

SmartPark

Arturo Fonseca de Souza e Lucas Vinícius Góis Nogueira

Instituto Metrópole Digital - IMD / Universidade Federal do Rio Grande do Norte - UFRN

Resumo

Neste trabalho especificamos os requisitos do sistema SmartPark: um Sistema de Estacionamento que visa atender as diversas demandas e problemáticas de que estacionamentos passam atualmente. Munido de um aplicativo informativo e de dispositivos de reconhecimentos de placas de veículos, definimos um sistema robusto e que pretende trazer tanto uma melhor experiência para usuários de estacionamento como um controle e aumento de lucros da parte da gerência deles.

Palavras-chave: estacionamento, arquitetura de software

1. Introdução

Com o crescente aumento na quantidade de veículos nas estradas e áreas urbanas, a gestão eficiente de estacionamentos se tornou uma necessidade emergente. A disponibilidade de áreas exclusivas para estacionamento está se tornando progressivamente escassa, levando os proprietários de veículos a estacionar nas vias públicas, à medida que a procura por estacionamentos continua a aumentar. Nesse cenário, a implementação de um Sistema de Estacionamento é crucial para otimizar a utilização dos espaços disponíveis, melhorar a experiência dos motoristas e reduzir o congestionamento nas áreas urbanas.

Um Sistema de Estacionamento é um conjunto de tecnologias e processos que visa facilitar o estacionamento de veículos de forma organizada, eficiente e segura. Esses sistemas variam desde soluções simples, como barreiras automáticas e sinalização, até sistemas mais avançados que empregam sensores,

câmeras, aplicativos móveis e algoritmos inteligentes para otimizar a gestão do estacionamento.

Em resumo, um Sistema de Estacionamento desempenha um papel vital na otimização da mobilidade urbana e na oferta de uma experiência de estacionamento mais eficiente e conveniente. Esta introdução é apenas o ponto de partida para uma exploração mais aprofundada do mundo dos sistemas de estacionamento e das inovações que estão transformando a maneira como lidamos com o estacionamento de veículos.

2. Descrição do problema

Alguns problemas comuns encontrados em estacionamentos incluem uma série de desafios que afetam tanto os motoristas, operadores de estacionamento e investidores (donos do estacionamento, acionistas, etc.). Esses problemas podem variar de acordo com o tipo de estacionamento e o ambiente em que estão localizados. Abaixo estão listados alguns dos problemas mais comuns:

- 1. Escassez de Vagas:** A falta de vagas de estacionamento é um problema frequente, especialmente em áreas urbanas congestionadas. Os motoristas muitas vezes enfrentam dificuldades para encontrar um local para estacionar, resultando em desperdício de tempo e combustível.
- 2. Estacionamento Desorganizado:** A falta de sinalização adequada, pinturas no chão e orientações claras pode levar a um estacionamento desorganizado, onde os motoristas estacionam de maneira irregular ou bloqueiam outros veículos e, por consequência, causando transtorno ao ambiente como um todo.
- 3. Filas e Congestionamento na Entrada/Saída:** A entrada e a saída de estacionamentos muitas vezes se tornam pontos de congestionamento devido à falta de automação ou de pessoal para gerenciar o fluxo de veículos.
- 4. Falta de Segurança:** Roubos, vandalismos e outros problemas de segurança podem ocorrer em estacionamentos mal vigiados. A segurança dos veículos e dos ocupantes é uma preocupação constante.
- 5. Falta de Pagamento e Cobrança Eficiente:** Sistemas de pagamento

ineficientes, falta de opções de pagamento eletrônico e atrasos na cobrança podem frustrar os motoristas e reduzir a receita para os operadores de estacionamento.

6. Uso Ineficiente de Espaço: Em alguns casos, os espaços de estacionamento são mal dimensionados, resultando em subutilização de áreas valiosas e perda de receita para os operadores.

7. Má Gestão de Demandas Flutuantes: Alguns estacionamentos enfrentam flutuações significativas na demanda, tornando difícil a gestão eficiente de recursos.

Para enfrentar esses problemas comuns, muitos estacionamentos estão recorrendo a soluções de automação, como sistemas de gerenciamento de estacionamento, câmeras de segurança, sensores de ocupação de vagas e aplicativos móveis. Essas tecnologias podem melhorar a eficiência, a segurança e a satisfação dos usuários, contribuindo para uma experiência de estacionamento mais agradável e eficaz.

Nesta descrição, vimos as problemáticas presentes em um estacionamento e como um Sistema de estacionamento pode solucionar a maioria deles, sendo os principais benefícios a redução do tempo gasto procurando vagas, a diminuição do trânsito nas áreas urbanas, a maximização da receita para operadores de estacionamento e a melhoria da experiência do usuário. Também discutimos as tecnologias comuns utilizadas em sistemas de estacionamento, como a automação, a integração de dados em tempo real e a segurança.

3. SmartPark: Sistema de Gestão de Estacionamento

3.1. Descrição geral do sistema

O SmartPark é um sistema de gestão de estacionamento projetado para atender às necessidades mais comuns de um estacionamento mas também trazendo um aspecto inovador de automação e passe fácil para motoristas previamente cadastrados. A ideia central do sistema é controlar a entrada e saída por meio de câmeras que identificarão a placa do veículo para registrar a duração da

permanência no local. Possuir terminais de pagamento e ter a possibilidade de se vincular a sistemas externos, caso haja parceria com empresas próximas, também é um diferencial.

Outra funcionalidade que veio para suprir um dos problemas descritos na seção acima é a melhor visualização das vagas disponíveis. O estacionamento pode contar com sensores de presença e painéis de LED para indicar a quantidade de vagas livres por setor. Já o aplicativo do passe fácil para motoristas, também conterá um mapa do estacionamento indicando mais facilmente onde estacionar, evitando desperdício de tempo e combustível do motorista.

3.2. Interessados

1. **Motoristas:** para terem acesso à uma forma de pagamento mais prática e terem uma experiência de usuário satisfatória, retendo o cliente e gerando receita para o dono do estacionamento;
2. **Operadores das cancelas:** para terem uma interface performática e de fácil manuseio, para utilizar quando necessário;
3. **Proprietário(s) do sistema:** para ter um sistema seguro, disponível e informativo, para que seja possível gerir os funcionários e analisar os dados financeiros e administrativos do sistema;
4. **Proprietário(s) do local:** para ter um sistema com um bom custo-benefício e que se adeque as necessidades do local;
5. **Equipe de vigilância:** para ter um sistema de segurança eficaz e confiável;
6. **Empreendimentos próximos ao local:** para fazer parcerias e aumentar o faturamento;

3.3. Componentes

1. **Cancelas:** para controlar o bloqueio e liberação do fluxo de veículos no estacionamento;
2. **Câmeras:** para liberar o acesso aos veículos fazendo a leitura da placa e para efetuar o monitoramento de segurança do local;

3. **Terminais de pagamento:** para evitar gargalos na saída do estacionamento (caso pagasse só de frente a cancela, como é nos pedágios, por exemplo);
4. **Sensores de presença:** para promover uma melhor experiência do usuário, atuando juntamente com o display LED para informar quantas vagas disponíveis possui aquele setor;
5. **Display LED:** para melhorar a experiência de usuário e diminuir o tempo de procura por vagas;

3.4. Requisitos

O objetivo deste documento é fornecer uma definição e detalhamento arquitetural do projeto SmartPark e suas propriedades, definindo a configuração do sistema, fluxos de dados e interfaces com outros sistemas ou processos.

O sistema SmartPark possui serviços que atendem as necessidades de melhoria dos processos presentes em um estacionamento nos setores financeiro, administrativo e público geral.

Para especificar os requisitos do sistema utilizamos SysADL, uma linguagem de descrição arquitetural baseada em SysML [1]. Na figura 7 é mostrado os requisitos principais do sistema descritos nela.

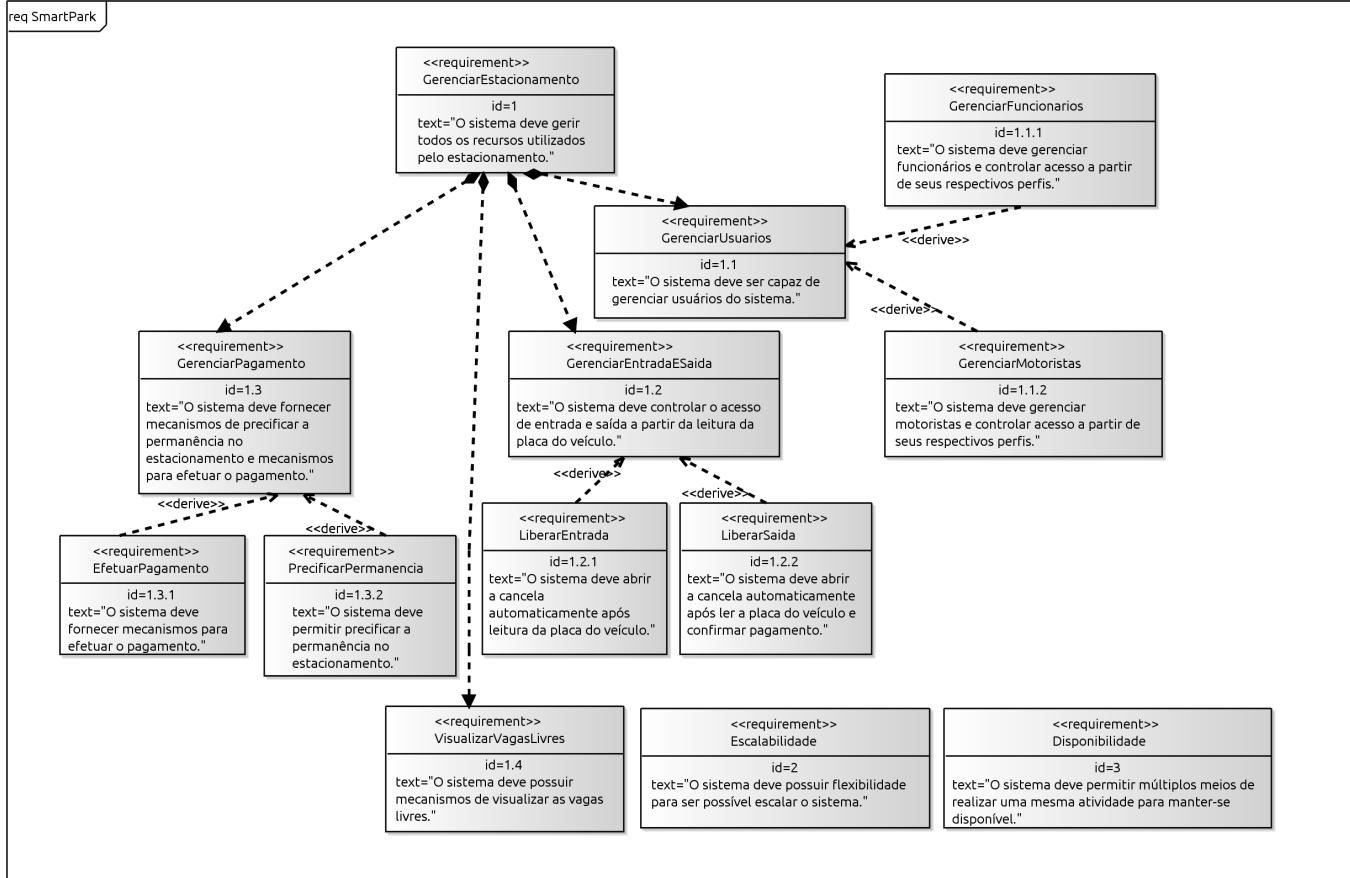


Figura 1: Diagrama de requisitos do SmartPark.

Note que é possível identificar três requisitos principais: **GerenciarEstacionamento**, **Escalabilidade** e **Disponibilidade**.

3.4.1. GerenciarEstacionamento

O requisito **GerenciarEstacionamento** é composto por quatro requisitos:

1. GerenciarPagamento:

Este requisito atende o problema de **Falta de Pagamento e Cobrança Eficiente** ao exigir mecanismos automáticos de pagamento e precificação.

Se os requisitos **EfetuarPagamento** e **PrecificarPermanencia** forem satisfeitos, este também o é.

O pagamento deve estar disponível para ser realizado tanto de forma manual em caixas eletrônicos tanto de forma digital pelo aplicativo utilizando pix, cartão de crédito ou débito.

2. **VisualizarVagasLivres:**

Este requisito atende os problemas de **Escassez de Vagas, Uso Ineficientemente de Espaço e Estacionamento Desorganizado** ao tornar opcional um visualizador de vagas que guia o fluxo de carros, tornando a experiência dos usuários mais fácil. Ele seria opcional pois implicaria em custos consideráveis em sensores de presença para cada vaga do estacionamento, junto a painéis LEDs que mostrariam o número de vagas naquela região.

Além disso o aplicativo deve ter a possibilidade de mostrar um mapa por região (ou andar) do estacionamento, mostrando onde vagas disponíveis estão localizadas.

3. **GerenciarEntradaESaida:**

Este requisito atende os problemas de **Filas e Congestionamento na Entrada/Saída e Falta de Segurança** ao exigir um sistema rápido de identificação de veículos e do seu status de pagamento por meio de suas placas. Caso o sistema falhe, o registro de placas deverá ser feito pelo atendente próximo a cancela. Se os requisitos **LiberarEntrada** e **LiberarSaida** forem satisfeitos, este também o é.

4. **GerenciarUsuarios:**

Este requisito atende os problemas de **Falta de Segurança Falta de Pagamento e Cobrança Eficiente** ao oferecer serviços de cobrança automáticos a usuários cadastrados. Se os requisitos **GerenciarFuncionarios** e **GerenciarMotoristas** forem satisfeitos, este também o é.

Também deverá possuir diferentes controles de permissão para usuário baseado no perfil do usuário.

3.4.2. Escalabilidade

A escalabilidade atende o problema **Má Gestão de Demandas Flutuantes** ao permitir múltiplas cancelas, aumentar a quantidade de vagas e permitir administrar múltiplos estacionamentos. A escalabilidade do ponto de vista do aplicativo e servidores associados não é considerada crítica tendo em vista que em mesmo em horários de pico o fluxo de usuários que utilizam o sistema não é muito significativo computacionalmente, levando em conta que cada estacionamento poderá ter seu próprio servidor interno.

3.4.3. Disponibilidade

A disponibilidade do sistema depende, dentre outras coisas, da sua tolerância a falhas. A seguir uma lista do que deve ser feito caso os seguintes equipamentos ou recursos não estejam utilizáveis:

- Camera entrada: operador registra via placa ou CPF;
- Camera saida: operador recebe comprovante ou consulta a placa/CPF;
- Cancela entrada: operador abre manualmente;
- Cancela saida: operador abre manualmente;
- Internet: sistema funciona offline mantendo um bd local que sera sincronizado com o bd da nuvem;
- Computador: ter redundancia de pc em cada cabine;
- Sistema: ter redundancia de instancias.

3.5. Ponto de vista estrutural

O ponto de vista estrutural em SysADL refere-se à visão que se concentra na estrutura da arquitetura de software, descrevendo os elementos envolvidos e suas interconexões. Isso inclui a especificação dos componentes, conectores, portas, valores e tipos de configuração que compõem a arquitetura. A descrição estrutural também abrange a forma como esses elementos interagem para alcançar a funcionalidade do sistema.

O SmartPark é organizado nos seguintes pacotes:

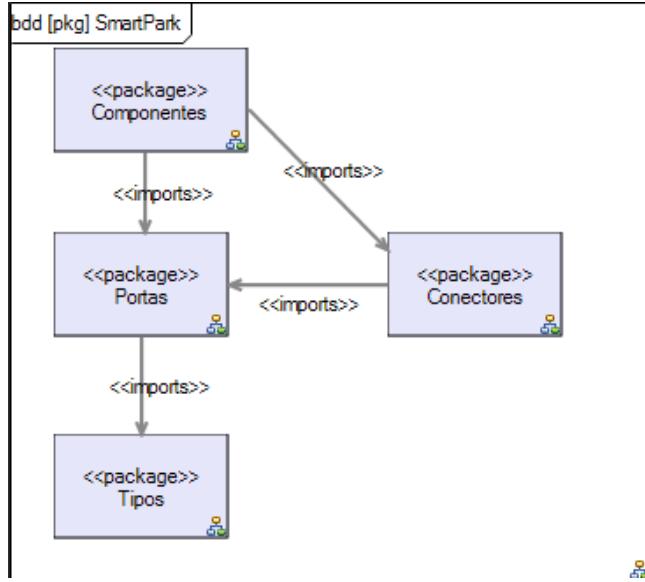


Figura 2: Diagrama de pacotes do SmartPark.

Temos o pacote responsável por definir tipos, o pacote para definição de portas, o pacote para definição de conectores e por fim, o pacote responsável por definir os componentes e como eles interagem entre si. Nesse diagrama também é possível observar as dependências entre eles. Por exemplo, o pacote de Componentes importa o pacote de Conectores e de Portas.

Vamos começar mostrando a definição de tipos necessários para o gerenciamento de estacionamento.

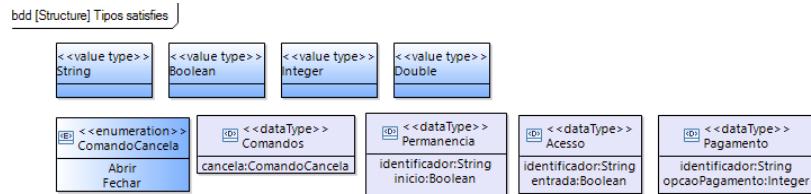


Figura 3: Diagrama de tipos do SmartPark.

Para tipos primitivos (*ValueType*), foi considerado os seguintes tipos: String, Integer, Double e Boolean. Para tipos mais complexos optou-se por utilizar *DataTypes*. Especificamente para lidar com as ações da cancela, foi utilizado um *Enumeration* com os comandos "Abrir" e "Fechar".

Dentro do *DataType Permanencia* os dados são "identificador" e "inicio". Identificador pode ser a placa do carro ou uma chave única identificadora como o CPF do proprietário do veículo, por exemplo. Já o dado inicio marca se esse *DataType* está sendo enviado no início ou final da permanência do veículo no estacionamento.

O próximo *DataType* é o de **Acesso** e os dados são "identificador" e "entrada". Identificador segue a mesma definição supracitada de Permanencia. Já o dado entrada marca se a câmera que enviou a foto da placa é uma câmera que lê entrada ou saída de veículos no estacionamento.

Por fim, o *DataType Pagamento* possui os dados "identificador" e "opcaoPagamento". Identificador segue a mesma definição supracitada de Permanencia. Já o dado opcaoPagamento marca qual das opções disponíveis vai ser utilizada para pagar a taxa de permanência no estacionamento.

Agora vamos ver o pacote de Portas, que importa o pacote de Tipos.

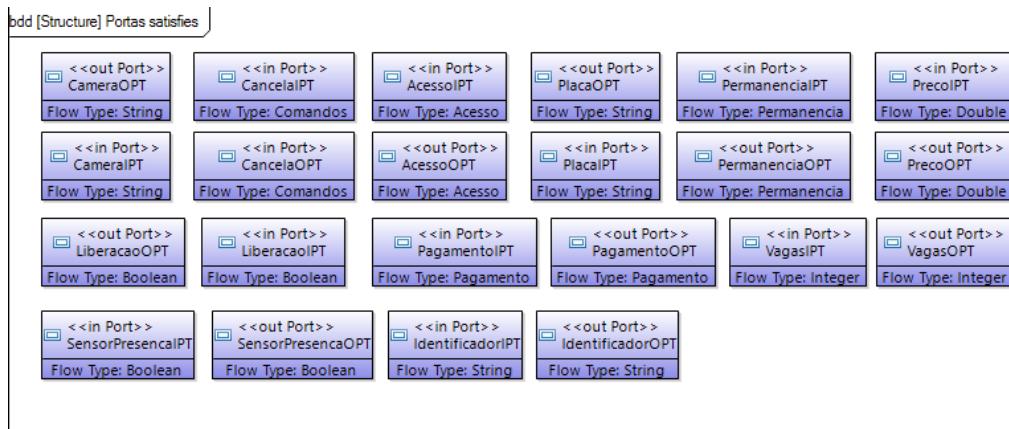


Figura 4: Diagrama de portas do SmartPark.

Note que para todo *DataType* definido no diagrama de Tipos possui uma porta de entrada e de saída correspondente. Além disso, novas portas que utilizam os *ValueTypes* são criadas. As portas de **Camera** que usam String são responsáveis por enviar e receber a foto tirada da placa do veículo. Já as portas de **Placa** usam uma String com o valor da placa já processado, sem ser uma foto. Temos também as portas de **Preco** do tipo Double que demarcam o envio e recebimento da taxa de permanência do estacionamento. As portas de **Liberacao** são responsáveis por enviar e receber um valor booleano que informa se o veículo já realizou o pagamento da estadia no estacionamento. As portas de **Vagas** são responsáveis por enviar e receber o valor da quantidade de vagas, sejam elas as vagas disponíveis ou as vagas totais do estacionamento. Por fim, temos as portas do **SensorPresenca**, responsáveis por enviar e receber valores dos sensores de presença, indicando se uma vaga está ocupada ou não.

Para que a comunicação entre essas portas e esses tipos de dados ocorra, faz-se necessário criar conexões que mapeiam esses fluxos. Para realizar tal feito, utilizamos de um conceito do ponto de vista estrutural chamado de Conector. A seguir é exibido o diagrama de conectores do SmartPark.

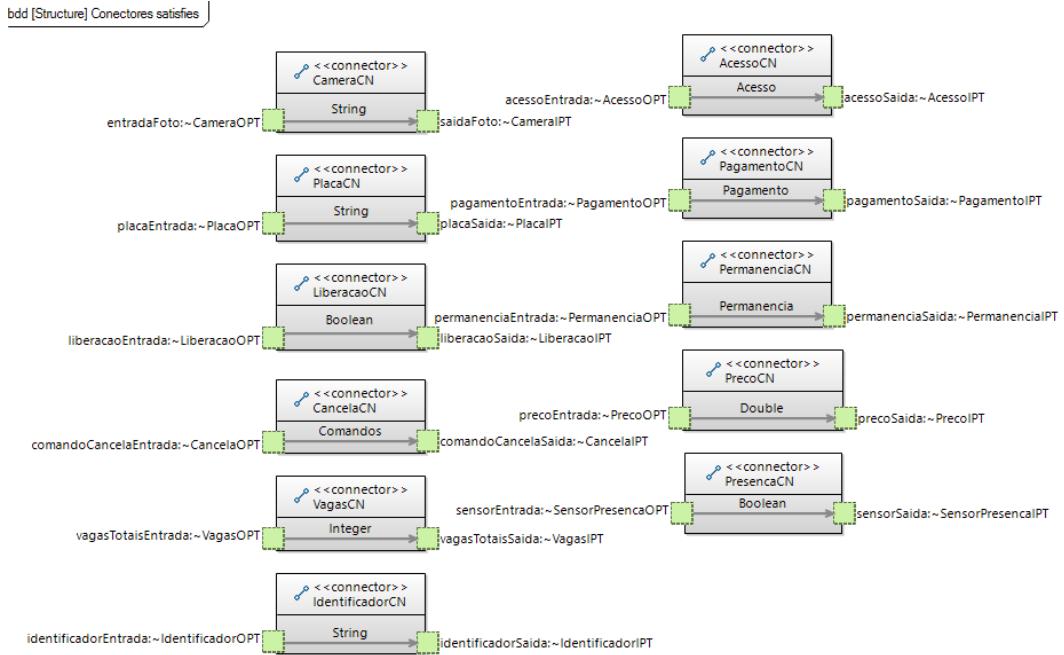


Figura 5: Diagrama de conectores do SmartPark.

Como pode-se observar, não há nenhum conector que mapeia diferentes tipos de dados. Basicamente, todos pegam um dado da porta de saída (convencionada no final como OPT) e levam a uma porta de entrada (convencionada no final como IPT). Todos os conectores são convencionados com o sufixo CN ao final de seus nomes.

Para a implementação do SmartPark não foi necessária a definição de nenhuma porta composta, conector composto ou conector mapeando diferentes dados, muito embora tudo isso seja possível em SysADL.

Já definidas as portas e os conectores, agora tem a definição de componentes, que pode ser observada logo abaixo:

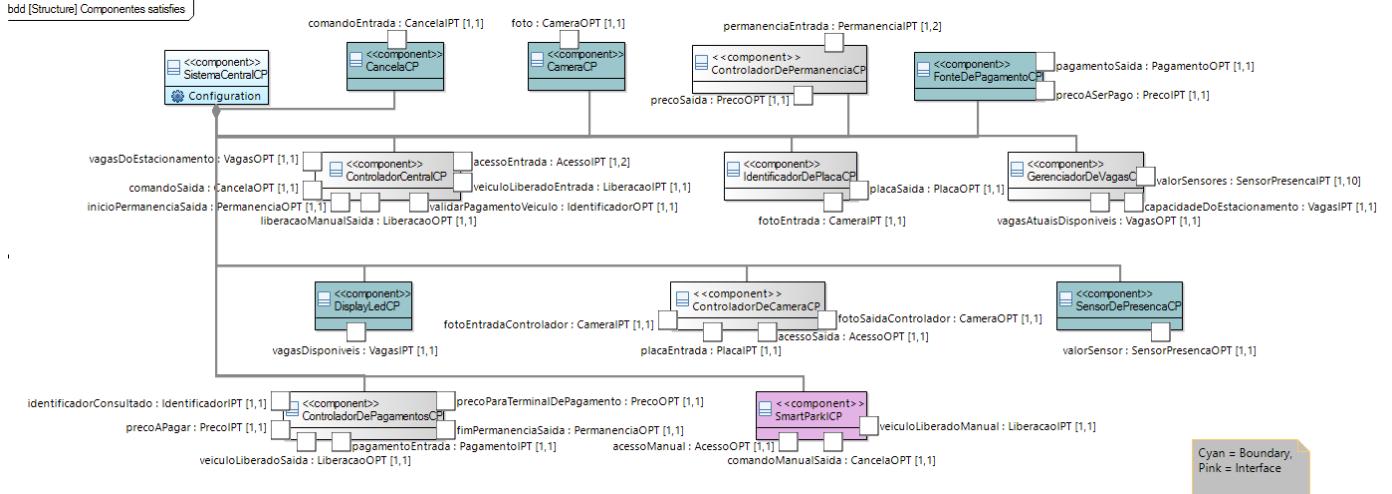


Figura 6: Diagrama de componentes do SmartPark.

Note que existe um componente central (**SistemaCentralCP**) cuja responsabilidade é combinar os componentes a partir de uma configuração que veremos em breve. Para a definição de componentes, se convencionou colocar como sufixo CP ao final dos nomes. Para esse projeto, foi decidido utilizar a cor ciano para identificar componentes de fronteira e a cor rosa juntamente com o sufixo ICP (Interface Component) para componentes de interface.

Por fim, vamos ver como tudo isso está combinado em uma configuração.

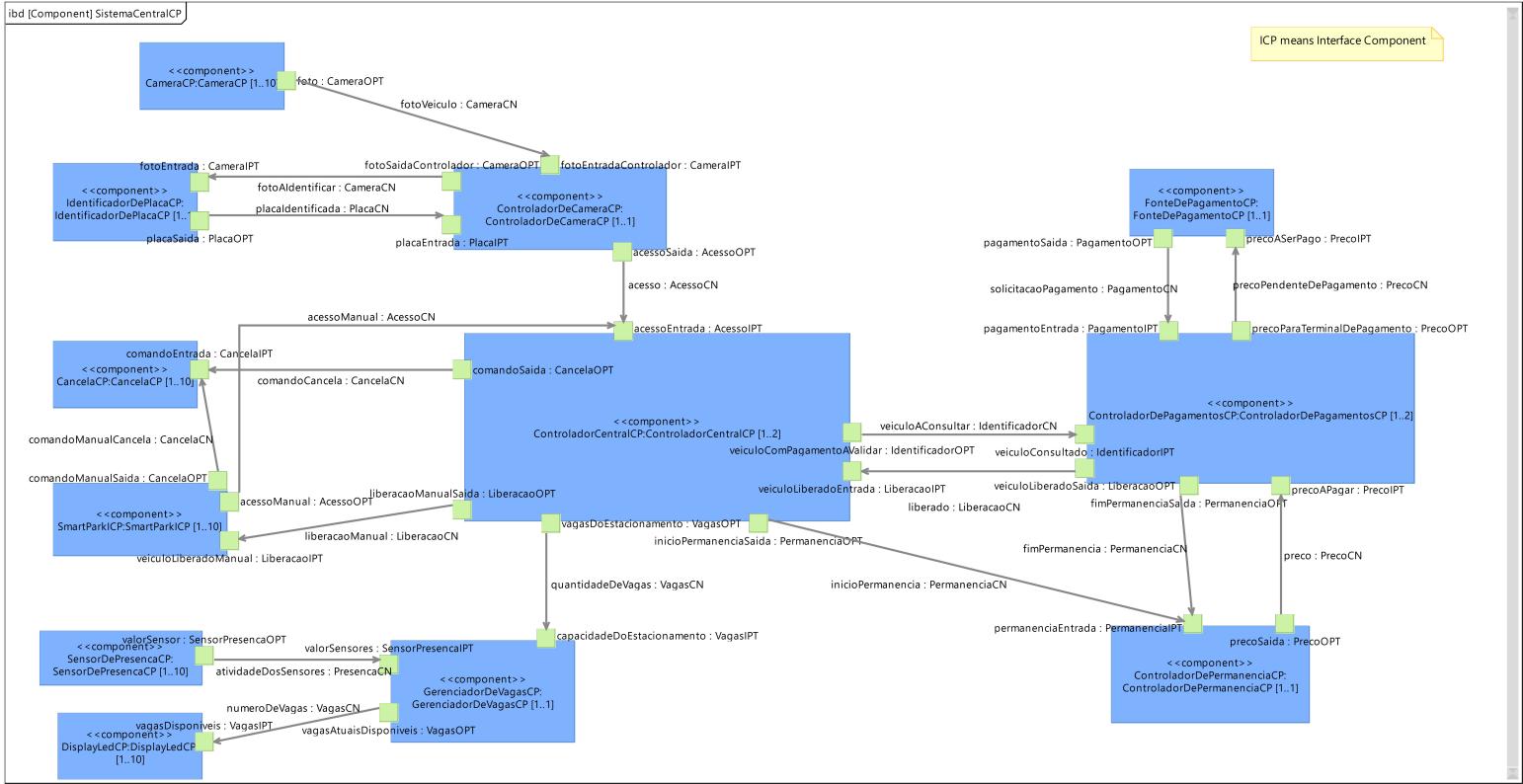


Figura 7: Diagrama de configuração do SistemaCentralCP.

Para entender essa configuração, vamos entender os fluxos de comunicação por meio de funcionalidades.

3.5.1. Entrada no estacionamento

O motorista entra no estacionamento e para em frente a cancela, que vai ter uma câmera associada para identificar a placa do veículo. Assim, **CameraCP** envia uma foto para o **ControladorDeCameraCP**, que identifica se a câmera é de entrada e manda a foto para **IdentificadorDePlacaCP** processá-la e retornar o valor da placa em si. Depois que recebe o valor da placa, o **ControladorDeCameraCP** envia um dado do tipo *Acesso* para o **ControladorCentralCP**. O **ControladorCentralCP** identifica que esse é um *Acesso*

de entrada, envia um dado do tipo *Permanencia* para o **ControladorDePermanenciaCP** e emite um comando de "Abrir" para a **CancelaCP** correspondente.

3.5.2. Vagas no estacionamento

O **ControladorCentralCP** é responsável por saber quantas vagas totais possui o estacionamento e envia para o **GerenciadorDeVagasCP**. Cada vaga vai ter um **SensorDePresencaCP** que envia um valor booleano se a vaga está ocupada ou não. Em cada setor, em cada andar ou até mesmo em cada vaga pode possuir um **DisplayLedCP** responsável por informar quantas vagas estão disponíveis (a depender do escopo previamente mencionado). Quando uma vaga é ocupada, o **SensorDePresencaCP** emite uma mensagem para o **GerenciadorDeVagasCP**, que por sua vez decrementa a quantidade de vagas disponíveis. Uma vez que uma vaga é liberada, ele incrementa a quantidade de vagas disponíveis e é responsável por esse controle.

3.5.3. Saída do estacionamento

O motorista usa uma **FonteDePagamentoCP**, no qual ele informa sua placa ou sua chave identificadora como CPF para consultar a taxa a pagar. O **ControladorDePagamentosCP** envia uma mensagem do tipo *Permanencia* para o **ControladorDePermanenciaCP** indicando que aquele veículo encerrou sua permanência e ele devolve o preço a pagar. Após isso, o motorista sai com o veículo e para em frente a cancela, que vai ter uma câmera associada para identificar a placa do veículo. Assim, **CameraCP** envia uma foto para o **ControladorDeCameraCP**, que identifica que a câmera é de saída e manda a foto para **IdentificadorDePlacaCP** processá-la e retornar o valor da placa em si. Depois que recebe o valor da placa, o **ControladorDeCameraCP** envia um dado do tipo *Acesso* para o **ControladorCentralCP**. O **ControladorCentralCP** identifica que esse é um *Acesso* de saída, envia o identificador para o **ControladorDePagamentosCP** para verificar se está pago e recebe de volta um booleano que diz se está pago ou não. Caso esteja pago, emite um comando

de "Abrir" para a **CancelaCP** correspondente.

3.5.4. Atributos de qualidade

- Disponibilidade: o componente **SmartParkICP** é responsável por fazer gerenciamento manual do sistema em caso de falhas. Tanto **ControladorCentralCP** quanto **ControladorDePagamentoCP** possuem redundância para obter alta disponibilidade.
- Escalabilidade: aspectos de escabilidade podem ser observados principalmente nos componentes de fronteira como **CameraCP**, **DisplayLedCP**, **SensorDePresencaCP** e **FonteDePagamentoCP**, cujo todas as portas que se conectam a esses componentes são multiplexadas, permitindo escalabilidade sem modificar a arquitetura.

4. Satisfação dos requisitos

Ao fim da elaboração dos componentes e suas configurações, tratamos de estabelecer as relações de satisfação entre eles e os requisitos previamente propostos. A figura 8 retrata essas relações. Repare que decidimos fazer alterações pontuais em relação ao diagrama mostrado na figura 1 como por exemplo a adição dos requisitos **GerenciarVagasLivres** e **PerceberOcupacaoVagas**. O requisito **VisualizarVagasLivres** do diagrama anterior agora deriva o novo requisito **GerenciarVagasLivres**. Essa mudança foi realizada após novas reflexões originadas do processo de desenvolvimento de componentes, fazendo com que a nova configuração de requisitos se torne mais consistente com eles.

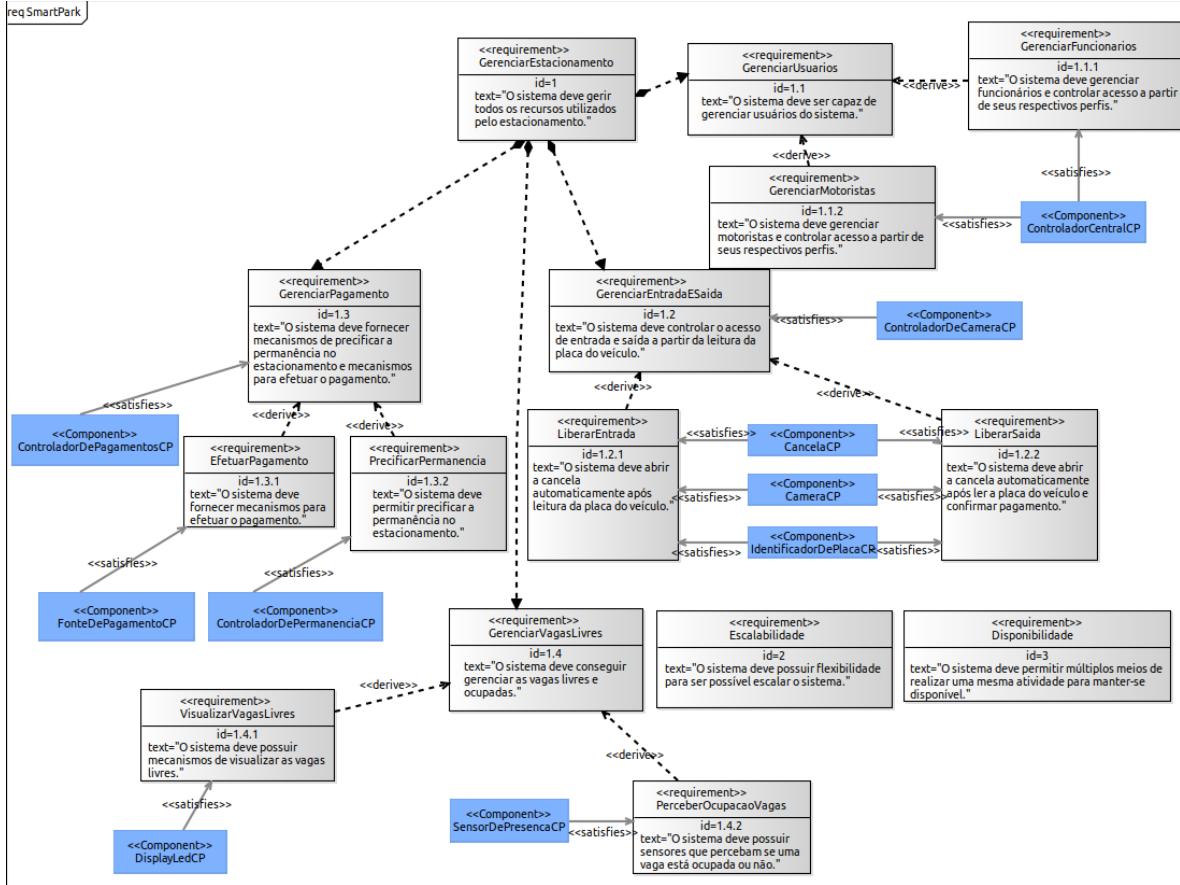


Figura 8: Diagrama de requisitos e os componentes que os satisfazem.

5. Conclusão

Através das especificações apresentadas, demonstramos que o SmartPark, um sistema de estacionamento concebido para proporcionar conforto e facilidade de uso aos seus usuários, representa uma solução viável para os diversos desafios enfrentados pelos estacionamentos em todo o mundo atualmente. Ao adotarmos a linguagem SysADL, conseguimos descrever de maneira sistemática todos os requisitos do sistema e suas inter-relações. Este processo incluiu a aplicação do ponto de vista estrutural, proporcionando uma visão clara da arquitetura do

sistema.

A utilização do ponto de vista estrutural em SysADL permitiu-nos detalhar a estrutura da arquitetura, especificando os elementos fundamentais, como componentes, conectores, portas, valores e tipos de configuração. Além disso, destacamos as interconexões desses elementos, revelando como são utilizados e combinados para alcançar as funcionalidades necessárias e as propriedades de qualidade desejadas para o sistema SmartPark.

Essa abordagem estrutural não apenas fornece uma base sólida para a compreensão do sistema, mas também servirá como suporte valioso na especificação e desenvolvimento subsequente da arquitetura do SmartPark em trabalhos futuros, como a especificação do ponto de vista comportamental.

Referências

- [1] T. B. Flavio Oquendo, Jair Leite, Soft Architecture in Action: Designing and Executing Architectural Models with SysADL Grounded on the OMG SysML Standard, Springer Cham, 2016. [doi:
https://doi.org/10.1007/
978-3-319-44339-3](https://doi.org/10.1007/978-3-319-44339-3).