HAC Integration Layer

Architecture Notebook

# Purpose

This document describes the philosophy, decisions, constraints, justifications, significant elements, and any other overarching aspects of the system that shape the design and implementation.

[Always address Sections 2 through 6 of this template. Other sections are recommended, depending on the amount of novel architecture, the amount of expected maintenance, the skills of the development team, and the importance of other architectural concerns.]

# Architectural goals and philosophy

O principal objetivo da arquitetura do HAC Integration Layer, é ser bem definida permitindo uma fácil compreensão de seu funcionamento e uma fácil manutenção para desenvolvimentos futuros. Na arquitetura do projeto, está sendo levada em conta também a necessidade de, futuramente, aplicar seu funcionamento em um ambiente real, aonde o sistema operacional de tempo real não será virtualizado. Para tanto, está sendo evitada a dependência entre a camada de integração e sistema operacional host.

# Assumptions and dependencies

## Assumptions

* + - * + Essa arquitetura assume a implementação das funções da LibC necessárias e inclusive as funções matemáticas no HellfireOS.

|  |  |
| --- | --- |
| HAC Integration Layer |  |
| Architecture Notebook | 14/05/2015 |

## Dependencies

* + - * + **COMPaaS**: Caso haja mudanças na API de comunicação do COMPaaS, alterações na arquitetura podem ser necessárias.
        + **HellfireOS**: Caso haja mudanças no suporte à rede do HellfireOS, alterações na arquitetura podem ser necessárias.

# Architecturally significant requirements

1. Arquitetura deve ser de fácil atualização e manutenção.
2. A arquitetura deve ser divida em três camadas.
3. A camada de dados provida deve ser baseada em banco de dados relacional.
4. A camada de lógica de negócio deve ser baseada em objetos.
5. A camada de apresentação poderá ser implementada em duas alternativas:
   1. Cliente Web, utilizando-se um navegador; ou
   2. Cliente dedicado, utilizando-se uma aplicação cliente.

# Decisions, constraints, and justifications

## Interface Gráfica

A Interface gráfica que será utilizada será um cliente Web de testes, desta forma será possível validar a arquitetura e o funcionamento da integração sem que seja investido muito tempo na criação das telas que para a implementação deste projeto, são apenas para teste e validação.

# Architectural Mechanisms

## Gerência de Erros

Prover a capacidade de detectar e reportar os erros.

## Debugging

Prover elementos para possibilitar o debug da aplicação.

# Key abstractions

[List and briefly describe the key abstractions of the system. This should be a relatively short list of the critical concepts that define the system. The key abstractions will usually translate to the initial analysis classes and important patterns.]

# Layers or architectural framework

* **Camada 1**: A camada de mais baixo nível será composta pelo HellfireOS e seus respectivos sensores, gerando dados que serão enviados pela camada 2.
* **Camada 2**: Será a camada responsável pela integração entre as camadas 1 e 3, denominada HAC.
* **Camada 3**: Será composta pelo modulo COMPaaS, receberá dados da camada 1, alimentando sua base de dados e possibilitando o uso de informações em sua camada de negócio.

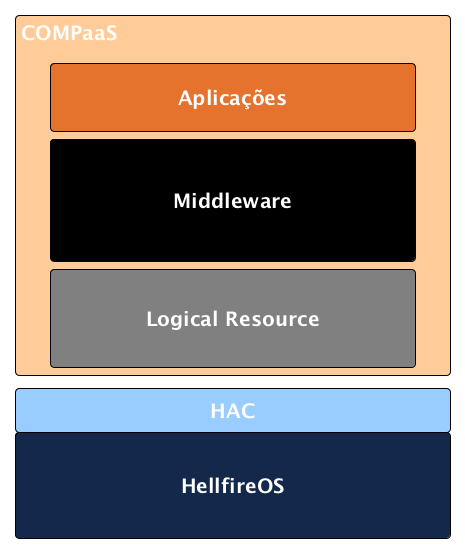


Figura1: Disposição das camadas da arquitetura.

# Architectural views

[Describe the architectural views that you will use to describe the software architecture. This illustrates the different perspectives that you will make available to review and to document architectural decisions.]

|  |  |
| --- | --- |
| HAC Integration Layer |  |
| Architecture Notebook | 14/05/2015 |

## Recommended views

* **Logical:** Describes the structure and behavior of architecturally significant portions of the system. This might include the package structure, critical interfaces, important classes and subsystems, and the relationships between these elements. It also includes physical and logical views of persistent data, if persistence will be built into the system. This is a documented subset of the design.
* **Operational:** Describes the physical nodes of the system and the processes, threads, and components that run on those physical nodes. This view isn’t necessary if the system runs in a single process and thread.
* **Use case:** A list or diagram of the use cases that contain architecturally significant requirements.