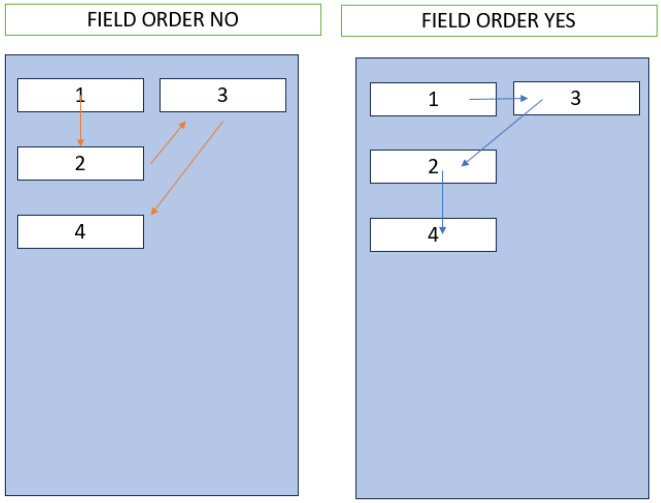
SMITERM 9.19 Details

With each new release of SMITERM our goal is to provide a closer approximation of the final release of TEKTERM that ran on the PSION XT15 terminal. This has been a slow and gradual process because we do not have the original source code and need to reverse-engineer the functionality and the user experience provided by the original TEKTERM product. In order to do this we need "use cases" to examine. Originally, all the "use cases" were from the TEKRF and TEKCONSOLE products. These only used a limited subset of the Commands and Functionality available in the original TEKTERM product. In order to make SMITERM useful for other applications, such as those used in many Port Facilities, we need to continue this reverse-engineering process. We would encourage anyone using SMITERM (or attempting to use SMITERM) to share their "use cases" with us.

We are also driven by changes to the Android Operating System and Android Studio that are controlled by Google. Google mandates that only applications developed with specific APIs can be run on specific releases of the Android Operating system or be distributed through the Google PlayStore. The new APIs are often only available on the newest release of Android Studio. This often includes both new functionality, and removal of old functionality. The result is constant maintenance of the code, and often require significant changes to the code every few years. Unlike Microsoft, Google has a complete disregard for backwards compatibility. They often block older code from running on newer versions of Android.

<u>Problems with Field Order.</u>

The largest problems with developing applications that can run on the many 9010/TESS emulators developed over the last 30 years are related to the "next field" strategies (or rules) used by the emulator to determine where the cursor moves next after specific events. These rules are not implemented the same way in all emulators, and in the case of TEKTERM, depend on how the Field Order parameter is set. Field order in TekTerm is illustrated below, it determines what field is next when an Arrow or TAB key is pressed.



The problem gets more complicated when the terminal must determine what the next field is after various exceptions occurs, such as trying to place the cursor in a field that does not accept focus (like a text field) or a field that does not exist on the screen, or the current field with the cursor is deleted. Again, there is no consistency on how these exceptions are handled by various emulators.

The developer must consider where the cursor will move next. The "Best Practice" is to make sure the code always controls where the cursor moves next, and to avoid situations where the emulator must decide what field is next. This means specifically placing the cursor in the desired field each time there is a change to the screen. This means the developer's code will run the same way on any emulator and will be independent of what Field Order is used. Sadly this "Best Practice" is not always followed, so a compatible next field strategy may need to be selected through configuration.

SMITERM has 3 different strategies available that are controlled by setting the FIELD_ORDER Parameter. These are summarized in the table below:

FIELD_ORDER	ORDER	NEXT FIELD AFTER ERROR	EMULATES
0 (default)	by field number	Start Searching after the Current Field, if the Current Field is invalid then start from the top of the screen	TekTerm with Field Order = no
1	by field location	Always Start Searching from the top screen	TekTerm with Field Order = yes
2	by field number	Always use the Current Field, if the Current Field is invalid then start from the top of the screen	TekTelnet and older versions of SMITERM

Problems with Screen Orientation and Full Screen Mode

Android does not consistently determine the size of the usable screen area. Originally this was not a problem because the screen was always rectangular and had off-screen navigation buttons. Now there are navigation buttons at the bottom of the screen, there may be a cut-out for a camera, and the corners of the screen may be rounded. (which is OK for the Status Bar, but not for displaying the app). This has been further complicated by the new EDGE to EDGE design now required by Android where the default is to use the entire screen. When the screen is rotated, some versions of Android will incorrectly calculate the usable size. In full screen mode, many versions will not properly scroll the popup keyboard.

The best solution was to discontinue providing changes to rotation and screen size through configuration, and only display the app in the device's default orientation and screen size. This meant removing the ORIENTATION and FULL SCREEN Parameters. These no longer have any effect in Version 9.19.

The screen will now automatically display in Portrait on Hand Helds, and in Landscape on Tablets. The screen will not be full size, so the Status Indicators at the top (Date, Time, Battery, Signal Strength, etc) will always be visible.

Changes to Sound.

There was a problem in the way the “Normal” sound was played on older versions of SMITERM. We had originally thought the sound was made each time a Function key was pressed, but this was not correct.

The sound is play after the function key is pressed and immediately after the terminal unlocks. This indicates that the function key was pressed and the Host system has responded to the function key press.

We also separated the Sound and Vibration into 2 sub parameters. This way there are more options for sound events.

The Format is **BEEP,**<sound parameter>**:**<vibration parameter>

BEEP,0:0 no sounds, no vibration
BEEP,1:0 sound played for error messages, no vibration
BEEP,2:0 sound played for error messages and after function key presses, no vibration
BEEP,1:1 sound played and vibration for error messages.
BEEP,2:1 sound played for error messages and after function key presses, vibration for error messages
BEEP,2:2 sound played and vibration for error messages and after function key presses
BEEP,0:1 vibration for error messages, no sounds
BEEP,0:2 vibration for played for error messages and after function key presses, no sounds

Paging

Paging was used on Narrow Band terminals that had limited bandwidth, often with many terminals sharing a 9.6 KB/s wireless network. Bandwidth could be conserved by preloading screens (refer to as pages) on the terminal and then recalling these from memory rather than downloading them each time from the host system. Many were used in Ports, where Narrow Band was used to cover a wide area. Paging is no longer used since we now have Terminals with WiFi and 4G that can transfer data at 10-100 MB/s

Previous versions of SMITERM only supported a limited number of “use cases”. These all incorrectly assumed that the “Open” Command occurred immediately after the “Display Page” Command, and no attempts were made to update Pages after the “Open” command. We have made changes in 9.19 to avoid these limitations and better support paging.

NOTE: The Default for SMITERM is to use a Single Page. This can be changed using the PAGES parameter. Example: PAGES,16 enable 16 Pages (or screens to be saved).

AIAG Limitations

SmiTerm has limited AIAG support, all based on “use cases” from TEKRF and TEKCONSOLE solutions. AIAG functionality is seldom used, as it is limited to Teklogix Hardware, and is not handled on many emulators. It is also difficult to implement and can have unpredictable effects when there is a requirement to both scan and manually enter data into a field.

Typically, AIAG functionality is used to remove the leading prefix on AIAG barcodes (example: part number with barcode P12345, should have the P removed and the 12345 send to the host). This also prevents barcodes with other prefixes from being scanned into the same field. In addition, if a AIAG barcode is scanned it will automatically be placed in the next open field that is configured to accept that prefix. (focus on the correct field is not required). SmiTerm can simulate this behavior is the Scanner adds a prefix (typically ~) to the scan. Smiterm also has a basic AIAG Filter that can be enabled that simply removes any AIAG prefix entered into an AIAG enabled field.

AIAG functionality has also been used to help parse barcodes that contain multiple data fields (example: EAN and 2D barcodes) The best option is not to use AIAG for this purpose, and instead pass the entire scan back to the host for processing. Code running on the host can be used to properly parse the barcode and place the correct data in the correct fields.

No More Text to Speech

We have removed the Text to Speech (TTS) functionality found in previous versions of SMITERM. To the best of our knowledge, feature has not been used by a customer. It was fun for demos, but has not been put to practical use. Please contact us at support@smilog.ca if you require this feature.