Lacmus

SOFTWARE REQUIREMENTS SPECIFICATION ISO-IEEE-29148-2011

1. Purpose

Lacmus is a cross-platform application that helps to find people who are lost in the forest using computer vision and neural networks.

Conventions

UAV - unmanned aerial vehicle;

2. Scope

2.1 Software product name - Lacmus

2.2 Explaining what the software product(s) will do

1. The operator determines the area of searching missing human and starts the UAV according to the prepared route.
2. The UAV carries a photo shoots and transfers information to the operator's laptop with installed Lacmus (RescuerLA).
3. Lacmus sends images to the server, where they are processed with RetinaNet or other neural network.
4. In case of human detection, Lacmus gives mark of the image.

2.3 Describing the application of the software being specified, including relevant benefits, objectives, and

goals

Ближайшим аналогом является ПО, обрабатывающее снимки с БПЛА, оснащенных тепловизорами. Недостаток таких решений - высокая стоимость оборудования и отсутствие свободного ПО для анализа снимков.
Основными преимуществами RescuerLaApp является свободное использование и возможность использования совместно с недорогими моделями БПЛА, оснащенных камерами, снимающими в видимом диапазоне. Это позволит использовать RescuerLaApp в некоммерческом проекте Liza Alert.

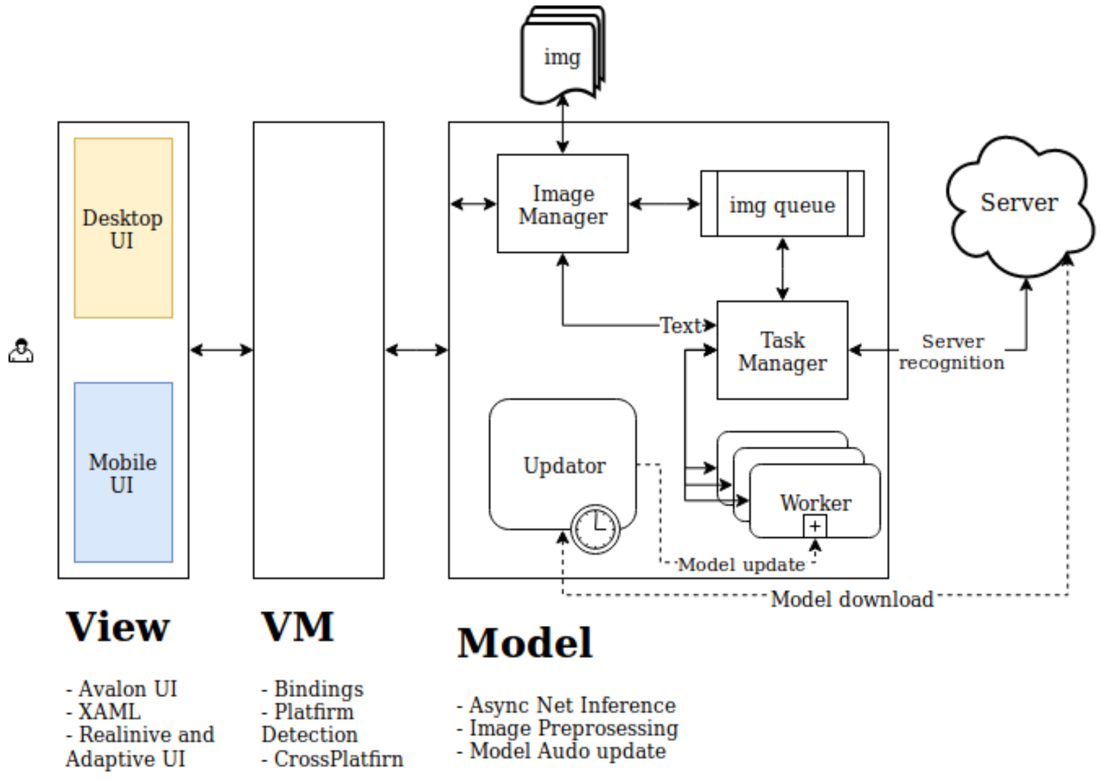
2.4 Higher-level specifications

Not exists.

3. Product perspective

3.1 System interfaces

A block diagram showing the major elements of the system, interconnections, and external interfaces.



3.2 User interfaces

a) The logical characteristics of each interface between the software product and its users. This includes those configuration characteristics (e.g., required screen formats, page or window layouts, content of any reports or menus, or availability of programmable function keys) necessary to accomplish the software requirements.

b) All the aspects of optimizing the interface with the person who uses, maintains, or provides other support to the system. This may simply comprise a list of do's and don'ts on how the system will appear to the user. One example may be a requirement for the option of long or short error messages. This may also be specified in the Software System Attributes under a section titled Ease of Use.

3.3 Hardware interfaces

Specify the logical characteristics of each interface between the software product and the hardware elements of the system. This includes configuration characteristics (number of ports, instruction sets, etc.). It also covers such matters as what devices are to be supported, how they are to be supported, and protocols. For example, terminal support may specify full-screen support as opposed to line-by-line support.

3.4 Software interfaces

Specify the use of other required software products (e.g., a data management system, an operating system, or a mathematical package), and interfaces with other application systems (e.g., the linkage between an accounts receivable system and a general ledger system).

Server software requirements - see lacmus/Dockerfile.gpu

3.5 Communications interfaces

4. Product functions

Provide a summary of the major functions that the software will perform. For example, an SRS for an accounting program may use this part to address customer account maintenance, customer statement, and invoice preparation without mentioning the vast amount of detail that each of those functions requires.

Sometimes the function summary that is necessary for this part can be taken directly from the section of the higher-level specification (if one exists) that allocates particular functions to the software product.

Note that for the sake of clarity
a) The product functions should be organized in a way that makes the list of functions understandable to the acquirer or to anyone else reading the document for the first time.
b) Textual or graphical methods can be used to show the different functions and their relationships. Such a diagram is not intended to show a design of a product, but simply shows the logical relationships among variables.

5. User characteristics

Describe those general characteristics of the intended groups of users of the product including characteristics that may influence usability, such as educational level, experience, disabilities, and technical expertise. This description should not state specific requirements, but rather should state the reasons why certain specific requirements are later specified in specific requirements in subclause 9.5.9.

6. Limitations

Provide a general description of any other items that will limit the supplier's options, including
a) Regulatory policies;
b) Hardware limitations (e.g., signal timing requirements);
c) Interfaces to other applications;
d) Parallel operation;
e) Audit functions;
f) Control functions;
g) Higher-order language requirements;
h) Signal handshake protocols (e.g., XON-XOFF, ACK-NACK);
i) Quality requirements (e.g., reliability)
j) Criticality of the application;
k) Safety and security considerations.
l) Physical/mental considerations

7. Assumptions and dependencies

List each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but any changes to these factors can affect the requirements in the SRS. For example, an assumption may be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.

8. Apportioning of requirements

Apportion the software requirements to software elements. For requirements that will require implementation over multiple software elements, or when allocation to a software element is initially undefined, this should be so stated. A cross reference table by function and software element should be used to summarize the apportionments.

9. Specific requirements

Specify all of the software requirements to a level of detail sufficient to enable designers to design a software system to satisfy those requirements.

Specify all of the software requirements to a level of detail sufficient to enable testers to test that the software system satisfies those requirements.

At a minimum, describe every input (stimulus) into the software system, every output (response) from the software system, and all functions performed by the software system in response to an input or in support of an output.

The specific requirements should:
a) Be stated in conformance with all the characteristics described in subclause 5.2 of this International Standard.
b) Be cross-referenced to earlier documents that relate.
c) Be uniquely identifiable.

10. External interfaces

Define all inputs into and outputs from the software system. The description should complement the interface descriptions in 9.5.3.3.1 through 9.5.3.3.5, and should not repeat information there.

Each interface defined should include the following content:
a) Name of item;
b) Description of purpose;
c) Source of input or destination of output;
d) Valid range, accuracy, and/or tolerance;
e) Units of measure;
f) Timing;
g) Relationships to other inputs/outputs;
h) Screen formats/organization;
i) Window formats/organization;
j) Data formats;
k) Command formats;
l) Endmessages.

11. Functions

Define the fundamental actions that have to take place in the software in accepting and processing the inputs and in processing and generating the outputs, including:
a) Validity checks on the inputs
b) Exact sequence of operations
c) Responses to abnormal situations, including 1) Overflow, 2) Communication facilities, 3) Error handling and recovery
d) Effect of parameters
e) Relationship of outputs to inputs, including 1) Input/output sequences, 2) Formulas for input to output conversion

It may be appropriate to partition the functional requirements into subfunctions or subprocesses. This does not imply that the software design will also be partitioned that way.

12. Usability requirements

Define usability (quality in use) requirements. Usability requirements and objectives for the software system include measurable effectiveness, efficiency, and satisfaction criteria in specific contexts of use.

13. Performance requirements

Static numerical requirements may include the following:
a) The number of terminals to be supported;
b) The number of simultaneous users to be supported;
c) Amount and type of information to be handled.

Static numerical requirements are sometimes identified under a separate section entitled Capacity.

Dynamic numerical requirements may include, for example, the numbers of transactions and tasks and the amount of data to be processed within certain time periods for both normal and peak workload conditions. The performance requirements should be stated in measurable terms.

For example, 95 % of the transactions shall be processed in less than 1 second. rather than, An operator shall not have to wait for the transaction to complete.

14. Logical database requirements

Specify the logical requirements for any information that is to be placed into a database, including:
a) Types of information used by various functions;
b) Frequency of use;
c) Accessing capabilities;
d) Data entities and their relationships;
e) Integrity constraints;
f) Data retention requirements.

15. Design constraints

Specify constraints on the system design imposed by external standards, regulatory requirements, or project limitations.

16. Standards compliance

Specify the requirements derived from existing standards or regulations, including:
a) Report format;
b) Data naming;
c) Accounting procedures;
d) Audit tracing.

For example, this could specify the requirement for software to trace processing activity. Such traces are needed for some applications to meet minimum regulatory or financial standards. An audit trace requirement may, for example, state that all changes to a payroll database shall be recorded in a trace file with before and after values.

17. Software system attributes

Specify the required attributes of the software product. The following is a partial list of examples:

a) Reliability - Specify the factors required to establish the required reliability of the software system at time of delivery.

b) Availability - Specify the factors required to guarantee a defined availability level for the entire system such as checkpoint, recovery, and restart.
c) Security - Specify the requirements to protect the software from accidental or malicious access, use modification, destruction, or disclosure. Specific requirements in this area could include the need to: c.1) Utilize certain cryptographic techniques;
c.2) Keep specific log or history data sets;
c.3) Assign certain functions to different modules;
c.4) Restrict communications between some areas of the program;
c.5) Check data integrity for critical variables;
c.6) Assure data privacy.

d) Maintainability - Specify attributes of software that relate to the ease of maintenance of the software itself. These may include requirements for certain modularity, interfaces, or complexity limitation. Requirements should not be placed here just because they are thought to be good design practices.

e) Portability - Specify attributes of software that relate to the ease of porting the software to other host machines and/or operating systems, including: 1) Percentage of elements with host-dependent code; 2) Percentage of code that is host dependent; 3) Use of a proven portable language; 4) Use of a particular compiler or language subset; 5) Use of a particular operating system.

18. Verification

Provide the verification approaches and methods planned to qualify the software. The information items for verification are recommended to be given in a parallel manner with the information items in subclause 9.5.10 to 9.5.17.

19. Supporting information

The SRS should contain additional supporting information including
a) Sample input/output formats, descriptions of cost analysis studies, or results of user surveys;
b) Supporting or background information that can help the readers of the SRS;
c) A description of the problems to be solved by the software;
d) Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.
The SRS should explicitly state whether or not these information items are to be considered part of the requirements.