Automated Operator Workplace for Portable Measurement Devices

A. U. Battalova
Department of computer science and robotics
Ufa state aviation technical university
Ufa, Russia
e-mail: BattalovaAlina@mail.ru

L. P. Kostyukova
Department of computer science and robotics
Ufa state aviation technical university
Ufa, Russia
e-mail: stasis_mail@mail.ru

Abstract[1]

The article deals with structural complex scheme and the main functions of soil corrosive activity measurement device. Measurement results (data transmission) are presented in form of symbol expression described by regular expressions and closed automation. Transformation to program data model is done with the help of formal rules. Working algorithm of data transmission model editor is shown.

1. Introduction

Nowadays different portable measurement devices for noninvasive control in field conditions are widely spread. Existing devices fix the measurement results with the help of digital display and have enough memory to keep them. The problem is that these results are presented in such form that makes it uneasy to visualize them and make statistical analysis of such data. So further processing and decoding is necessary. Complex structure scheme including portable measurement device (soil corrosive activity measurement (SCAM) device) and automated operator worklace (AOW) is presented in fig. 1

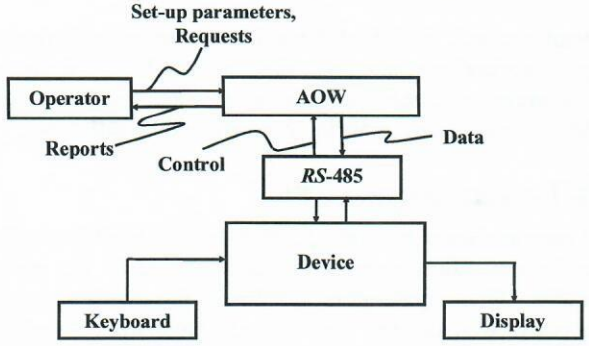


Fig 1. Structure complex scheme "SCAM-AOW"

Proceedings of the 12th international workshop on computer science and information technologies CSIT'2010, Moscow – Saint-Petersburg, Russia, 2010

The main functions of SCAM device are:

- Data keeping in power independent memory
- Memory block control from the keyboard of a device
- Information display
- Connection providing through RS-485 interface.

The main functions of AOW are:

- Phrase recognition, transmitted in message form;
- Logical message processing;
- Connection providing through RS-485 interface;
- Alert set-up;
- Message structure set-up (transmitted message creation and editing, data base support, prognostication methods);
- Object condition prognostication;
- Data display;
- Decision making with the help of a rule base.

Information from device memory is transmitted to automated operator workplace where it is processed and stored through RS-485 interface. Operator inputs transmitted message set-up parameters and system requests. All software modules are divided into2 groups: system for message receiving set-up and user modules.

System for message receiving set-up involves 4 editors: message model editor, decision support editor, user instructions editor and report editor. Input message structure models, rule base for decision support and report forms are developed with the help of these editors. Also system for message receiving set-up involves prognostication models and data base support block. User modules involve lexical analyser, stored data view and decision support block, prognostication block, and connection with data source block.

Measurement data are put to the data base of automated operator workplace. The functional scheme of "SCAM-device- AOW" complex is presented in figure 2.

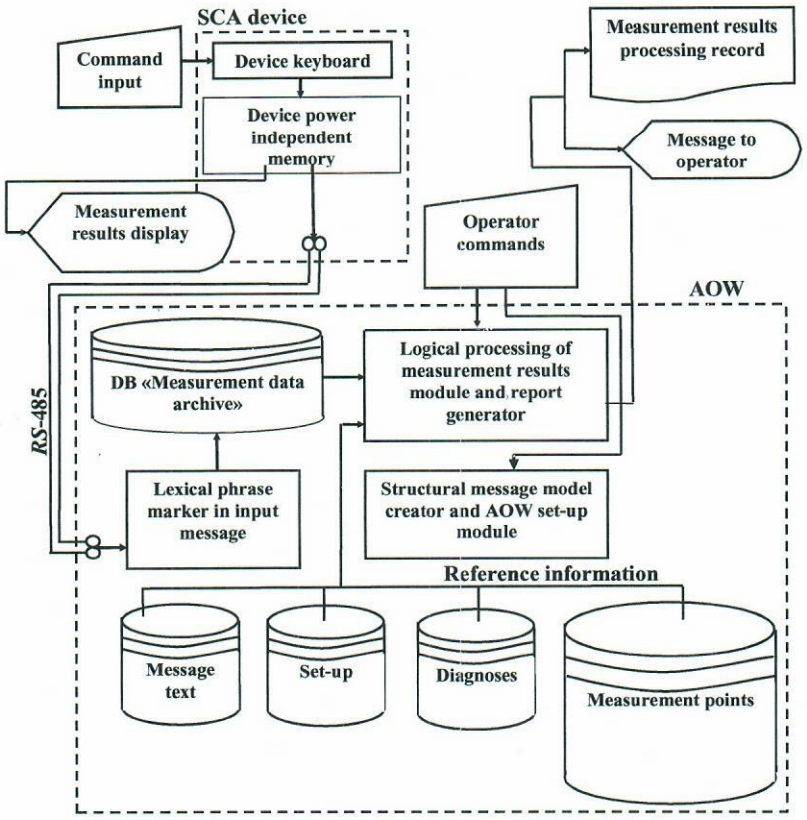


Fig 2. Functional scheme of "SCAM-AOW" complex

2. Message model editor

Transmitted measurement results are presented (fig. 3) in form of symbol expression.

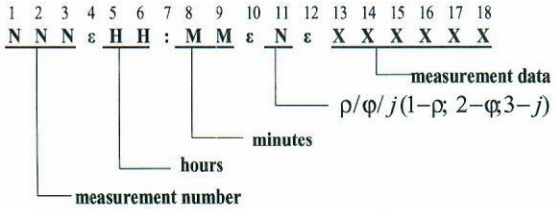


Fig 3. Measurement results in form of symbol expression structure

Symbol expressions are presented by regular expressions:

$$N^*\varepsilon N^* : N^*\varepsilon NN^*;$$

$$N^*\varepsilon N^* / N^*\varepsilon NN^*,$$

where N - count ($N = \{0,1..9\}$; ε - space.

The message is divided in the set of chain parts that are connected with each other by concatenation. Every single chain part is also expressed through iteration. The message is chain of alphabet $z(N, \varepsilon, :, /)$.

In order the operator got the message with measurement results lexical analyzer of phrases of input message should work on operator's computer.

Lexical analyzer (or scanner) reads alphabets of input message X and builds words $Y[j]$, where j is the number of data types in concerned message. The words are further processed.

Mathematical model of lexical analyzer is created and program code is generated, realizing the lexical analyzer for given message structure in message model editor. Model code for each portable device will be different.

3. Lexical analyzer editor

There are some program generators of lexical analyzers code that allow automating analyzer code for given words. These generators are based on words that are already well-known, that is not true for concerned challenge or the software to provide relation between user and lexical analyzer code generator should be developed. Such programs generate lexical analyzer code that is oriented for concerning word structure but the sequence of the words in analyzed message is ignored. This kind of software is not acceptable for portable measurement devices because for these devices the structure and sequence of words are both important.

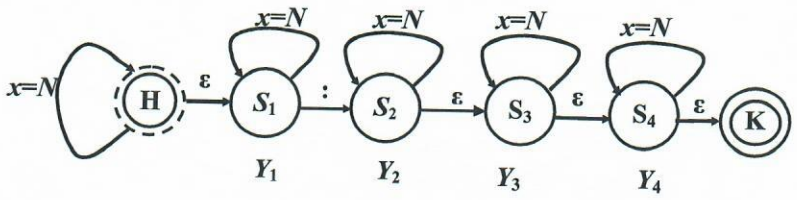
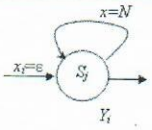


Fig 4. Skeleton automation: H_1 is a first state of the automation; S_i is a set of states; K_1 is a finish state; Y_j is a set of words; ε is space; x is input symbol; $N = 0,1,...9$

Table 1. Formal rules of skeleton automation to program code crossing

Graphical component	Name	Program code

	Initialize	sost:=AUTO_H;
		{FA is in initial state H}
		dl:=Length(stroka);
		{dl– Message length}
		zn:=0; {zn – data type}
		for j:=1 to dl do
		{j – current message symbol number}

	Input symbol	*str:=str+st;*
	Data processing	*case sost of* *AUTO_H: case stroka[j] of* '0..9':*sost:=AUTO_H;* *str:=str+st;* 'ε': *sost:=AUTO_S1; zn:=1;*
	End	{word otput} *end*

Lexical analyzer code is generated with the help of lexical analyzer editor.

Message models in lexical analyzer editor are presented in form of skeleton automation. Skeleton automation is a kind of finite automation (FA) or a finite automation with determined states (DFA).

In order to build skeleton automation for message procession and results view the symbol chain should be recognized. This chain consists of regular expressions like $y = y_1 + y_2 + ... + y_k$, where every symbol is involved into input alphabet z. Skeleton automation is to recognize and remember alphabets, words and chain parts from input message.

The determined automation M_y for chain $y_1 + y_2 + ... + y_k$ identification is built with the help of chain y in form of expression like $N^*\varepsilon N^* : N^*\varepsilon NN^*$ (where $N \in [0..9]$, ε is space). First determined finite automation having $K_1 + 1$ states (H_1, S_1, ... , S_4, K_1) and changing its state from $i-1$ to i with x as input signal is built (fig. 4).

Output data including measurement number, hour and minute of measurement and measurement results are formed in S_i states.

Automation begins working in H_1 first state and stays in it until ε symbol or space will come that means the word end. Then the automation comes to state S_1 and goes to S_2 state after ":" symbol comes that means the beginning of measurement time message part. After that the automation changes its state every time when ε comes and stays in the same state if the input symbol is $x = N$.

The determined automation M_y has the same set of states as the skeleton automation but with single input symbol comes it can do several state changes.

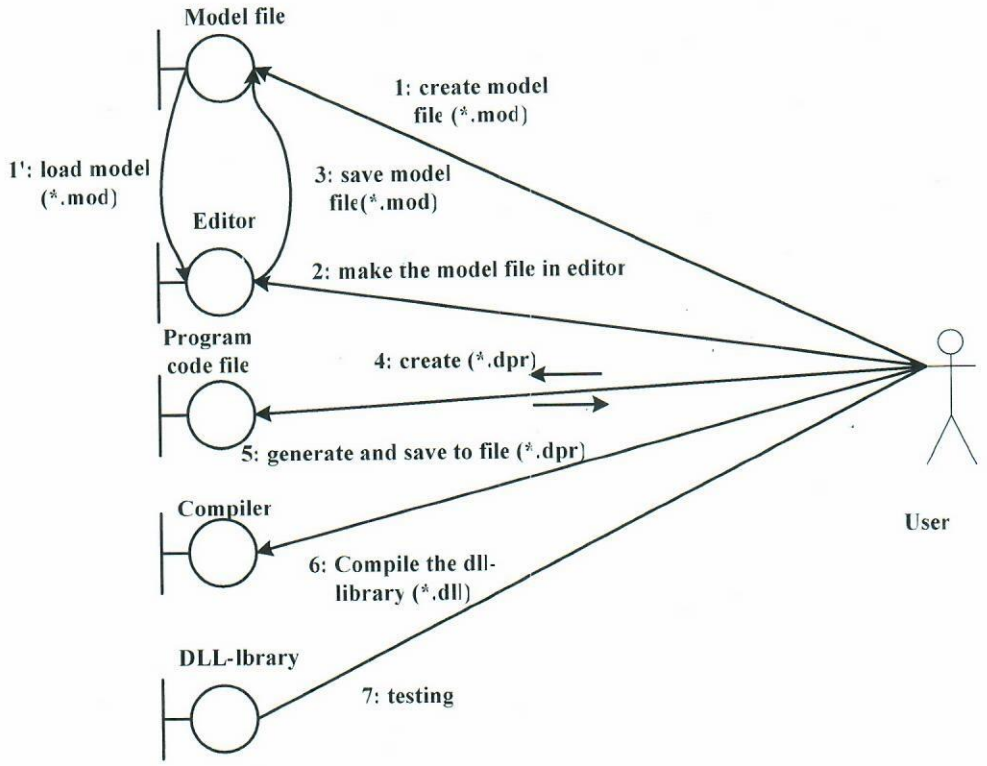


Fig 5. The process of analyzer code generating and message model creating

The skeleton automation crossing to program code [1] is made with help of formal rules (table 1).

Such formalization (table 1) is very helpful for easing lexical analyzer code generation. The next figure (fig. 5) shows the user's act sequence for analyzer module creation and files created during this process. So the user has the opportunity to create and edit manually program files including model and code files and compiler.

4. Conclusion

The lexical analyzer editor for input messages of portable measurement devices was developed. The editor uses only visual means for creation and editing of input message models. This editor allows easing input message structure creation, displays the developed structure, automates lexical analyzer code generation and tests the model.

References

1. Karpov Yu. G. "Automation theory". Piter, Saint-Petersburg, 2003.