A Strategic Primer for Army Software and Automation Capabilities Development

Case study

Christopher Carver
Army Contracting Command

Disclaimer: The opinions and views expressed in this article are those solely of the author and do not represent the official position of any U.S. government agency.

In the 21st century, use of specialized software to manage everyday functions of the Army Generating Force has become essential, even critical, to managing near and long-term requirements. Where possible, commercial off-the-shelf software is utilized, but often the necessary capabilities of the specific and usually unique mission requirements are too specialized to be adapted to commercial software, and the development of customized software is required.

How this software is developed is dependent upon the complexity and resources of the unit or organization that needs and develops it. If complexity and software licensing are not a factor, the system can be developed in-house using information technology specialists (be they either Army civilians or contractors on staff). If system requirements are more extensive, an outside software vendor can be contracted to develop software for the unit/organization.

Background

U.S. Army Force Management Support Agency (USAFMSA) originally developed The Army Authorization & Documentation System (TAADS) in the 1960s -1970s, and Block II was finalized in the 1990s. Most of the data elements established by this system are still in use today and form the basis of the current Army Force Management System (FMS). TAADS was developed with a very basic user interface designed for data entry that did not even incorporate use of a mouse. The system was accurate and functional but tedious, inefficient, and time consuming. In 2002, the data tables and system architecture of TAADS was incorporated into a Windows-based user interface (i.e. WINTAADS) that incorporated use of a mouse and other basic Windows functions such as select, copy, paste, and additional useful data input-output features. The new user interface allowed manpower documentation to be much more efficient and reduced the workload for building Tables of Distribution and Allocations (TDAs) and Modified Tables of Organization and Equipment (MTOEs).

WINTAADS was created using internal personnel/resources as an intermediate measure until the FMS could be developed. The creation of WINTAADS was essential to ensuring that Army manpower documentation could keep pace with rapidly changing national military requirements in the 21st century.

The complexity of FMS required roughly a decade to go from concept to operational capability so that WINTAADs was maintained in-house far longer than anticipated. This illustrates a key lesson that must be incorporated into all public sector software system development: the time required to bring a new system online is often significantly longer than expected, and therefore resources and funding must be allocated to maintain/update legacy or intermediate software systems in the meantime. Remaining compatible with other data systems and sustaining input and output interfaces is also a challenge.

Current software systems must be maintained until newer systems become operational and continually modernized to keep pace with user needs and changing hardware-software-operating systems. This requires investment in current/legacy systems even when they are scheduled to be replaced in the near future. Some organizations may be reluctant to invest resources in a system that will not be in use much longer, however, this can lead to significant impairments to mission success.

Planners and system developers are advised to anticipate that replacement systems could be delayed for many reasons, and legacy systems must function until replaced/sunset. Development of new custom software often takes longer than anticipated. Changing environments, resources, and priorities can result in reshaping – or even cancellation of the project – forcing organizations to rely on existing software systems far longer than planned. For this reason, investment in improvements to current software is essential until such time, as the new system is operational. This challenge of balancing the need for continuous improvement, along with the need to limit expenditures on a system set to be replaced, is not as difficult as it may seem.

In order to maintain and update WINTAADS to keep up with new documentation requirements caused by Operations Enduring Freedom and Iraqi Freedom, USAFMSA used a combination of in-house contractors, Army civilians with a basic knowledge of database software, and summer hires majoring in software

development. Thereby with minimal expense, the current system, WINTAADS, could continually be improved to keep pace with Army needs over the course of a decade until the new FMS was operational. This approach works well but must also be balanced with design strategies for more robust future automation. These transitional systems modifications can also provide insight and process information for the future systems.

System Development

The development team is key to ensuring any software system is of the highest quality. One of the common reasons for delays, cost overruns, and poor quality is a "translation gap" between the software developers and primary users (i.e. a failure in effective communication and elaboration regarding system requirements). The cost of developing customized software from an outside vendor is usually significant and therefore requires approval of high-ranking officials. It is also unusual for approving officials to be among the primary users. Therefore, although ultimate authority for the new system lies with them, the role of specific design details should be placed with someone who will be a primary user and is a subject matter expert (SME).

Consider an example in the operational force of a new targeting computer for the Abrams tank. To create the most efficient system, the developer should receive most of the detailed input from the tank crew and not the senior acquisition officer in charge of the program. The same principle of the primary user being essential to the success of software development applies in the generating force.

Another example of a translation gap between software developers and primary users can occur when there is too broad a gap between the knowledge and experience of software developers and the knowledge/experience of primary users on the development team. Outside vendors are likely unaware of most (if any) of the acronyms, processes, input and output data, timelines, roles-permissions, and policies that are essential to the organization's operations. There must be a dedicated effort by both the vendor and organization to familiarize the developer with a basic understanding of data elements essential to the system as well as the procedures and processes driving the mission of the organization.

Additionally, a gap may exist between primary users and basic software development. Users may know what they want the software to do but might be unable to communicate to the vendors how they expect the software to do it. This translation gap can result in unrealistic expectations of vendors. When utilizing an outside vendor to build a software system, there can be an expectation that its developers bear the entirety of figuring out the "how" of the software's processes. This is a mistake that will lead to cost overruns, inefficiencies, delays, inadequate features/analytics, and possibly the project's failure.

Both the vendor and user equally share responsibility for determining how the system operates. The user may not know how to write code for the software; at minimum, a few of the organization's development team needs to have basic knowledge of the type of computer operations that they are asking the vendor to program. These individuals are essential to translating *what* exactly the organization needs into *how* the software should operate.

A real-world example of these principles in practice took place at USAFMSA during development of the TDA module of the FMS. The senior leader in charge of the project (TDA division chief) assembled a team of SMEs and assigned an individual who had experience in manpower documentation, database development, and programing as the team lead. Rather than dictate the minutiae of the system requirements, the TDA division chief empowered individuals who would be daily users of the software and placed SMEs on the team in key roles (in TDA documentation and software development) while retaining the authority for vital decisions. Regular updates and key decision points that would affect time and resource expenditure were brought to the division chief but otherwise authority for most of the decisions was moved to where the information was (i.e. the SMEs and end users).

The result was a manpower system that was highly efficient, user friendly, and highly adaptable to changing Army policies and guidance. These attributes should be the primary characteristics sought for any software system in development for the Army. This goal is attainable and can be achieved by adopting the design philosophy utilized by the FMS TDA module development team.

End User Participation in Development

As discussed earlier, primary system developers were ultimately end users with extensive subject matter expertise that incorporated virtually every aspect of TDA manpower documentation. This ensured all functional requirements would be reviewed and addressed in development.

The approach also promoted system efficiency due to the end user directing the system inputs required to perform any given task. As originally designed, several basic functions required multiple mouse clicks and scrolling through screens or options to complete a task. This lack of efficiency increased the overall workload. End users were able to suggest changes to

the user interface that reduced the number of steps to achieve a task because they understood what the build process entailed and what situations they would likely encounter .

Maximum Flexibility

One of the development team lead's mantras was "maximum flexibility." The manpower documentation experts often had different approaches to performing a common task. Rather than choose "one right way" to perform a function, multiple options were built into the system software.

Another example of this philosophy in was that if a built-in feature of the software was deemed unnecessary, that feature was not eliminated but rather simply turned off. This meant if a need for this feature was discovered in the future, it could easily be incorporated into the existing architecture.

Additionally, inevitable changes in Army policy/guidance were built into the system architecture so that a local administrator could alter algorithms that validated data in accordance with changing guidance – without requiring the Army to seek contractor support for minor changes – thus keeping updates in-house and reducing overall maintenance cost and time to implement needed updates.

User Interface Intuitiveness

In any software system, user interface is key to a successful product. In some cases with individuality developed systems, the end user is stuck with whatever the developer designs and leadership approves, but in the commercial industry, any software with an overly complicated and unintuitive user interface ultimately fails, as customers will not purchase software that is more difficult to use than others that are available. User interface of the TDA module of the FMS was driven by the end user's focus on efficiency and reduction of the current workload.

Among the innovations resulting in efficiency gains include:

- Allowing manual data entry into most data fields with an auto-fill function based on existing data tables used for validation
- Grouping basic functions under the same menu heading
- Reducing the number of menu screens to get to a particular function
- Use of screen formats and functions similar to commonly used commercial software
- Interoperability with commercial software (i.e. Excel) and open system architecture
- Using predictive methods and analytics to populate required data fields
- "Smart" edits to prevent any "bad data entries"

These and other innovations reduced the overall workload of data entry, analysis, and validation. The user could manage a larger dataset in less time and more accurately as a result of the new software.

Summary

There are several concepts to facilitate developing efficient, user-friendly software for Department of the Army use. These include the importance of end user participation in development of the software, ensuring that the end product was highly flexible and adaptable not only to the changing mission requirements of the Army but to the preferences of the individual user. Included in this requirement is the importance of user interface and overall intuitiveness of the system, process, and functions. An intuitive user interface leads to less training time, fewer errors, and greater productivity. When developing a new software system, the lessons learned from prior iterations of data systems must be remembered. As newer systems are developed, the trend must be toward equal or greater functionality, interface intuitiveness, and capabilities as compared to the software being replaced.