (ISO) 9000 and 9001on quality management systems (QMS) and compliance and ISO 5807:1985 on information processing may help with process flow and mapping. Organizations distribute process information to their suppliers and customers usually in an effort to obligate their contract. By complying with ISO 9001, the written procedures and work instructions written by LM would help communicate necessary information to the government consistently. By capturing what is done in the process, why it is done, where it is done, when it is done, who does what, and how it is done the government retains the desired traceability for a multitude of program and process improvement factors. If a high-level supplies, inputs, processes, outputs, and customers (SIPOC) map was in place, the cause-effect relationship of requirements could be visualized by a fishbone diagram. After the cause-and-effect diagram with additions of cards (CEDAC) was brainstormed and analyzed, a list could be composed ranking the priorities of the enterprise. They could then be visualized in a Pareto chart where LMs resources could be put into issues critical to the government or issues that have more financial impact where they would be weighted.

A number of measurements could be tracked to show the program’s overall progress. LM seemed to lack basic measurements that would have foretold the struggles they would encounter down the road. Measurements stemming from software and training such as technical performance measures (TPMs) and measures of performance (MOPs) could have defined what specifically the government wanted and what benchmarks they needed to hit going into operational tests instead of just showing the TBMCS was an improvement over the legacy system. More detailed plans could have been generated showing what technical processes needed to be put in place to reach features requested by the government without schedule delays.

As with any program, LMs TBMCS program as operating under a number of constraints vying for resources. By amassing data with statistical significance, valid conclusions may be drawn. The population of the software selections may have included statistics such the binomial distribution to show the number of defects in the software as a result of the software cohesion. Quality and other divisions within TBMCS must decide what measurements are critical to determining how their product is coming along with the implementation of software early on.

## Baseline Data Collection

Some data collections methods LM could utilize with government are surveys, face-to-face interviews, focus groups, mystery shopping, customer feedback, automatic data capture, and manual data capture. From the methods, one-on-one methods retain a higher integrity and provide opportunities to clarify with respondents. Surveys would have a lower response rate and can mislead the end-results.

Some pitfalls that the data would aim to avoid:

* Undefined units of measure
* Illegible handwriting
* Inadequate measurement system resolution/discrimination
* Precision error
* Emotional bias
* Guesswork or personal bias
* Multiple points of data entry
* Poor instructions or training causing erroneous data entry
* Ambiguous terminology
* Clerical or typographical errors

With the main issue stemming from hard-to-integrate software selections, the TBMCS program would have benefitted immensely by constant feedback from the customer, the government’s military forces. The vision of the government branches involved with the program did not align with LMs vision because of the disconnect between the two leading to a lack of baseline target performance measurements. While there is little indication about the amount of automatic and manual sampling from the technical process aspects of the TBMCS, case studies on the program seem to point the finger at the confusion spreading due to what measurements would be critical to proving their product’s functionality.

Ideally, LM would get together with the government and make a definitive list of software selections for each sub-system and functionality, not mere demands by the government later on. From that point, TPMs and other measures stemming from the software could be set and agreed upon by LM and the government.

## Data Validation

Many of the technical measurements’ validation would derive from the software performance set forth. Many technical measures will have upper and lower specification limits with designated criteria such as “acceptable”, “unacceptable”, “exceptional”, etc. Some pitfalls LM would look out for in the measurement system analysis (MSA) are:

* Calibration
* Stability
* Repeatability
* Reproducibility
* Linearity
* Bias
* Accuracy
* Precision

LM could conduct a gage repeatability and reproducibility (GR&R) or similar study in hopes of figuring out what measurement variables are important with what parts over many trials. A GR&R would relate back to the avoidance of pitfalls listed above to understand variation in the production process.