Dr Mobil

Dr Mobil is a fictitious device that will help EMPs in the medical emergency area to diagnose the injured.

Signoff

TOC

Revision History

|  |  |  |
| --- | --- | --- |
| Date | Revision# | Description |
| 6/1/2020 | 1.0 | Initial Content |

# The Introduction

Dr Mobil is a fictitious device that will help EMPs in the medical emergency area to diagnose the injured. The device is a handheld device the size of an iPad. It is made out of clear plastic carbonate, with two handles on either side.



Figure 1:Dr Mobil Prototype[[1]](#endnote-1)

Most TV shows have different versions of this device, some include:

* Star Trek – Medical Tricorder
* Expanse -   
  

Figure 2:Expanse Communication Device[[2]](#footnote-1)

* Marvel: Iron Man & Agents of Shield
* Subnautica

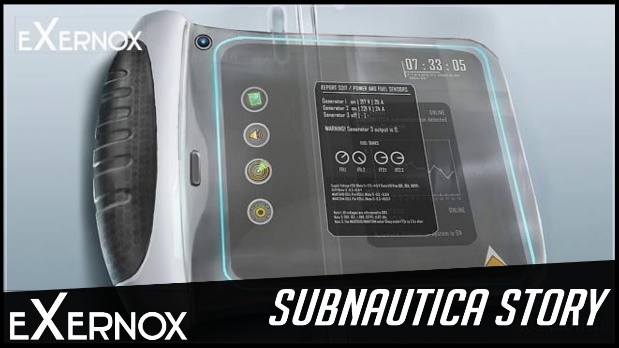


Figure 3 Subnautica PDA

## Purpose

Dr. Mobile is a solution to the problem of having too many devices. A EMP needs to be mobile and light. Today’s technology is already contained in mobile devices, with a few modification and software packages tables can be easily converted to assist in EMP in an emergency.

## Scope

Dr. Mobile will consist of hardware and software components.

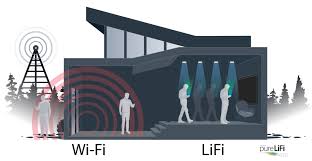
The hardware components will include:

* Dr Mobile Tablet
  + Dr Mobile Authentication
  + Dr Mobile Home page
  + Dr Mobile Patient
    - Dr Mobile Add Patient
    - Dr Mobile Diagnosis
    - Dr Mobile Monitor
  + Dr Mobile Communication
    - Homebase Services
    - Order Supplies Services
    - Transportation Services
  + Dr Mobile Settings
    - Communication
    - Power
    - Sensor Setting/Calibration
    - Account Information
* Basecamp Medical Devices
  + Monitor Wristbands
  + Other Dr Mobil Tablets
  + Relay communication devices
* Homebase Services
  + Homebase Server Services
  + Homebase Database Services
  + Homebase Transportation Services
  + Homebase Supply Chain Services

This device will have all of the current sensors found in today’s devices:

* Internal focus Sensors: Accelerator, Gyroscope, Inclinometer, Oscilloscope, gForce
* External focus Sensors:
  + Camera: Spectrograph, Color detection, Light meter
  + Barometer, Proximeter, Magnet Meter, Compass, GPS, ,
  + Sound: Mic, Sound meter, Tone Detector
  + Spectrum Analyzer
  + Touch: Finger Print, User Input, Tap
  + Radio: FM/AM
  + Network: Cell, Wireless, Bluetooth
* Emitters: Speaker, Infrared, Light,

Additional feature of the product:

* Uses latest solar power technology, using the clear glass as a collection energy from the sun, and storing it in 24 hour batteries.
* Can communicate via Satellite, Ham Radio, Tower to Tower (Relay, Cell or Ham)
* Communicate via external sensors
  + Wrist band – pick biometrics, monitoring, etc.
  + Other devices: Reading information that is need to be broadcast
  + 360 degrees of sensory of temperature change, depth of 25 feet through solid rock (IF Sensor)

Communication:

* Communication with main office (Cell Phone)
* Communication with other medical devices (Point to Point)
* Communication with base camp (Wireless network)
* Communication with supply chain (Satellite Phone)
* Communication with transport (Short wave radio)

This device will be used during disasters (natural and manmade) scenarios.

Software Application Functionality

* Determine biometrics: temp, pulse, pressure, oxygen level, diseases,
* Wrist band – pick biometrics, monitoring, etc.

Software simulation – Tiddlywiki

* Show prototype screens
  + Main screen
  + Monitor screens
  + Communication connectivity
  + Diagnose
  + Algorithms
* Integration of packages
  + Communication pkgs
  + Sensor pkgs

You are part of the testing team that has been hired by your H.E.L.P., Inc (Health Emergency Lifeline Protectors) to oversee the testing of the product before it is delivered out to the field.

## Product Overview

### Perspective

Product perspective

Define the system's relationship to other related products.

If the product is an element of a larger system, relate the requirements of that larger system to the functionality of the product covered by the SRS.

If the product is an element of a larger system, identify the interfaces between the product covered by the SRS and the larger system of which the product is an element.

Consider a block diagram showing the major elements of the larger system, interconnections and external interfaces.

Describe how the software operates within the following constraints:

* 9.6.4.1 System interfaces: List each system interface and identify the functionality of the software to accomplish the system requirement and the interface description to match the system
* 9.6.4.2 User interfaces: Specify the logical characteristics of each interface between the software product and its users
* 9.6.4.3 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
* 9.6.4.4 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).
  + For each required software product, specify:
    - a) name;
    - b) mnemonic;
    - c) specification number;
    - d) version number; and
    - e) source.
  + For each interface, specify:
    - a) discussion of the purpose of the interfacing software as related to this software product;
    - b) definition of the interface in terms of message content and format. It is not necessary to detail any well-documented interface, but a reference to the document defining the interface is required.
* 9.6.4.5 Communications interfaces: Specify the various interfaces to communications such as local network protocols.
* 9.6.4.6 Memory constraints: Specify any applicable characteristics and limits on primary and secondary memory.
* operations;
* 9.6.4.7 Operations: Specify the normal and special operations required by the user such as:
  + a) the various modes of operations in the user organization (e.g., user-initiated operations);
  + b) periods of interactive operations and periods of unattended operations;
  + c) data processing support functions; and
  + d) backup and recovery operations.
* 9.6.4.8 Site adaptation requirements: The site adaptation requirements include:
  + a) definition of the requirements for any data or initialization sequences that are specific to a given site, mission or operational mode (e.g., grid values, safety limits, etc.);
  + b) specification of the site or mission-related features that should be modified to adapt the software to a particular installation.
* 9.6.4.9 Interfaces with services: Specify interactions with services, e.g., Software as a Service (SaaS) or cloud services.

The perspective cover the architecture of the system. The best way to accomplish is to explain in realms of MVC or any other architecture design pattern. The Perspective also establishes design boundaries such as interfaces (system, user, hardware, software), hardware requirements (memory, operations, site requirements, and services [network, platform]).

#### System Interfaces

9.6.4.1 System interfaces: List each system interface and identify the functionality of the software to accomplish the system requirement and the interface description to match the system

#### User Interfaces

9.6.4.2 User interfaces: Specify the logical characteristics of each interface between the software product and its users

#### Hardware interfaces

9.6.4.3 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

#### Software interfaces

9.6.4.4 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).

For each required software product, specify:

a) name;

b) mnemonic;

c) specification number;

d) version number; and

e) source.

For each interface, specify:

a) discussion of the purpose of the interfacing software as related to this software product;

b) definition of the interface in terms of message content and format. It is not necessary to detail any well-documented interface, but a reference to the document defining the interface is required.

#### Communication Interfaces

9.6.4.5 Communications interfaces: Specify the various interfaces to communications such as local network protocols.

#### Memory

9.6.4.6 Memory constraints: Specify any applicable characteristics and limits on primary and secondary memory.

operations;

.

#### Operations

9.6.4.7 Operations: Specify the normal and special operations required by the user such as:

a) the various modes of operations in the user organization (e.g., user-initiated operations);

b) periods of interactive operations and periods of unattended operations;

c) data processing support functions; and

d) backup and recovery operations.

#### Site Adaptations Requirements

9.6.4.8 Site adaptation requirements: The site adaptation requirements include:

a) definition of the requirements for any data or initialization sequences that are specific to a given site, mission or operational mode (e.g., grid values, safety limits, etc.);

b) specification of the site or mission-related features that should be modified to adapt the software to a particular installation.

#### Interfaces with Services

9.6.4.9 Interfaces with services: Specify interactions with services, e.g., Software as a Service (SaaS) or cloud services

### 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.

**Use cases, user stories and scenarios are also used to describe product functions.**

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.

Overall characteristics & features, non-formal, mapped from elicitation data.

A complete list of Product Functions should be listed here. This should be based off the Elicitation Techniques that will be reference in Section 2: Internal References.

### 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 9.6.9.

#### User Profiles:

Identify profiles of people that will use the product. User Profiles will be used to identify business markets

#### User Accounts:

Identify what type of accounts each User Profile will use to interact with the software product.

#### Storyboards:

Identify specific tasks action frames that user

#### Use Cases:

Identify cases or scenarios a user will use the software.

#### Workflows:

Identify a series of events/tasks/goals a user does to complete a solution to the problem the product overcomes.

### Limitations –

Provide a general description of any other items that will limit the supplier's options, including:

a) regulatory requirements and 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; and

m) limitations that are sourced from other systems, including real-time requirements from the

controlled system through interfaces.

what it will not do, storage & performance constrains, etc

## Definitions – titles, terms.

If you give a name to anything it should be defined here.

# References

## These are all documents that you will reference.

## Internal Reference

If not already done so, title your elicitation data reports from each team

The Raw elicitation data should be in other document(s), which is referenced in this section, it should not be in the SRS.

## External References

Define all inputs into and outputs from the software system. The description should complement the interface descriptions in 9.6.4.1 through 9.6.4.5, 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) data formats;

i) command formats; and

j) data items or information included in the input and output.

Any document that is referenced within the SRS, i.e.: IDE, Interface documents, specs, legal, etc., should be also listed

# Specification

## External interfaces

You reference the external interface, here you are specifying specifics about that interface.

Much like elicitation data, you are collecting and reporting information that will be used to create requirements.

You can include requirements in this section

## 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) hardware faults and failures; and

4) error handling and recovery;

d) effect of parameters;

e) relationship of outputs to inputs, including:

1) input/output sequences; and

2) formulas for input to output conversion.

It may be appropriate to partition the functional requirements into sub-functions or sub-processes. This does not imply that the software design will also be partitioned that way.

From the features or product functions, you will be deriving components of the system. These components will be named and are sub-system elements of the product. These components can be categorized as Front-End, Middleware, Backend, and Database. (Don’t use the category names, just mentioned).

## Usability Requirements

Define usability and quality in use requirements and objectives for the software system that can include measurable effectiveness, efficiency, satisfaction criteria and avoidance of harm that could arise from use in specific contexts of use.

More than HCI Usability, includes Interfaces and fulfilling the users needs

## Performance requirements

Specify both the static and the dynamic numerical requirements placed on the software or on human interaction with the software as a whole.

Static numerical requirements may include the following:

a) the number of terminals to be supported;

b) the number of simultaneous users to be supported; and

c) the 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 s**.

rather than, **An operator shall not have to wait for the transaction to complete**.

This is an expansion of the limitations with additional information.

Magic numbers are ok here as explanations to requirements

## Logical database requirements

**9.6.15 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) security; and

g) data retention requirements.

Backend and database components

## Design constraints

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

System and Component requirements dealing with End to End (Full-Stack)

## Software system attributes

9.6.18 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 the 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:

1) utilize certain cryptographic techniques;

2) keep specific log or history data sets;

3) assign certain functions to different modules;

4) restrict communications between some areas of the programme;

5) check data integrity for critical variables; and

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; and

5) use of a particular operating system.

Requirements that don’t fit neatly in the a single component.

## Supporting information

9.6.20 Supporting information: Additional supporting information to be considered includes:

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; and

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.

Kitchen sink, Summary of the Elicitation data

# Verification

### Verification uses similar specific wording.

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 9.6.10 to 9.6.18.

6.5.2 Requirements activities in verification

6.5.2.1 General: The purpose of the Verification Process is to provide objective evidence that a system or system element fulfils its specified requirements and characteristics.

6.5.2.2 Prepare for verification: This activity consists of the following task.

* Select appropriate verification methods or techniques and associated criteria for every verification action.
  + This activity is facilitated by initially associating a verification method as requirements are created. Verification methods should be documented. Documentation may include requirements verification and traceability matrix or verification statements in a verification plan. A verification method defines how (including success criteria and closure approach), where and when each requirement's compliance can be proven for acquirer acceptance. A verification method is associated with each requirement to define activities that yield objective information to prove satisfaction of the requirement. A good verification method definition addresses some or all of the following content considerations.
    - How – identify which verification method to be applied (see list below).
    - Who – identify the organization or person with the lead responsibility for performing the verification, such as a contractor, subcontractor, vendor, product team or supplier.
    - When – designate a time in the program plan when the verification is to be done. This should be an event-based, and not a calendar date, accomplishment.
    - Where – specify any unique venue and environment needed for the verification activity.
  + There are four standard verification methods to use to obtain the objective evidence that the requirements have been fulfilled: inspection, analysis or simulation, demonstration and test.
  + **Inspection** - an examination of the item against applicable documentation to confirm compliance with requirements. Inspection is used to verify properties best determined by examination and observation (e.g., - paint colour, weight, etc.). Inspection is generally non-destructive and typically includes the use of sight, hearing, smell, touch and taste; simple physical manipulation; mechanical and electrical gauging; and measurement.
    - Good practice: Include identification of the document(s) or drawing(s) to use to make the comparison between what is required versus what is being inspected.
  + Analysis (including modelling and simulation) - use of analytical data or simulations under defined conditions to show theoretical compliance. Used where testing to realistic conditions cannot be achieved or is not cost-effective. Analysis (including simulation) may be used when such means establish that the appropriate requirement, specification, or derived requirement is met by the proposed solution. Analysis may also be based on 'similarity' by reviewing a similar item’s prior verification and confirming that its verification status can legitimately be transferred to the present system element. Similarity can only be used if the items are similar in design, manufacture and use; equivalent or more stringent verification specifications were used for the similar system element; and the intended operational environment is identical to or less rigorous than the similar system element.
    - Good practice: Identify the generic name of the analysis (like Failure Modes and Effects Analysis), analytical or computer tools, or numeric methods; the source of input data; and how raw data is to be analyzed. Review and agree with the acquirer that the analysis methods and tools, including simulations, are acceptable for the provision of objective proof or requirements compliance.
  + **Demonstration** - a qualitative exhibition of functional performance, usually accomplished with no or minimal instrumentation or test equipment. Demonstration uses a set of test activities with system stimuli selected by the supplier to show that system or system element response to stimuli is suitable or to show that operators can perform their allocated functions when using the system. Observations are made and compared with predetermined responses. Demonstration may be appropriate when requirements or specifications are given in statistical terms (e.g., mean time to repair, average power consumption, etc.).
    - Good practice: State who the witnesses should be for the purpose of collecting the evidence of success, what general steps are to be followed, and what special resources are needed, such as instrumentation, special test equipment or facilities, simulators, specific data gathering, or rigorous analysis of demonstration results.
  + **Test** - an action by which the operability, supportability, or performance capability of an item is quantitatively verified when subjected to controlled conditions that are real or simulated. These verifications often use special test equipment or instrumentation to obtain very accurate quantitative data for analysis.
    - Good practice: State who the witnesses should be for the purpose of collecting the evidence of success. Identify the test facility, test equipment, any unique resource needs and environmental conditions, required qualifications and test personnel, general steps that are to be followed, specific data to be collected, criteria for repeatability of collected data, and methods for analyzing the results.
  + This information is included and documented in a Requirements Traceability Matrix (RTM) or a Verification Cross Reference Matrix (VCRM).

6.5.2.3 Manage results of verification

This activity consists of the following task.

* Maintain traceability of the verified [software] system elements.

6.5.3 Requirements activities in validation

6.5.3.1 General The purpose of the Validation process is to provide objective evidence that the system, when in use, fulfils its business or mission objectives and stakeholder requirements, achieving its intended use in its intended operational environment.

6.5.3.2 Prepare for validation: This activity consists of the following task.

* Identify the validation scope and corresponding validation actions.
* Identify the validation scope, including the characteristics of the software system, element, or artifact to be validated, and the expected results of validation.

6.5.3.3 Manage results of validation

* Maintain traceability of the validated system elements.

i.e.: The system shall allow users to withdraw cash

i.e.: Verify that the system shall allow user to withdraw cash

Next how will that happen

Provide a brief description of what needs to be verified.

Then provide additional detail on the information needed to do the verification.

i.e.: Verify that the system shall allow user to withdraw cash

User will enter verification information and request from the system specified amount, if user has sufficient funds, the system will then return to the user cash, and update user account amount.

If there sufficient funds, notified user, system will not return any cash

Verification section can be a separate section or a sub-section in the same section as the requirements. (Should map 1-to-1 or 1-to-n, avoid n-to-1, don’t use n-to-m)

# Appendices

## 5.1 Assumptions and dependencies

There are going to be some assumptions that you will not get from the user, stakeholders, and customer.

These usually are cross-referenced or supported by limitations and decisions that needed to be made outside of the elicitation data or other fore mentioned information.

## 5.2 Acronyms and abbreviations

TLAs’ home

1. <https://previews.123rf.com/images/thekaikoro/thekaikoro1709/thekaikoro170900060/87019401-close-up-shot-asian-woman-using-clear-tablet-for-futuristic-cyber-technology-concept-with-hud-gui-us.jpg> [↑](#endnote-ref-1)
2. <https://tv-fanatic-res.cloudinary.com/iu/s--sfH0d47X--/t_xlarge_l/cs_srgb,f_auto,fl_strip_profile.lossy,q_auto:420/v1490847642/protomolecule-phase-4-hybrid-the-expanse-season-2-episode-10.jpg> [↑](#footnote-ref-1)