

## 7

### Conclusões

Este trabalho descreveu a arquitetura e a implementação do *middleware* Kaluana. Esse *middleware* tem como objetivo permitir a realização de adaptações e implantações dinâmicas em aplicações móveis, utilizando um modelo baseado em componentes e orientado a serviços.

Kaluana simplifica o desenvolvimento de aplicações móveis por se aproveitar de características de modelos orientados a componentes, como reuso e fraca amarração entre componentes, aliadas a características de modelos orientados a serviços, como a flexibilidade na implementação.

O *middleware* foi construído e testado sobre a plataforma Android, que provê facilidades como orientação a serviços e comunicação assíncrona. É possível implementar o *middleware* Kaluana sobre outra plataforma orientada a serviços, por exemplo OSGi, citada no capítulo 3.

A implementação foi validada por meio de testes integrados, fundamentais para a implementação do *middleware*, e pelo desenvolvimento de aplicações de exemplo, que se utilizam da capacidade de adaptação e implantação dinâmica oferecidas pelo *middleware* Kaluana. O desenvolvimento dessas aplicações demonstrou a simplicidade na criação de aplicações adaptáveis.

O estudo de caso implementado definiu um mecanismo de adaptação baseado em regras descritas pelo desenvolvedor da aplicação em função da localização do dispositivo. O mecanismo pode ser estendido de modo a utilizar outras fontes de contexto, tais como nível de energia, tipos e qualidade da conexão *wireless* ou inclinação do dispositivo.

São propostos, como trabalhos futuros, extensões e melhorias na implementação do *middleware* para torná-lo mais abrangente, bem como a sua integração à plataforma Mobilis [2], que resultará em um *middleware* para desenvolvimento de aplicações colaborativas e adaptáveis em função do contexto.

#### 7.1

##### Trabalhos futuros

Os trabalhos futuros dividem-se em extensões do *middleware* Kaluana e variações em sua implementação.

### 7.1.1

#### Integração à plataforma Mobilis

A plataforma Mobilis possibilita o desenvolvimento de aplicações colaborativas e sensíveis ao contexto, e provê integração entre as aplicações construídas sobre sua arquitetura.

As aplicações Mobilis, como são chamadas, podem ser executadas independentemente umas das outras, e assemelham-se aos componentes do *middleware* proposto neste trabalho pois podem conectar-se a serviços providos pelo *middleware* ou por outras aplicações.

Com a integração ao *middleware* Kaluana, o desenvolvimento de aplicações Mobilis seria reduzido à composição de componentes implantáveis e reutilizáveis, tornando clara a separação entre o desenvolvimento dos componentes e sua composição, além de tornar aplicações colaborativas, já sensíveis a contexto, adaptáveis segundo este contexto.

### 7.1.2

#### Gatilhos de adaptações dinâmicas

O *middleware* Kaluana é capaz de realizar adaptações dinâmicas em aplicações móveis. O mecanismo desenvolvido como exemplo, para fins de teste, baseou-se em informações sobre a localização do dispositivo móvel.

Outras informações, contudo, poderiam determinar a necessidade de uma adaptação dinâmica, como a qualidade da conexão, que poderia substituir um componente de *streaming* de vídeo para um de imagens estáticas, ou a energia disponível, que poderia fazer com que uma aplicação não utilize recursos de localização frequentemente.

### 7.1.3

#### Definição de tipos de componentes

A tipagem dos componentes Kaluana em função de suas categorias é definida somente em tempo de execução. Torna-se necessário criar um mecanismo que, a partir da interface ou do tipo do componente, realize a busca de todas as implementações disponíveis para tal interface.

### 7.1.4

#### Descrição do estado interno

A implementação atual de suporte à persistência do estado interno hoje somente dá suporte à persistência de tipos básicos da linguagem Java, além de assumir que desenvolvedores de componentes conheçam o mecanismo utilizado

para a persistência para que possam recuperar a informação persistida, como mostrado na seção 4.5.2.

Uma futura versão do *middleware* deve implementar um mecanismo mais eficiente, para que informações persistidas possam ser recuperadas sem que o desenvolvedor precise conhecer em detalhes o mecanismo de persistência.

### 7.1.5

#### **Critérios de seleção de componentes e serviços**

Kaluana assume que existe um componente de determinado tipo ou uma implementação para determinado serviço mais adequada a cada condição de contexto. Entretanto, é possível criar mecanismos mais complexos para escolha do componente ou serviço a ser ativado pelo *middleware* a partir do tipo ou interface, respectivamente.