Raul Bertone

Elis Haruni

Muyassar Kokhkharova

Saidar Ramazanov

Xhoni Robo

Total Team Effort: 23h30m

**Project Proposal - Fall-detection based on**

**acceleration & gyroscope data**

**A) Project Vision (Raul Bertone, Saidar Ramazanov, 2h30m)**

This project is realized as part of the course in Smart Sensor Network Systems for the summer semester 2018. It consists in the design and implementation of a fall-detection system based on acceleration and gyroscopic data. Mr La Blunda will act as project owner. [1]

The set-up has two main parts: the first consists of two Sensortags, which have to be worn around the waist of the test subject, that will gather the sensor data and send it over a Bluetooth connection to the Base Station for elaboration; the second part is the Base Station, a Bluetooth equipped PC which will run the application that will elaborate the sensor data, try to identify falls, and if necessary request help.

The project span is 7 weeks, the latest possible delivery date being June 22nd.

This is a standalone project, with no interaction with other groups or organizations, and no dependencies on other projects by this or other teams.

As the project is intended to develop the understanding of smart sensor networks and the technical understanding of their development process, the software itself is not the sole product. Every relevant document produced during the development, including but not limited to, this document, weekly individual reports by the team members, and a final report and presentation, will be part of the delivered artifacts.

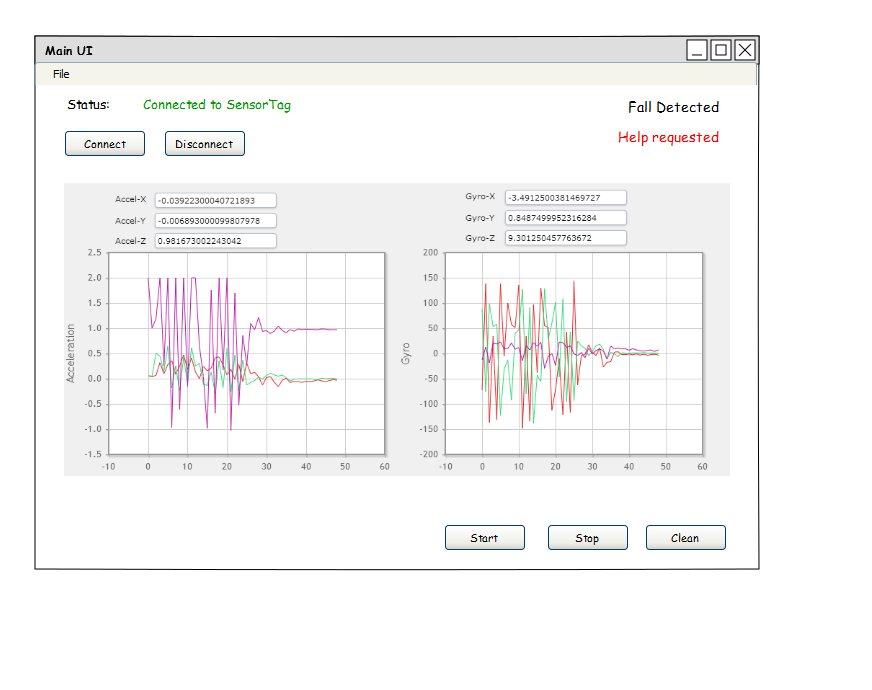
The following elements do not fall within the scope of this project and will not be included in the finished product:

* considerations on the hardware design of the wearable part
* a user manual
* maintenance and support of the product after initial delivery

**B) List of functional and non-functional requirements (Muyassar Kokhkharova, 12h)**

**Functional requirements**

1. Two CC2650 SensorTags are acting as peripherals and sending periodically acceleration and gyroscope data to the PC.



2. A PC application should be able to connect simultaneously to multiple peripherals via BLE.

3. A PC application should be able to receive sensor data (accelerometer & gyroscope) in real time.

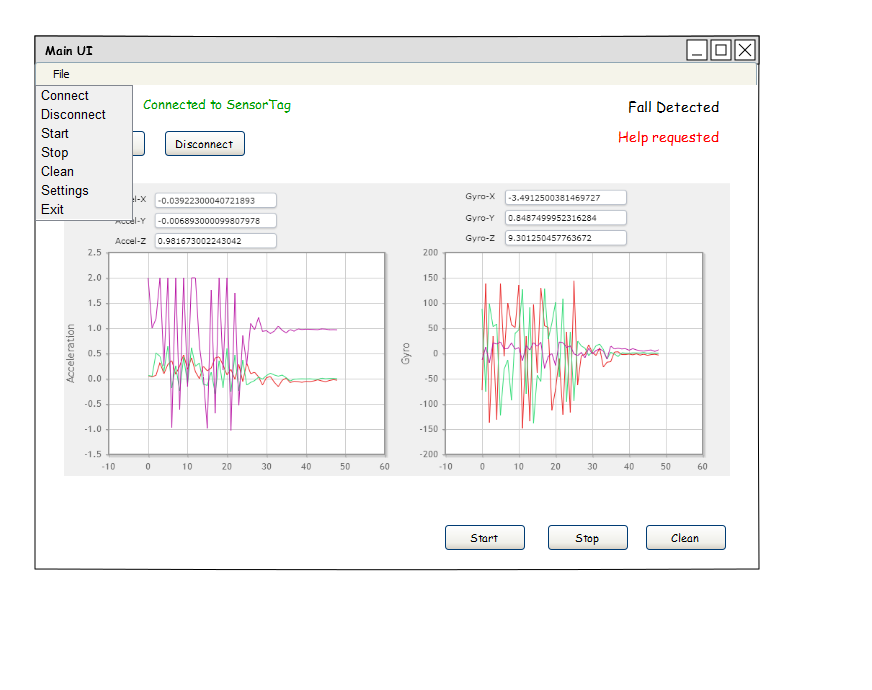
4. Data visualization. Line graphs with accelerometer and gyroscope data received from SensorTag.

5. The graph should depict the average value of received sensor data within the offset time.

6. Possibility to enter User‘s general information.

7. Possibility to calibrate system thus differentiate between sudden movements like walking the steps and free fall .

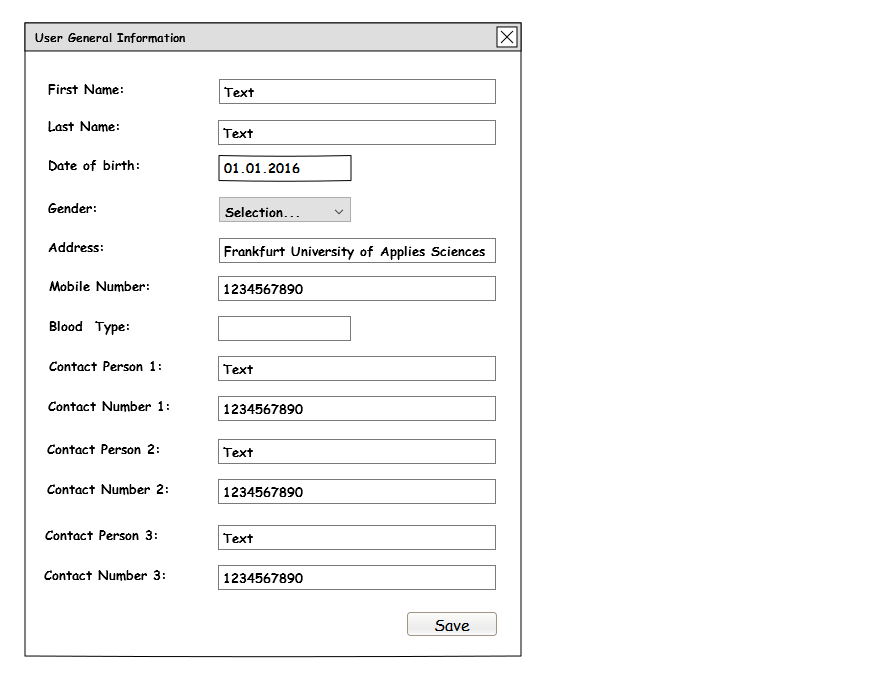
8. Main UI with basic control functions for operator working with a PC application.



**Non-functional requirements**

1. User general Information is a pop up window and it should contain:

- First name



- Last Name

- Date of Birth

- Gender

- Address

- Mobile number of user

- Blood type

- Contact Person1

In case of fall this Person will be contacted.

- Contact Number1

Phone number of a Contact Person.

- Contact Person2

In case of fall and if Contact Person1 is not replying this Person will be contacted.

- Contact Number2

Phone number of a Contact Person2.

- Contact Person3\*

In case of fall and if Contact Person1 and Contact Person2 are not replying this Person will be contacted.

- Contact Number3\*

Phone number of a Contact Person3.

- Save button to save changes

2. Application Settings is a pop up window containing following information:

2.1 Offset with default value of 0.25 seconds.

Graphs of gyroscope and accelerometer will be updated every offset time with average values of gyroscope and accelerometer received during offset time.

2.2 Delta for fall detection (gyroscope and accelerometer).

While falling values of accelerometer will gain acceleration and then will be equal to null. If delta of accelerometer received from sensor is bigger than configured delta in settings it will mean that the person is probably fell. The duration of free fall is also important because we have to differentiate between sudden movements like walking the steps and free fall, the duration will be different although delta values might be the same. While falling values of gyroscope will change and after reaching the ground values will be in horizontal position.

2.3 Set Default button

After clicking on it, all the default values will be set and Settings pop up window will remain open.

2.4 Save button

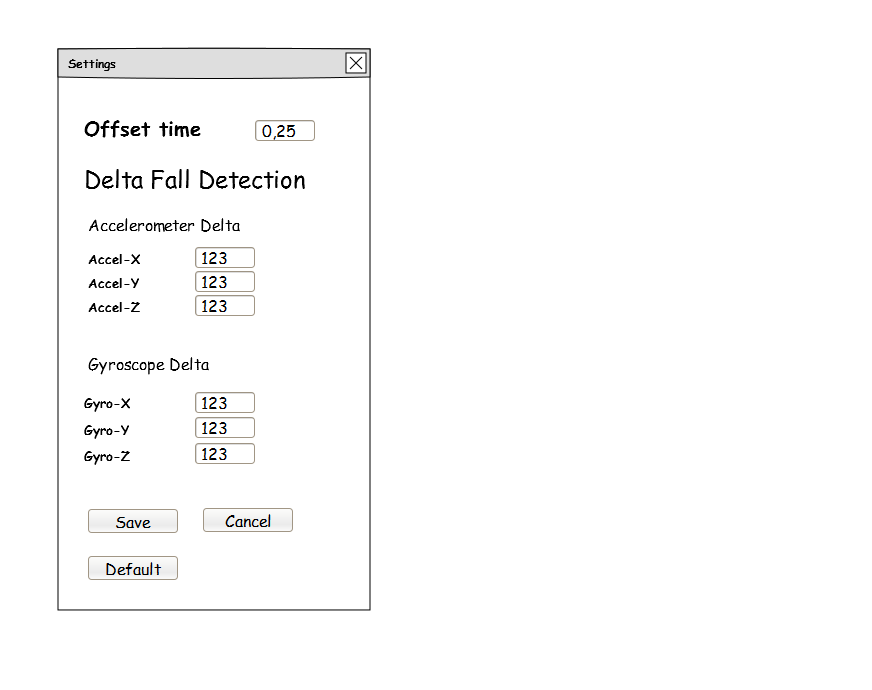
After clicking on save button, all the changes will be saved and Settings pop up window will remain open.

2.5 Cancel button

Cancel changes.

2.6 Close button

Clicking on close button will close Application Settings pop up window.



3. In Main UI:

3.1 Graphs with Accelerometer and Gyroscope data.

3.2 Accelerometer and Gyroscope data in the graphs will be updated at the same offset time.

3.3 Buttons connect/disconnect(to establish the bluetooth connection).

3.4 Buttons start/stop receiving gyroscope and accelerometer data.

3.5 Button Clear graph.

3.6 Label for Sensor Status (Connected to the sensor or No connection).

3.7 Label for Fall Detection.

If fall is detected inscription about falling will be displayed.

If not inscription «Fall was not detected» will be displayed.

3.8 Label for «help requested».

If fall was detected and user did not pressed the button on SensorTag to report about False alarm inscription on requesting help will be displayed.

3.9 When the Sensor is not connected to the PC Application(e.g. when Data is not being received), than Buttons Start, Stop and Clear should be deactivated\*

3.10 Stop button is Activated when Start is pressed.

4. „False alarm“. When device senses a fall it beeps first and if button on SensorTag is not pressed quickly it calls for help

**C) Safety, security and reliability requirements (Xhoni Robo, 2h)**

For an application whose sole purpose is the detection of a person falling, it is important to ensure that, at the very least, do what is required of it. However, it becomes of critical importance when paired with the fact that the user may be in danger following this fall. If this were a project made to be used for actual health benefits, failure to correctly assess when a person is in danger may even be fatal. Safety in this project translates to the protection of the hardware as well as software mechanisms that allow the fall to be detected. Preventing physical damage of the sensors is pretty self explanatory, and can be done by simply changing the design so that the fall itself would not be enough to break the sensors. As far as software is concerned, we need to ensure that the final application not only receives the data and correctly uses it, but also ensure that there are no interferences by other devices. In the case of low connectivity, the application should immediately notify the user. Lastly, the application should also ensure that it picks only the data received from the sensor tags. That way, there will be no issues with interference. Should anything not work as intended, the user should be notified immediately.

Once safety and security is ensured, the final application needs to also be reliable. The most basic reliability requirement is to prevent the application of notifying us of events that are similar to a fall, but that provide no danger to the user. This includes physical activities such as walking, running and even jumping. Below is the full list of safety, security and reliability requirements:

**Safety:**

* Software correctly notifies when a person has fallen
* Software correctly notifies user when one or both sensor tags do not work
* Software or SensorTags correctly notify user when connectivity is low
* If transmission stops abruptly during an activity similar to a fall, count that as a fall

**Security:**

* Data is collected from the associated SensorTags
* Other devices cannot send data to the application
* If any interferences are detected, notify the user

**Reliability:**

* Software differentiates between falls and other similar activities
* Software does not crash during long sessions where a lot of data is streamed
* Software notifies user when the SensorTags are low on battery
* SensorTags only send the necessary data. Other sensors should be disabled.

**D) Project Plan (Raul Bertone, 6h30m)**

**i) Project Estimation**

For the estimation of effort, the COCOMO II model was used [3], which was based on the value of Function Points [2].

**i-1) Function points**

The Function Points calculation process was conducted only until the Unadjusted Function Points values where obtained, because it is these values which are employed by the COCOMO II model.

In identifying the Application Boundary, we considered the two Sensortag devices and the PC application not as standalone systems, but as two of three modules that make up the complete application. As a consequence, the internal communication between the modules does not constitute a transaction; also, the complete system results stand-alone, and does not therefore possess External Interface Files.

**i-1.1) Transactions**

In the following table Transaction (External Input, External Output, External Inquiry) are listed.

They are subdivided according to the Actor that is responsible for them.

|  |  |  |  |
| --- | --- | --- | --- |
|  | External Input | External Output | External Inquiry |
| User | Insert system calibration (3FP)  Insert user general information (3FP)  Insert helper contact information (3FP)  Load defaults (3FP) | Accelerometer graph (4FP)  Gyroscope graph (4FP)  Label “Fall detection” (4FP)  Label “Help requested” (4FP) | Open application settings (3FP)  Start/Stop (3FP)  Connect /Disconnect (3FP)  Close button (3FP)  Clear graph button (3FP) |
| Helper |  | Send email (4FP) |  |
| Sensors | Gyroscope (3FP)  Accelerometer (3FP)  Snooze Alarm (Button 2) (3FP)  Bluetooth connection PC (4FP) Bluetooth connection Sensortags (4FP) |  |  |
| Actuators |  | Buzzer “false alarm” (4FP) |  |

**i-1.2) Internal Logical Files**

In the following table, ILFs are listed. They are subdivided according to the software module they

belong to.

|  |  |
| --- | --- |
| Module | Internal Logical Files |
| Sensortags | None |
| PC Application | User Information (7FP)  System calibration values (7FP)  Helper contact data (7FP) |

Total (unadjusted) Function Points: 89

**i-2) Estimation of Effort**

The estimation of effort was conducted with COCOMO II. Considered the early stage of

development of the project, the Early Design Model was selected.

**Scaling Drivers**

|  |  |
| --- | --- |
| *Driver* | *Value* |
| Precedentedness | High |
| Development Flexibility | Nominal |
| Risk Resolution | Nominal |
| Team Cohesion | High |
| Process Maturity | Very Low |

**Cost Drivers**

|  |  |
| --- | --- |
| *Driver* | *Value* |
| Facilities | Nominal |
| Personnel Experience | Nominal |
| Personnel Capability | High |
| Required Reusability | Low |
| Platform Difficulty | Nominal |
| Product Reliability and Complexity | Low |
| Required Development Schedule | Nominal |

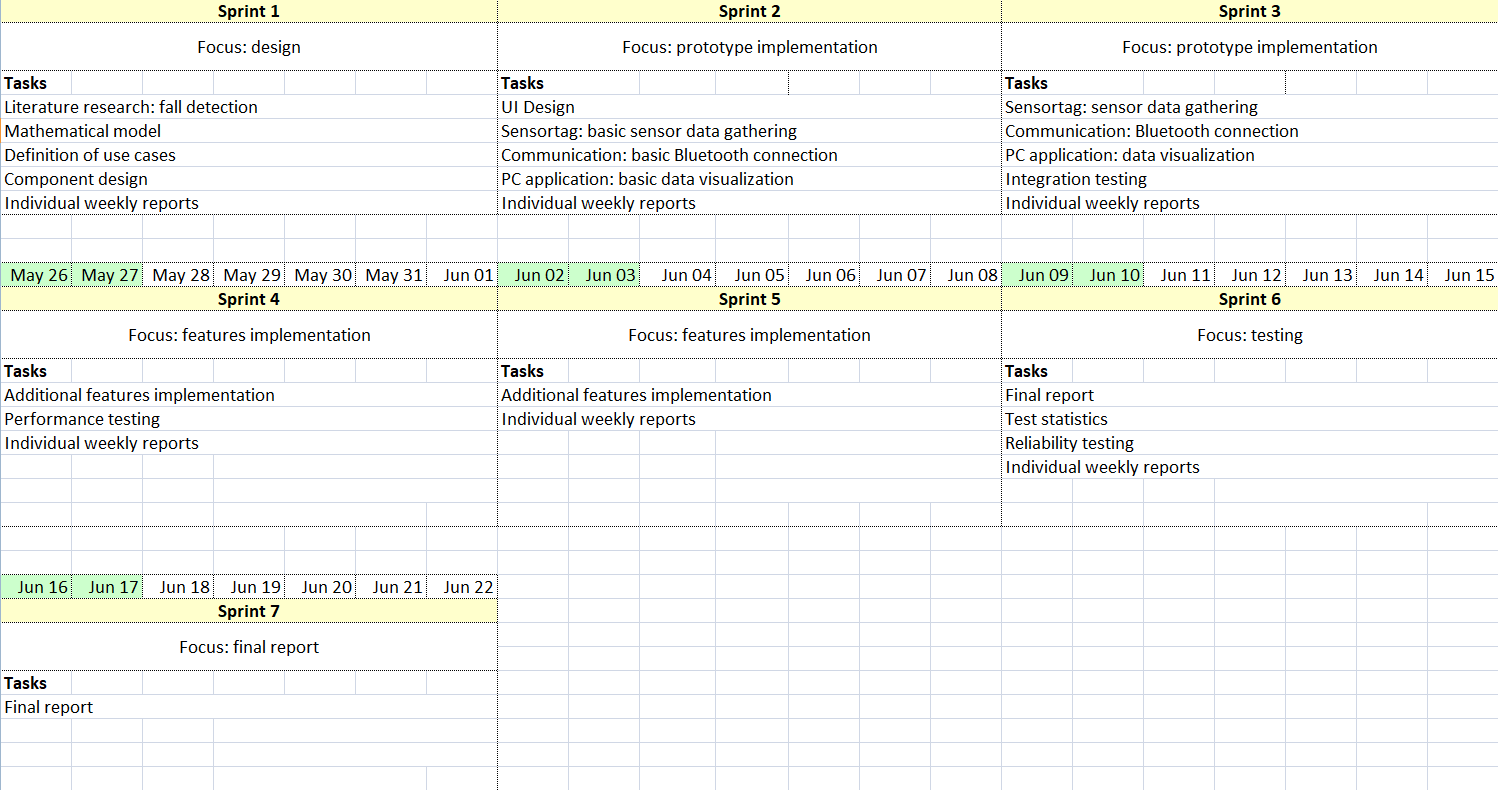
**Results**

|  |  |
| --- | --- |
|  | *Value* |
| Person-Months | 9.7 |
| Schedule Months | 1.9 |
| SLOC | 4717 |

**i-2.1)Notes**

The result of 1.9 months seems at first encouraging. However, our analysis must also consider that, on one hand, the team is working only part-time on the project, and on the other, that the final product will be a prototype, not a production ready system. In light of these facts, we believe the estimation to be generally accurate.

**ii) Project Scheduling (Raul Bertone, Saidar Ramazanov, 2h)**

****

**iii) Project Organization**

In consideration of the short time span of this project, and of the prototype nature of the final system, the team decided to employ agile development techniques, specifically Scrum. Sprints will have a duration of one week. Two scrum meeting will be held each week, tentatively on Wednesdays at 16:00 and Fridays at 15:00.

Official communication will be organized through two channels:

• for short or urgent messages and general coordination, the Slack “SSNS” group chat;

• for communication with the project owner, the forum “Group A” on Moodle, or per email

All common artifacts (source code, documentation, reference sources, etc.) are to be uploaded on the team’s GitHub repository.

**iv) Responsibilities of all team members**

All team members will assume several roles during the project. However, each person has been assigned a main role, making him or her the coordinator of all the individual efforts for a specific subject.

|  |  |
| --- | --- |
| Name | Main Role |
| Raul Bertone | Project Manager, Scrum Master |
| Elis Haruni | Lead Java Developer |
| Muyassar Kokhkharova | Statistics, UI Designer |
| Saidar Ramazanov | Mathematical Model |
| Xhoni Robo | Lead C Developer |

**v) Project risk analysis (Elis Haruni, 6h)**



****

**E) Safety and Security Plan (Xhoni Robo, 1h)**

Many of the requirements for safety and security can be considered by most to be common sense. However, they will be implemented in the later parts of the development process. This is due to the fact that a somewhat functional application can be made without these requirements in mind. However, the final application should include all the requirements. Most of them are easy to implement and should not take too much time. Even with setbacks, all the requirements should be implemented before the deadline. However, in case of some unexpected events, we need to prioritize them. The priority takes into account not only a requirements importance, but also the time it takes to implement it. This is done with an arbitrary scale in mind, and is so far only an estimate. Below is a list of the safety, security, and reliability requirements, ordered in the way that we initially plan to implement them:

1. Data is collected from the associated SensorTags
2. SensorTags only send the necessary data. Other sensors should be disabled.
3. Software correctly notifies when a person has fallen
4. Software correctly notifies user when one or both sensor tags do not work
5. Software or SensorTags correctly notify user when connectivity is low
6. Software notifies user when the SensorTags are low on battery
7. Software differentiates between falls and other similar activities
8. Software does not crash during long sessions where a lot of data is streamed
9. If transmission stops abruptly during an activity similar to a fall, count that as a fall
10. Other devices cannot send data to the application
11. If any interferences are detected, notify the user

**F) Setup and description of development environments**

*SensorTags*

The development for the SensorTags (and if necessary for the Launchpad) will be in C. For it we will use Texas Instrument’s own Code Composer Studio as well as the toolchain provided by Texas Instruments.

*PC application*

The PC application will be implemented in Java. For its development we will use Eclipse as an IDE, Junit for unit testing.

All mentioned software are available for both Linux and Windows, so we will leave the choice of an OS open for each team member (none makes use of MacOS). Additionally, the choice of Java as the implementation language allows the deployment and testing of the PC application on all OSs.

As a version control system software we selected *git,* and, based on this, GitHub as an online shared repository. All common artifacts (source code, documentation, etc.) will be uploaded there. In the root folder a readme file will describe the intended use of the different folders. The master branch is protected from accidental modifications by requiring the use of pull-requests for its update.

The repository can be found at the following address: <https://github.com/raulbertone/SSNS>

We integrated GitHub into our Slack group to be timely informed about commits performed by other team members.

**G) Literature References**

[1] M.Sc. L. La Blunda, "Project Proposal," [Online]. Available: https://moodle.frankfurt-university.de/pluginfile.php/512521/mod\_forum/attachment/94672/Project1.pdf

[2] A. J. Albrecht, "Measuring Application Development Productivity," in Proceedings of the Joint

SHARE, GUIDE, and IBM Application Development Symposium, Monterey, California, 1979.

[3] B. Boehm, "COCOMO II Model Definition Manual," [Online]. Available:

http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII\_modelman2000.0.pdf.