Design de Software

IoTApp
Software Architecture Document (SAD)

CONTENT OWNERS: Ye Yang (49521), João David (49448)

DOCUMENT NUMBER:	RELEASE/REVISION:	RELEASE/REVISION DATE:
• 1	• 1.0	• 30th December, 2019
•	•	•
•	•	•
•	•	•
•	•	•
•	•	•

Table of Contents

List of Figures

1 Views

1.1 IoTApp Module Decomposition View

1.1.1 Primary Presentation

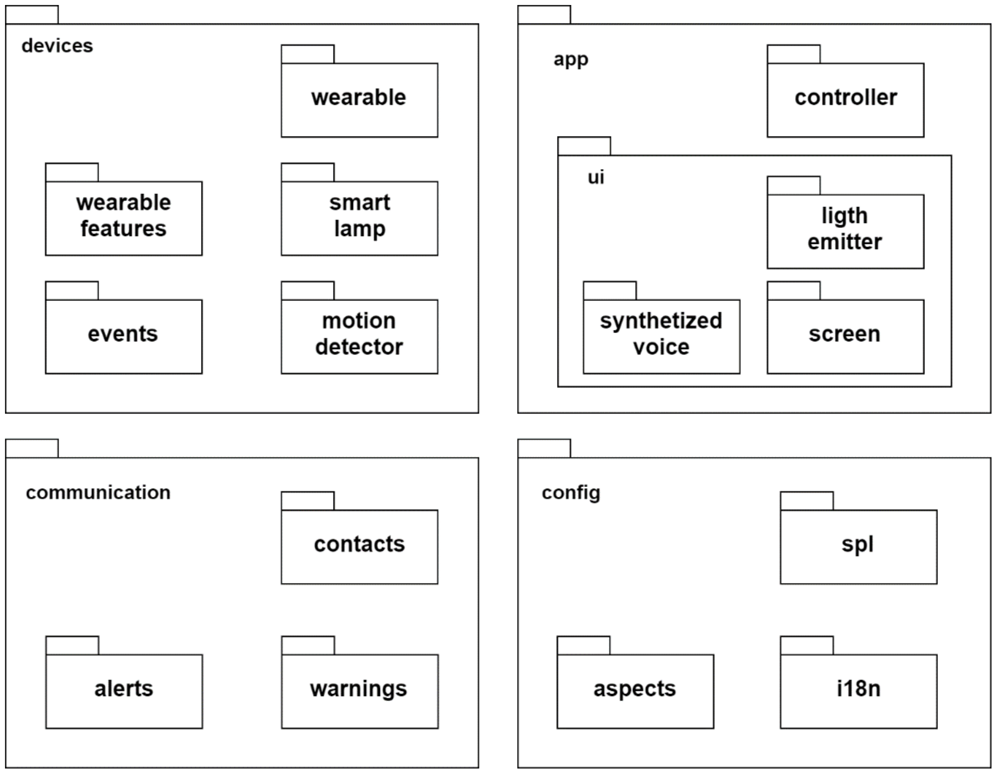

Figure 1 - IoTApp Module Decomposition View

- devices - this module contains all the devices supported by the app
  - wearables features – contains all the features that the wearable device can support, for instance, the button, the activity tracker or even the LED.
  - events – contains all the events that can be published/subscribed in order to establish communication between the app and the devices
- app – this module contains the logic behind the app itself
  - user interface – contains the variability related to way the app notifies its users, for instance, if this product is aimed at blind people, there will be a Synthetized Voice

    - controller – where the events published by the devices are being subscribed, and then processed based on the alerts defined by the user

- communication – this module contains the way the system triggers alerts and notifies the selected contacts
    - contacts – where the contacts are defined
    - alerts – where the alerts defined by the user are kept, for instance, an alert definition for detecting movement in the kitchen during a period of the night
    - warnings – where the warnings the user defines are tracked, for instance, a warning for alerting the user and a single contact that it's time to take the medicine
- config – the module where the configurations of the app are defined
    - aspects – contains the AspectJ files that allow the developers to build the different products of the SPL (Software Product Line)
    - spl – where is defined a simple program that parses the config XML file of the Feature Model and then generates an aspectj properties file that is used to configure the build
    - i18n – contains all the languages that are supported

1.1.2 Relations with Feature Model

- Idioma – The language feature is implemented with aspects within the i18n module
- Output – Both output features are implemented in the ui module, more specifically the light emitter and voice synthetized voice.
- Ecra – It's implemented within the screen module, and is used by all product on the SPL
- Alerta – Since all the alerts are optional, they are all implemented in the aspects module, which contains all the AspectJ files, which are weaved in during compilation time
- Aviso – It's implemented in the warning module
- ListaContatos/EnvioMensagem – These features are implemented in the contacts module, more specifically in the ContactList class

- DispositivoVestivel – All three of the wearable device features are implemented within the devices module, however since the devices are optional their behavior is implemented through aspects (i.e. Bezirk event subscriptions and message handling) in the aspects module

- LampadasInteligentes/DetetorDeMovimento – Similar to the wearable devices feature, these are also implemented as java classes in the devices module, and their behavior in aspects within the aspects module

1.2 IoTApp Module Uses View

1.2.1 Primary Presentation

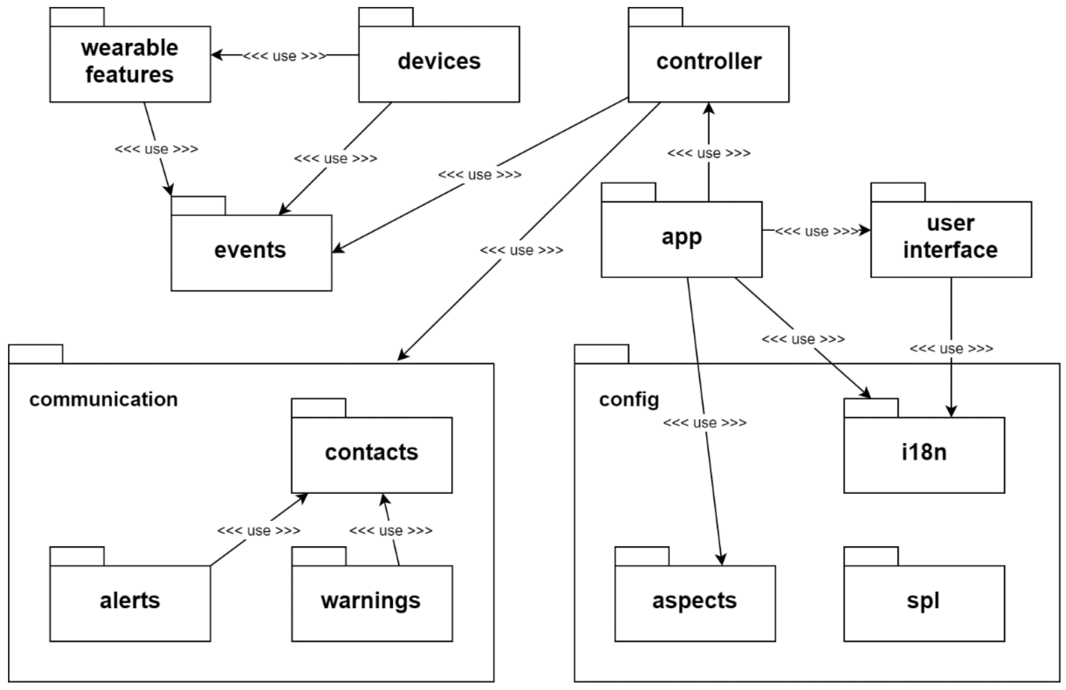

Figure 2 - IotApp Module Uses View

1.2.2 Element Catalog

- devices - this module contains all the devices supported by the app
    - wearables features – contains all the features that the wearable device can support, for instance, the button, the activity tracker or even the LED.
    - events – contains all the events that can be published/subscribed in order to establish communication between the app and the devices
- app – this module contains the logic behind the app itself
    - user interface – contains the variability related to way the app notifies its users, for instance, if this product is aimed at blind people, there will be a Synthetized Voice
    - controller – where the events published by the devices are being subscribed, and then processed based on the alerts defined by the user

- communication – this module contains the way the system triggers alerts and notifies the selected contacts
    - contacts – where the contacts are defined
    - alerts – where the alerts defined by the user are kept, for instance, an alert definition for detecting movement in the kitchen during a period of the night
    - warnings – where the warnings the user defines are tracked, for instance, a warning for alerting the user and a single contact that it's time to take the medicine
- config – the module where the configurations of the app are defined
    - aspects – contains the AspectJ files that allow the developers to build the different products of the SPL (Software Product Line)
    - spl – where is defined a simple program that parses the config XML file of the Feature Model and then generates an aspectj properties file that is used to configure the build
    - i18n – contains all the languages that are supported

1.3 Pub/Sub C&C View

1.3.1 Primary Presentation

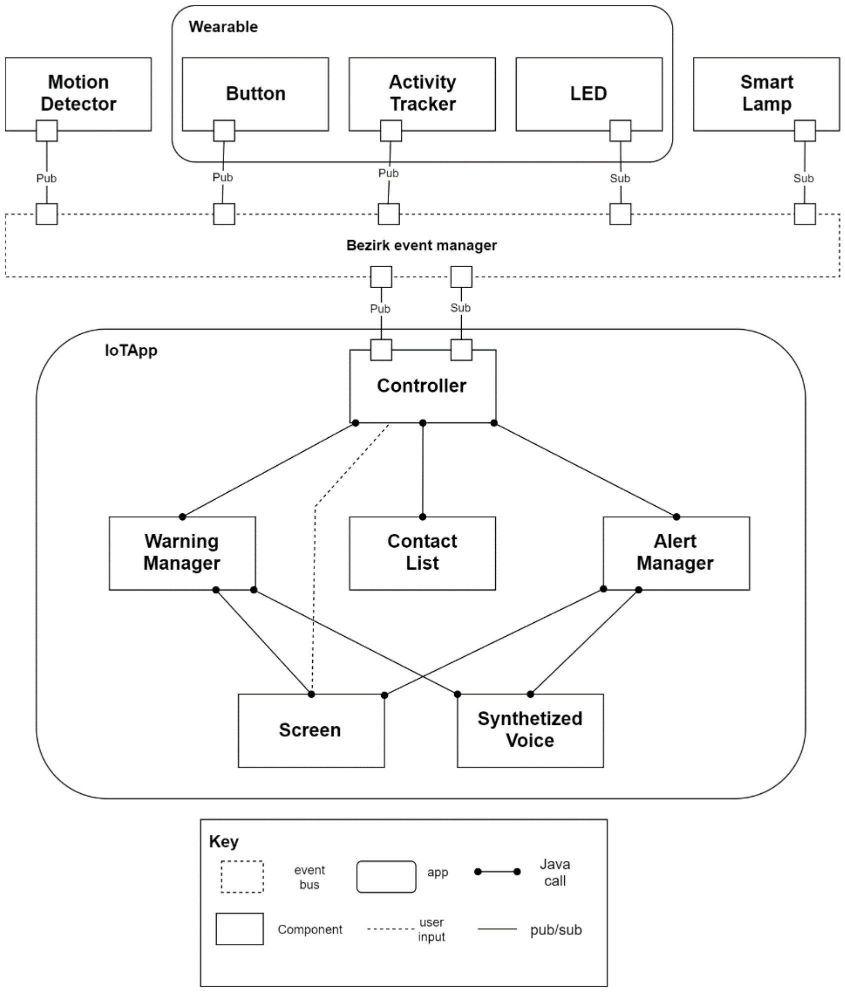

Figure 3 - Pub/Sub C&C View

1.3.2 Element Catalog

- MotionDetector/SmartLamp – When these features are not selected in the Feature Model, their events are not recognized by this specific product in the SPL

- Wearable – Similar to the MotionDetector/SmartLamp, the wearable device features' events are also not recognized if they are not selected in the Feature Model

- Bezirk event manager – This middleware api, allows the communication between several devices by handling publication and sending to their respective subscribers

- Controller – Acts as a information expert between the middleware and the rest of the application, each time it receives an event, it handles it based on the features this product has
- Warning Manager – Responsible for managing the warnings that the user has created
- Alert Manager - Responsible for managing the alerts that the user has created, since it is an optional feature, if it’s not selected in the Feature Model, the user won’t be able to create alerts
- Contact List – Mandatory feature of all product, keeps the information of all added contacts
- Screen – Also a mandatory feature, outputs app’s information and reads user input from the keyboard
- Synthetized Voice – Optional feature, when it’s selected in the Feature Model, the product form the SPL will “read” out loud the alerts/warnings outputted to the screen

Note: When in the Feature Model, the SinaisDeLuzes are selected, the controller will publish light events, so that the LED or Smart Lamp emit light signals

1.3.3 Variability

Since the communication between the devices and the IotApp itself is done through a pub/sub system, there’s no coupling between them, this allows the user to connect any device he wants, the only constraint being the app must recognize its events.

1.4 Product build C&C View

1.4.1 Primary Presentation

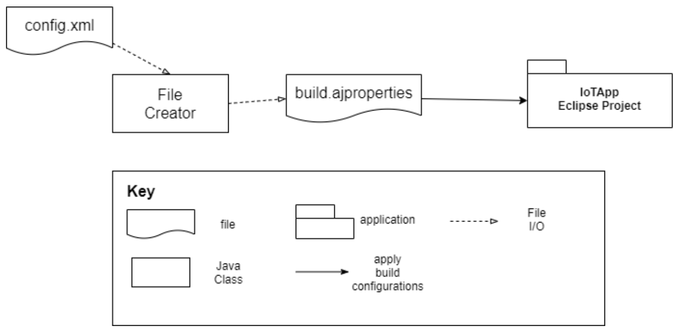

Figure 4 - Project Building C&C View

1.4.2 Element Catalog

- config.xml – this file is originated from the Feature Model build configuration; it contains the selections of the features that the user wants the application to have

- File Creator – this component is responsible for parsing the previous configuration file and outputting a new file that contains the aspects to be included and excluded on the application itself, managing the points of variability. The parser checks each ticked feature and excludes all the aspects not related to the selection.

- build.ajproperties – this is the output file of the file creator which contains the aspects to be excluded in relation to the features picked for this particular product

- IoTApp Eclipse Project – this component will execute this particular product in the SPL given the feature selection, it will run solely with the features selected initially by the user, this is guaranteed by using the previous build.ajproperties file as its build file, which will exclude all unwanted features for the current product.