Ciências ULisboa | Informática

Design de Software

Mestrados em Informática e Engenharia Informática

Exercício de Avaliação

2020/2021

O objetivo deste trabalho é fundamentalmente propiciar aos alunos uma experiência:

- na utilização dos estilos arquiteturais, nomeadamente familiarizá-los com as restrições que os estilos impõem nas fases subsequentes da construção do sistema;
- na conceção e implementação duma linha de produtos de software e também na utilização de diferentes mecanismos na gestão de variabilidade.

O trabalho deve ser resolvido em grupo com 2 ou 3 alunos. A entrega final é feita através da página da disciplina, até 24 de Janeiro de 2021. Antes de arrancar com a implementação, cada grupo deve marcar com o docente uma data para apresentação da sua solução de desenho. Após a entrega final cada grupo deve marcar uma data com o docente para fazer demonstração de alguns dos produtos desenvolvidos e discussão do trabalho realizado.

Introdução

Assistimos correntemente a um enorme crescimento no número e variedade de aparelhos IoT disponíveis no mercado. Estes aparelhos incluem, por exemplo:

- lâmpadas inteligentes
- tomadas inteligentes
- botões inteligentes

e uma miríade de sensores, por exemplo,

- detetores de movimento e de presença,
- monitores de dados biométricos (ex., nível de atividade, batimento cardíaco, taxa de respiração)
- monitores de condições ambientais (ex., temperatura, pressão, humidade, poluentes químicos e físicos)

Este fenómeno tem vindo a ser acompanhado por uma explosão de aplicações que tiram partido destes dispositivos em diferentes áreas. Exemplos variados são apresentados, por exemplo, em http://www.postscapes.com/internet-of-things-examples/. Porque a maioria destes produtos é chave-na-mão, a combinação de funcionalidades conseguidas com o mesmo tipo de dispositivos implica adquirir e instalar dispositivos em duplicado, ou triplicado, etc. Esta limitação é uma oportunidade de negócio.

Problema

Suponha que uma empresa pretende explorar esta oportunidade de negócio e decidiu começar por desenvolver uma linha de aplicações IoT para residências assistidas e também para residências de pessoas que necessitem de apoio por serem portadores de deficiência. A opção por uma linha de produtos foi

considerada a melhor forma de ter, a baixo custo, uma oferta de produtos, em termos de funcionalidade e preço, alinhada com as necessidades de diferentes tipos de clientes, com diferentes necessidades.

Após um estudo de mercado, a empresa definiu os seguintes requisitos para a sua linha de produtos.

- Os produtos da linha são aplicações IoT que são lançadas a partir de um computador pessoal com ligação à internet e a uma rede *wifi*, onde serão ligados todos os aparelhos IoT necessários ao funcionamento de cada aplicação.
- Há produtos da linha específicos para o mercado de língua portuguesa e inglesa.
- Há produtos da linha específicos para utilizadores com problemas de audição e com problemas de visão. Mais especificamente:
  - Em todos os produtos da linha, há dados de input que são recebidos através do teclado e rato do computador pessoal e o ecrã é usado para output (isto implica que, no caso de produtos para utilizadores com problemas de visão, a utilização do sistema em certas situações poderá exigir a presença de um acompanhante).
  - Os produtos da linha para utilizadores com problema de visão suportam alguns comandos por voz e usam adicionalmente voz sintetizada para comunicar com os utilizadores.
  - Os produtos da linha para utilizadores com problemas de audição recorrem adicionalmente a sinais luminosos para comunicar com os utilizadores. Estes sinais servem apenas para chamar a atenção do utilizador, sendo a informação específica dada no ecrã. Os sinais luminosos podem ser emitidos por lâmpadas inteligentes ou por um indicador luminoso de um qualquer dispositivo vestível.
- Todos os produtos da linha mantêm uma lista de contactos e permitem gerir esta lista. Os contactos são usados para o envio de mensagens por *sms*. Os produtos da linha que emitem alertas forçam a definição de um contacto de urgência.
- Todos os produtos da linha permitem criar e apagar *avisos*. Ao definir um aviso, o utilizador tem de indicar o nome do aviso, a data de início, a data de fim, a periodicidade com que o aviso deve ser feito e, opcionalmente, um dos seus contactos. O utilizador do sistema e o contacto indicado recebem os avisos.

  Exemplo de definição de aviso:

  - Tomar antibiótico de 2020-12-01|08:00 até 2020-12-20|20:00 de 8 em 8 horas com conhecimento da Maria

  Exemplo de aviso:

  - 08:00 - Tomar antibiótico
- Os produtos que têm associados detetores de movimento ou dispositivos vestíveis com monitor de atividade permitem definir padrões que devem despoletar o envio de um alerta. Estes alertas podem ser definidos para a deteção de movimento num determinado período do dia numa determinada divisão da casa ou pela inatividade durante um determinado número de minutos num determinado período do dia.

  Exemplos de definição de padrões:

  - inatividade durante 45mn no período [09:00,20:00]
  - deteção de atividade na cozinha no período [23:00,07:00].

  Exemplo de alerta:

  - Deteção de atividade - Cozinha - 02:30.
- Os produtos que têm associado dispositivos vestíveis com funcionalidade de botão, quando pressionados, despoletam o envio de um alerta com um pedido de ajuda. O próprio utilizador é também avisado que está a emitir um pedido de ajuda.

Exemplo de alerta:

- Acionado pedido de ajuda 17:32 2020-12-01

Foi também já decidido que:

- De forma a facilitar o desenvolvimento das aplicações, nomeadamente a interoperabilidade entre diferentes dispositivos, será usado o *middleware Berzik* (fornecido no material de apoio).
- A aplicação deve ficar dependente apenas de software de acesso livre.

Tarefas

Pretende-se que conceba e implemente uma primeira versão do software desta linha de produtos. Nesta versão não vai ser preciso ainda lidar com a incorporação de dispositivos de hardware concretos. Uma vez que foi tomada a decisão de usar o *Berzik,* é preciso apenas definir as interfaces apropriados dos elementos que encapsulam a comunicação com os dispositivos de hardware que irão estar envolvidos e fornecer implementações *mock* dos mesmos.

Mais especificamente devem realizar as seguintes tarefas:

1. Conceber um modelo de características (*feature model*) para a linha de produtos que permita à empresa ter uma oferta de produtos tão diversificada quanto possível. Todos os requisitos descritos na página anterior devem ficar cobertos, mas podem eventualmente considerar outros. Não se esqueçam de explicitar as restrições aplicáveis à combinação das funcionalidades constantes nesse modelo.
2. Apresentar um quadro com alguns dos produtos da linha e as suas características, incluindo os quatro produtos que decidirem implementar (ver ponto 4. abaixo).
3. Conceber uma solução de desenho para a linha de produtos, que
   - utilize o *Bezirk* para tratar da comunicação com os dispositivos de *hardware*
   - utilize *aspectos* para gerir a variabilidade, em pelo menos alguns pontos de variação
   - utilize *injeção de dependências* se, e onde, isso for apropriado

   e documente a arquitetura da sua solução, nomeadamente através de:
   - vistas de módulos que, além do que é comum neste tipo de vistas, explicitem a correspondência entre os módulos e as *features* do modelo
   - vistas de componentes e conectores que forneçam o contexto, explicitem os pontos de variabilidade e documentem os mecanismos usados para concretizar a variabilidade suportada
4. Implementar um protótipo da linha de produtos que permita demonstrar **quatro produtos** diferentes da linha de produtos. Estes quatro produtos devem incluir:
   - um produto para o mercado português
   - um produto para o mercado inglês
   - um produto específico para utilizadores com problemas de visão
   - um produto específico para utilizadores com problemas de audição
   - um produto que tem associado um dispositivo vestível com monitor de atividade
   - um produto que tem associado um dispositivo vestível com funcionalidade de botão

   Note que o seu código não precisa de implementar todas as funcionalidades. Algumas classes podem ter apenas um esqueleto ou serem apenas *mocks*.

   A implementação deve ser fornecida na forma de um projeto Eclipse contendo:
   - O código, bibliotecas, e outros recursos necessários para algum dos produtos da linha.

- Algo que permita facilmente construir e executar cada um dos 4 produtos escolhidos. Idealmente, a construção desses produtos deve incluir apenas os módulos de código estritamente necessários para concretizar as suas características (ou seja, não deve ter *dead code*). Pode, por exemplo, usar ficheiros de configuração do build ou implementar uma DSL que permita definir os produtos da linha.
- Um README com a descrição do projeto
- Um documento com eventuais decisões de implementação que considerar relevante documentar

Informação Adicional

- O middleware *Berzik* foi desenvolvido pela *Bosch* e disponibilizado para utilização livre em 2016. Entretanto, no início de 2017, a informação sobre o projeto em http://developer.bezirk.com/ e o código em https://github.com/Bezirk-Bosch/AdapterZirks deixou de estar disponível, pelo que a informação necessária para se familiarizarem com o *Berzik* é fornecida no material de apoio ao trabalho.
- Os projetos exemplo do *Berzik* usam o *gradle*. Como o uso simultâneo do *gradle* e *AspectJ* no Eclipse é potencialmente problemático, é fornecido um projeto exemplo que não precisa do *gradle* e que usa o *AspectJ*. O projeto é uma adaptação do exemplo descrito no tutorial 2 do *Berkiz* (incluído no material de apoio) e envolve:
  - um *mock* de um ***Air Quality Sensor Zirk*** que publica dados sobre a qualidade do ar — humidade, nível de poeiras e nível de pólens— que num sistema real serão recebidos de um sensor de hardware;
  - um ***Asthma Assistant Zirk*** que subscreve os dados de qualidade do ar e usa-os para dar instruções ao utilizador de como proceder (por exemplo, ligar o desumidificador);
  - um conjunto de classes, aspectos e outros recursos que permitem construir uma variante do sistema em que as instruções ao utilizador são dadas em português e outra variante em que são dadas em inglês

  Para correr um dos sistemas, depois de escolher a *build configuration* apropriada, deve proceder como está descrito no tutorial 2 (**importante**: precisa de estar ligado a uma rede *wifi*).

  Utilize o código fonte do *berzik-middleware-api* para dentro do Eclipse ter acesso ao seu *javadoc*.
- Para demonstrar um produto que exija dispositivos de hardware tem apenas de programar *mocks* apropriados. Se não quiser só usar *mocks*, pode usar a câmara do computador como um detetor de movimento. Para ficar com uma ideia de como, recorrendo a adaptadores, é suportada no *Berzik* a integração de dispositivos de hardware, por exemplo com as lâmpadas inteligentes da Philips, pode olhar para o projeto *AdapterZiker/PhilipisHue* fornecido no material de apoio. Pode aproveitar ainda para olhar para os eventos que estas lâmpadas publicam e subescrevem de forma a que lhe sirvam de inspiração para os tipos de eventos que vai considerar na sua família.
- Para demonstrar um produto que exija síntese de fala pode considerar o inglês e usar por exemplo o framework *FreeTTS* http://freetts.sourceforge.net/docs/index.php ou por exemplo esta API Java para usar os serviços da Google https://github.com/goxr3plus/java-google-speech-api
- Para demonstrar um produto que exija envio de mensagens de sms não precisa de concretizar o envio efetivo das mensagens, mas experimentar pode por exemplo usar o Twilio https://www.twilio.com/docs/sms/quickstart/java