Universidade do Minho
Escola de Engenharia

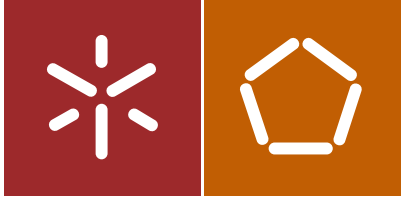
Marcelo Filipe Nunes Henriques

Simulação e melhoria de um sistema de gestão de elevadores

janeiro de 2016

Marcelo Filipe Nunes Henriques Simulação e melhoria de um sistema de gestão de elevadores

UMinho | 2016



Universidade do Minho
Escola de Engenharia

Marcelo Filipe Nunes Henriques

Simulação e melhoria de um sistema de gestão de elevadores

Tese de Mestrado
Ciclo de Estudos Integrados Conducentes ao
Grau de Mestre em Engenharia Industrial

Trabalho efetuado sob a orientação do
Professor Doutor Luís Miguel da Silva Dias

janeiro de 2016

DECLARAÇÃO

Nome: Marcelo Filipe Nunes Henriques

Endereço eletrónico: marcelo@nhenriques.com Telefone: 00351 965232584

Número do Bilhete de Identidade: 13370117

Título da dissertação: Simulação e melhoria de um sistema de gestão de elevadores

Orientador: Professor Doutor Luís Miguel da Silva Dias

Ano de conclusão: 2016

Designação do Mestrado: Mestrado em Engenharia Industrial

Nos exemplares das teses de doutoramento ou de mestrado ou de outros trabalhos entregues para prestação de provas públicas nas universidades ou outros estabelecimentos de ensino, e dos quais é obrigatoriamente enviado um exemplar para depósito legal na Biblioteca Nacional e, pelo menos outro para a biblioteca da universidade respetiva, deve constar uma das seguintes declarações:

1. É AUTORIZADA A REPRODUÇÃO INTEGRAL DESTA DISSERTAÇÃO APENAS PARA EFEITOS DE INVESTIGAÇÃO, MEDIANTE DECLARAÇÃO ESCRITA DO INTERESSADO, QUE A TAL SE COMPROMETE;
2. É AUTORIZADA A REPRODUÇÃO PARCIAL DESTA DISSERTAÇÃO (indicar, caso tal seja necessário, nº máximo de páginas, ilustrações, gráficos, etc.), APENAS PARA EFEITOS DE INVESTIGAÇÃO, MEDIANTE DECLARAÇÃO ESCRITA DO INTERESSADO, QUE A TAL SE COMPROMETE;
3. DE ACORDO COM A LEGISLAÇÃO EM VIGOR, NÃO É PERMITIDA A REPRODUÇÃO DE QUALQUER PARTE DESTA TESE/TRABALHO

Universidade do Minho, ____ /____ /_____

Assinatura:

AGRADECIMENTOS

Até este momento, houve várias pessoas que foram decisivas na minha vida, não só pela ajuda até o início desta Dissertação, mas também pelo apoio prestada durante a realização deste trabalho. Sem essa ajuda e apoio nunca teria alcançado o ponto onde me encontro hoje. Os seguintes agradecimentos tem um significado muito especial para mim, não estando por nenhuma ordem específica.

Agradeço aos meus pais por todo o apoio que me deram desde que nasci. Sinto-me um sortudo pelos pais que tenho, quer pelo carinho que sempre me deram, quer pelo sacrifício que todos os dias fazem para me dar todas as ferramentas que necessito para ter sucesso e ser feliz. Do fundo do meu coração: obrigado!

Agradeço à minha irmã, pois foi uma peça fundamental no meu crescimento. A sua vinda foi uma aprendizagem, essencial no meu crescimento e na minha maturação, acrescentando lições muito importantes na minha formação pessoal. Hoje partilhamos lições mais profissionais, igualmente importantes. Agradeço-te pelo apoio constante.

Agradeço a toda a minha família, quer pelo apoio incondicional, quer pelos conselhos. Sempre foram uma fonte de alento e força para continuar. O simples convívio é decisivo no meu bem-estar.

Agradeço à minha namorada, pois sempre me apoiou, mesmo quando as coisas não estavam fáceis, nem pareciam ter fim à vista. O carinho e a força foram decisivos na minha motivação para navegar até bom porto.

Agradeço ao professor Luís Dias pelo apontar deste caminho e pelo apoio durante a caminhada. A sua assertividade, ponderação e natureza calma foram muito importantes neste processo.

Agradeço ao colega e amigo António Vieira pela ajuda técnica e pelo apoio. Mesmo fora de horas, a disponibilidade sempre foi imensa e a atenção surpreendente.

Agradeço ao professor José Telhada e à minha prima Tânia Rodrigues, pois os seus conselhos foram decisivos numa altura em que as minhas forças pareciam estar perto do fim.

Agradeço a todos os meus amigos, porque os momentos de lazer são de igual importância aos momentos de trabalho, pois permitem-nos atingir o equilíbrio. Os conselhos foram muito importantes, mas a força foi decisiva para esta jornada.

Resumo

O presente trabalho tem como objetivo simular um elevador, quer do ponto de vista do transporte de passageiros, mas também do ponto de vista da tomada de decisão de quem transportar. Este último ponto é a génese do sistema de gestão de elevadores, responsável pelo controlo do elevador, mediante vários fatores, de onde se destacam as chamadas, onde cada passageiro solicita o elevador; e os destinos, o andar pretendido por quem se encontra no interior do elevador.

Será utilizado o *software* Simio® para construir o modelo, em duas vertentes: o elevador como veículo, um objeto nativo do *software* com processos e variáveis bem definidas; e o elevador como entidade, um objeto criado partindo do zero com processos e variáveis adequados às necessidades da aplicação.

O modelo desenvolvido cria uma base de trabalho, que servirá para analisar as seguintes medidas de desempenho: tempo total (ou de atravessamento do sistema), ocupação (ou carga) do elevador, movimentações do elevador, clientes no sistema, clientes em espera e tempo de espera. O modelo possui as seguintes propriedades de parametrização: tempo de abertura de portas, a capacidade máximo do elevador, a criação de clientes no andar do rés-do-chão e a criação de clientes nos andares superiores. A análise irá focar-se no impacte do tempo de abertura de portas nas várias medidas de desempenho, destacando-se o tempo total (ou de atravessamento do sistema).

O modelo desenvolvido poderá ser utilizado como objeto em futuros modelos.

Palavras-Chave

Elevador, Objeto, Modelo, Sistemas de Gestão, Simio, Simulação

Abstract

This work has the objective of simulating an elevator, on both the transporter side of things, and in decision-making. This last point is the genesis of the elevator management system, that is responsible for managing the elevator, depending on a variety of factors, such as calls, where each passenger asks for the elevator to pick him up; and destinies, the desired floor for those inside of the elevator.

The Simio® software will be used to simulate this model, in two approaches: the elevator as a vehicle, a built-in object from the software's library, with processes and variables defined by the makers of the software; and the elevator as an entity, a built from scratch object, with processes and variables adjusted to this application.

The model developed in this Thesis builds a framework, which allows the analyses of the following performance measures: total time (waiting time plus travel time), occupation (or load) of the elevator, number of elevator movements, customers in the system, customers waiting and wait time. The model has the following properties as variable parameters: door opening time, maximum capacity of the elevator, number of customers created on the ground floor and number of customers created on the upper floors. The analysis will focus on the impact that the door opening time has on various performance measures, especially the average of total time (waiting time plus travel time).

The model created can be used as an object in future models.

Keywords

Elevator, Lift, Object, Model, Management systems, Simio, Simulation

ÍNDICE

ÍNDICE DE FIGURAS

Índice de Gráficos

ÍNDICE DE TABELAS

Índice de Equações

Lista de Abreviaturas, Siglas e Acrónimos

CHUC: Centro Hospitalar e Universitário de Coimbra, EPE;

ETA: "*Estimated Time of Arrival*", ou Tempo Estimado de Chegada;

ETD: "*Estimated Time to Destination*", ou Tempo Estimado até ao Destino;

SDF: "*System Degradation Factor*", ou Fator de Degradação do Sistema.

TC: "*Total Cost*", ou Custo Total

1. O ELEVADOR

O elevador é o cerne do tema debatido na presente Dissertação e, portanto, deve ser analisada a sua génese, as evoluções que sofreu ao longo do tempo, as suas várias aplicações, os sistemas de gestão – quer permitem controlar sistemas com dois ou mais elevadores – e o estado da arte.

O elevador é, hoje, uma peça chave para a vida de milhões de pessoas que todos os dias sobem, ou descem, os arranha-céus espalhados pelo mundo. Metrópoles, como Nova Iorque, têm no elevador uma peça chave para o seu desenvolvimento.

Mesmo em pequenos edifícios, os elevadores são essenciais para garantir a acessibilidade a pessoas com mobilidade reduzida.

1.1 Génese e evolução

Para melhor compreender o sistema elevador é aconselhável acompanhar a génese do mesmo. Pois, tal como as grandes invenções que revolucionaram o mundo – o computador, o automóvel ou o avião – não existe apenas uma personalidade, um momento ou uma invenção por si só à qual se possa atribuir todo o mérito sobre o sistema que utilizamos hoje e é aplicado por todo o mundo. Este subcapítulo pretende expor os passos que foram decisivos para a chegada ao sistema elevador que é hoje mundialmente aceite e aplicado em vários locais do mundo.

A história demonstra que a necessidade de transportar bens e pessoas verticalmente é milenar.

O princípio básico de um elevador assenta em roldanas que içam verticalmente uma plataforma, da mesma forma que uma grua ou um guincho içam materiais de construção para um prédio em construção, independentemente da força motriz. A origem dos guinchos, ou simplesmente roldanas, é muito antiga, sendo possível observar tais sistemas em gravuras do Antigo Egito, nomeadamente na construção de pirâmides(Molyneaux, 2006; “Timeline of the Elevator,” n.d.).

Anteriormente à revolução industrial, a única forma de fazer mover bens ou pessoas verticalmente era a força humana, animal ou gravítica(“Timeline of the Elevator,” n.d.). Um exemplo desta última força é o elevador do Bom Jesus, em Braga, Portugal; onde a água é utilizada para compensar a diferença da força Peso entre as duas cabinas. A cabina que está no topo é carregada com água, enquanto, no ponto mais baixo do percurso, é descarregada a água da outra cabina. De uma forma simples: a adição do peso da água carregada com o peso da própria cabina que se encontra no topo terá que ser superior à soma da força de atrito inicial da cabina que se encontra no ponto mais baixo com o seu

próprio peso. Quando esta condição é cumprida e os sistemas de segurança são desacoplados, o ascensor inicia o movimento, permitindo à cabina que se encontrava no topo descer e à cabina que se encontrava na parte inferior subir.

A força humana era utilizada para mover elevadores que, vulgarmente, são chamados de monta-cargas. Isto é, plataformas que faziam subir bens de um piso inferior para um piso superior. Este tipo de esquema era implementado por roldanas e cabos ou cordas.

A força animal era muito utilizada, não só porque era vulgar – visto também ser aplicada na agricultura – mas também porque oferecia uma maior força.

Há diversas fontes que reclamam a criação do sistema elevador, e a resposta dependerá da própria definição de elevador.

A mais antiga, e consequentemente mais básica definição, foi mencionada nos manuscritos de Marcus Vitruvius Pollio, um escritor, arquiteto e engenheiro civil Romano, e dá ao Grego Arquimedes o mérito da criação de um sistema elevatório, no ano 236 a. C(“History of the Elevator,” n.d., “Who invented the elevator?,” n.d.-a), operado por homens que puxavam cordas, acopladas a um cilindro, que, por sua vez, fazia rodar um eixo de tração.

É também referido que(“Who Invented the Elevator?,” n.d.-a), na Roma antiga, por baixo do Coliseu, existia um complexo sistema deste tipo para içar os gladiadores e os seus animais para a arena de combate, movimentado por centenas de homens e animais.

Outra referência histórica data de 1743(“Who invented the elevator?,” n.d.-a), em que o Rei Luís XV, de França, possuía um sistema – apelidado de “a cadeira voadora” – para o permitir visitar as suas amantes nos seus respetivos aposentos, do rés-do-chão para o segundo andar do Palácio de Versailles, por meio de cordas, puxadas por criados. Este mesmo Rei, no Castelo de Choisy – hoje destruído – possuía uma “mesa voadora” para servir o jantar aos seus convidados sem a intromissão dos criados. Ao tocar de um sino, a mesa seria içada da cozinha abaixo até à sala de jantar, com uma ementa requintada e os apetrechos necessários.

Até meados do século XIX, o elevador foi evoluindo, e soluções propulsionadas por vapor ou água estavam disponíveis, mas os cabos que tracionavam as cabinas não ofereciam a segurança necessária para transportar pessoas, uma vez que a falha das cordas ou cabos que sustentavam as plataformas eram uma realidade.

Com a revolução industrial e o desenvolvimento de novos processos de fabrico e tratamento do metal, nomeadamente as suas ligas, como o aço, houve um grande desenvolvimento no setor da construção,

Com a aplicação de vigas, novas possibilidades foram abertas, permitindo a construção de prédios cada vez mais altos.

Estas novas construções seriam pouco práticas caso, a par desta evolução, não houvesse a aplicação de um sistema de transporte vertical – ou, também apelidado de, transversal – que permitisse transportar pessoas do piso do rés-do-chão para um piso superior, sem esforço físico por parte destes, em tempo aceitável e em segurança.

As aplicações de força humana aconteciam em situações com altura reduzida, ou de peso relativamente baixo, como é a subida de bens de apenas um andar. Outras aplicações exigiam outras medidas; mais caras, muito complexas e espaçosas.

Utilizar a força gravítica, através da água, era impraticável, uma vez que o transporte da própria água para o topo do edifício representava, só por si, um desafio.

A utilização da força animal era benéfica – e realista – em planos com declive, mas tornavam-se pouco práticos em situações verticais, onde o esforço é superior, uma vez que a força peso tem apenas componente vertical. Outros desafios seriam: garantir uma elevada taxa de disponibilidade e prontidão, o espaço e recursos necessários para manter os respetivos animais, que seriam incompatíveis com o estilo de vida cosmopolita contemporâneo e com o objetivo inicial da construção de um prédio: crescer na vertical, uma vez que os próprios animais teriam que se movimentar para fornecer a força motriz ao elevador.

Era, portanto, necessário dar resposta aos seguintes desafios: construir um sistema mecânico prático suficientemente forte para suportar valores de peso avultados, desenvolver um sistema de segurança que permitissem garantir a segurança dos bens e das pessoas e implementar um sistema de controlo que respondesse às solicitações dos passageiros.

Os primeiros dois desafios eram essenciais para qualquer aplicação de um sistema elevatório de transporte de pessoas e bens, sendo o último um processo que foi evoluindo ao longo dos tempos.

A princípio as solicitações eram realizadas por um só mordomo, que viajava continuamente no elevador e era responsável pela seleção do andar pretendido pelos vários passageiros, ou seja, um sistema simples de solicitação única.

Hoje em dia, o sistema mais básico de elevadores, embora assente em solicitações no interior do elevador, responde a múltiplas solicitações, desde os vários destinos pretendidos pela totalidade dos passageiros no interior do elevador, até às várias chamadas dos passageiros que o pretendem utilizar.

Foi no século XIX que começaram a surgir ou se vulgarizaram várias tecnologias que permitiram o avanço significativo dos elevadores. Nomeadamente os motores a vapor, a aplicação da hidráulica e o início do eletromagnetismo("Timeline of the Elevator," n.d.).

Em 1823("History of Elevators," n.d.), os arquitetos Burton e Hormer construíram um elevador primitivo, alimentado por uma máquina a vapor, apelidado de "sala ascensora", que elevava turistas até uma plataforma colocada a trinta e sete metros de altura, permitindo ter um visão diferente da cidade de Londres.

Esta atração turística viu, em 1835, duas melhorias: a adição de um cabo e de um contrapeso. Os autores destas melhorias, Frost e Stutt, chamaram-lhe "*Teagle*"("History of Elevators," n.d., "History of the Elevator," n.d.).

O uso da hidráulica propagava-se, quer com água, quer com óleo. A utilização da hidráulica em elevadores foi testada e aplicada("History of Elevators," n.d.), competindo em paralelo com a máquina a vapor.

Na utilização de sistemas hidráulicos em elevadores, para albergar o cilindro, era necessário escavar aberturas no solo, para que o pistão descesse conforme aumentava a pressão. Mas os limites eram evidentes, uma vez que a profundidade do cilindro era proporcional à altura do elevador.

Então, em aplicações com alturas elevadas a aplicação mais razoável seria a máquina a vapor a içar cabos. Esta aplicação, embora possível, era perigosa, uma vez que os episódios de quebras de cabos eram vulgares, resultando em perdas, de bens e de vidas humanas("Who Invented the Elevator?," n.d.-b).

Nos anos 50 ("Elisha Graves Otis," n.d.), Elisha Graves Otis – depois de ter produzido diversos sistemas mecânicos para a empresa onde trabalhava – deparou-se com um novo desafio: a necessidade de um sistema de transporte vertical seguro para os armazéns, que garantisse a segurança de bens transportados e das pessoas que o operavam.

Em 1852("The Inventors," n.d.-a), Elisha Otis, desenvolveu um sistema de segurança para elevadores que consistia, de forma básica, num travão de emergência, O dispositivo, colocado no topo da plataforma, iria, na eventualidade da corda quebrar, desbloquear uma mola que movimentava para fora um braço que engrenaria na cremalheira colocada ao longo do percurso da própria plataforma, evitando a queda livre. O sistema foi apelidado de "guincho de segurança".

Em 1854("Otis History," n.d.), na exposição de Crystal Palace, em Nova Iorque, Elisha Otis realizou uma demonstração ao vivo, em que o próprio cortou o cabo que mantinha a plataforma onde este estava suspenso. A plataforma desceu alguns centímetros e parou.

Este sistema de segurança revolucionou a indústria dos elevadores e mudou a visão das pessoas em relação ao elevador, pois dava garantias de segurança caso as cordas ou cabos que sustentavam o elevador se quebrassem, parando o elevador.

O primeiro elevador com este sistema, com o intuito de servir passageiros, foi instalado a 23 de março de 1857("Elisha Graves Otis," n.d., "Who invented the elevator?," n.d.-a) num centro comercial de cinco andares em Nova Iorque. Anteriormente, este sistema foi instalado em elevadores destinados a içar bens, em fábricas e armazéns.

Foi Elisha Graves Otis que deu origem à empresa Otis Elevator Company Inc. – inicialmente denominada de Otis Brothers – um dos *players* a nível mundial do negócio do transporte vertical.

A patente para este sistema foi apresentada em 1961(Gray, 2002; "National Inventors Hall of Fame," n.d., "The Inventors," n.d.-a), numa versão aprimorada do sistema originalmente apresentado. Na Figura 1 é possível visualizar o sistema de segurança patenteado (do lado esquerdo) e a réplica laboratorial (do lado direito) do sistema de segurança desenvolvido por Elisha Otis("The Other Elevator Inventor," n.d.).

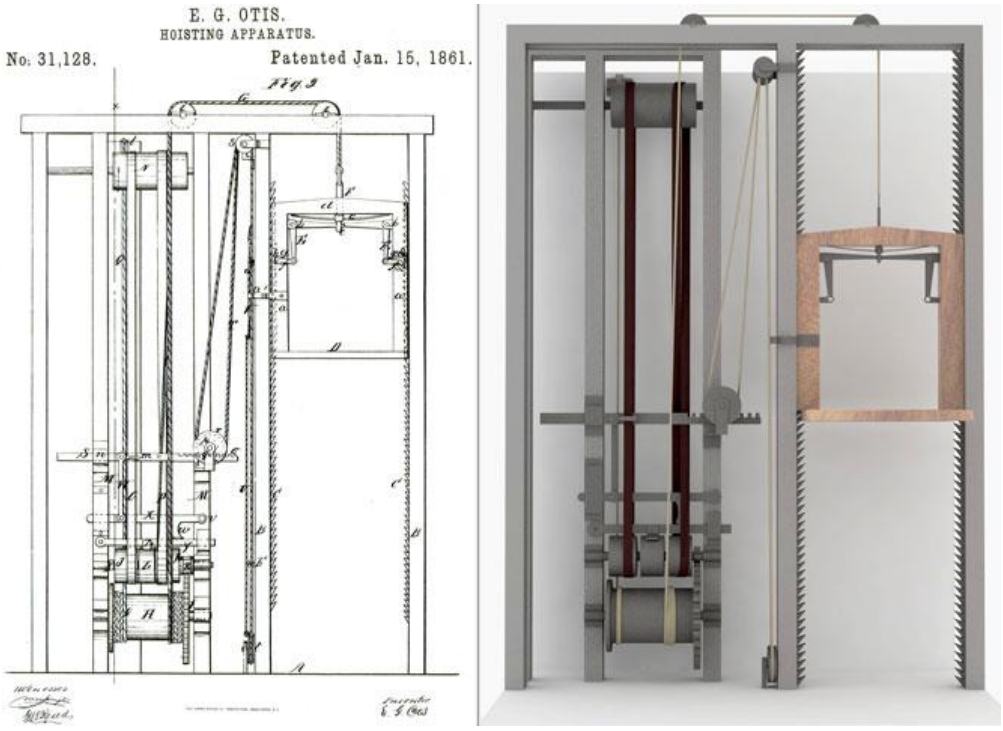


Figura 1 – Desenho patenteado do sistema de segurança desenvolvido por Elisha Otis

Há referências que colocam a pergunta: foi Elisha Otis ou Otis Tufts que inventou o elevador?("The Other Elevator Inventor," n.d., "Who invented the elevator?," n.d.-b). Embora nenhum tenha sido o

direto e inequívoco inventor do elevador, no verdadeiro sentido da palavra, ambos contribuíram para o sistema que é aplicado atualmente nos elevadores.

Otis Tufts – inventor do sistema de bate-estacas a vapor("The Inventors," n.d.-b, "The Other Elevator Inventor," n.d.), que permitiu a colocação de estacas a elevadas profundidades assegurando a estabilidade das estruturas – dedicou-se ao desenvolvimento de um sistema de elevação baseado na sua anterior descoberta.

Tufts focou-se na segurança, conforto e suavidade. Pelas suas próprias palavras: "um elevador para a conveniência das pessoas nos vários andares de hotéis, edifícios públicos e ainda residências privadas."("The Other Elevator Inventor," n.d.).

O seu sistema foi patenteado em 1859("The Other Elevator Inventor," n.d., "Timeline of the Elevator," n.d.). Este consistia num parafuso – colocado no centro da cabina – que se movimentava verticalmente através da rotação de uma guia – colocada ao longo do poço do elevador – e, através da sua movimentação vertical, fazia a cabina subir ou descer, conforme o sentido da rotação. A força motriz vinha de um motor a vapor. Este sistema foi apelidado pelo próprio de "Elevador-parafuso vertical". O desenho patenteado é visível abaixo na Figura 2.

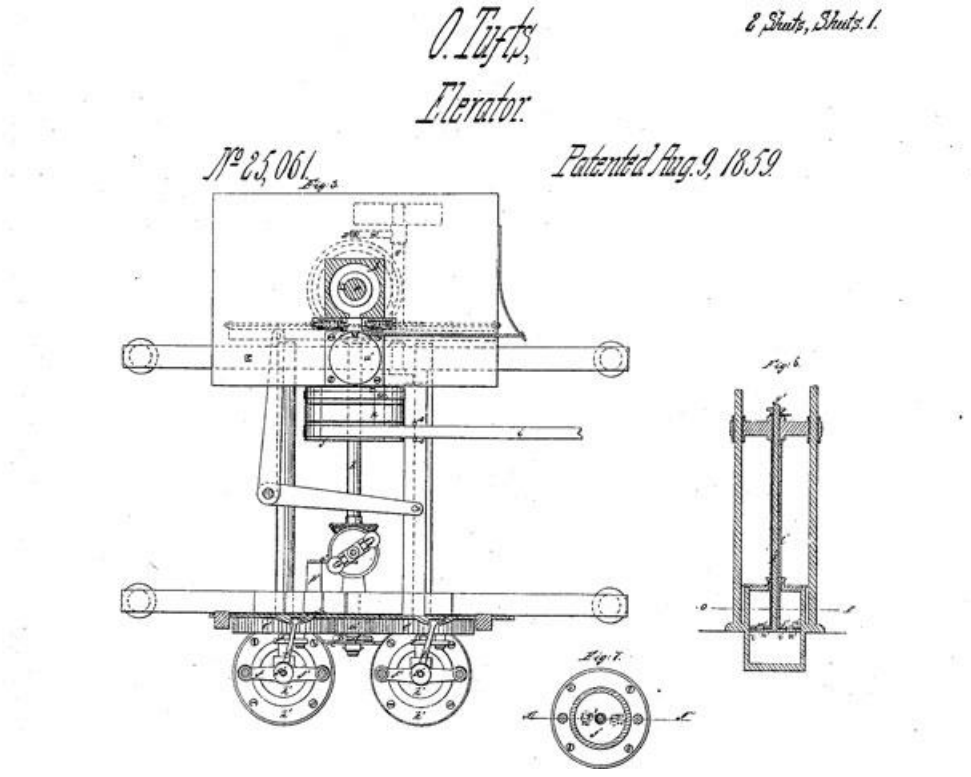


Figura 2 – Desenho patenteado por Otis Tufts("The Other Elevator Inventor," n.d.)

O que distinguiu esta solução de outras até então era o enfoque no conforto. Esta solução apresentava uma cabina completamente fechada, ao invés de uma plataforma. Este *design* impedia os passageiros,

ou as suas roupas, de se entrelaçarem em componentes do elevador, incluía portas automáticas e bancos para os passageiros. Era um elevador focado no transporte de pessoas, e não bens.

Otis Tufts contribui com várias inovações que trouxeram melhorias significativas ao elevador, quer em termos de conforto, quer em termos de segurança.

Foi Tufts que introduziu a utilização de dois, ou mais, cabos para sustentar a cabina("The Inventors," n.d.-b). Esta abordagem permitia não só dividir o esforço entre os cabos, mas também acrescentava redundância ao sistema.

Em 1866("Timeline of the Elevator," n.d.) Tufts desenvolveu um sistema, de três cabos, com guias tracionadas por molas, divididas em quatro secções ao longo do poço do elevador. Esta implementação permitia um melhor ajuste dos cabos, no motor e cabina, reduzindo o balanço da cabina e o entrelaçamento de cabos. Nesse mesmo ano Tufts patenteou uma versão com melhorias nas guias, que resultavam num movimento mais suave e menos ruidoso, apropriado para aplicações em hospitais.

Nos anos seguintes, o sistema "*Teagle*", de Frost e Stutt, juntamente com os sistemas de segurança e conforto introduzidos por Elisha Otis e Otis Tufts, eram os componentes mais utilizados na instalação de elevadores. Nova Iorque foi o epicentro deste novo meio de transporte, abrindo novos horizontes ao mercado imobiliário, não só pela possibilidade da construção de prédios cada vez maiores, mas também pela mudança do desejo dos clientes: os andares de topo passaram a ser os mais desejados, pela sua vista privilegiada sobre a cidade(Bernard, 2014).

Apesar do sistema de propulsão mais comum ser a máquina a vapor, houve diversas instalações de sistemas hidráulicos – com água ou óleo – e ainda experiências com a força gravítica, com máquinas a vapor nos topos a fazerem subir a água para o elevador. Os sistemas que utilizavam a força gravítica tiveram algum relevo pela sua velocidade, apontado como ideal para servir funcionários, especialmente os jovens encarregues de entregar correio.

Em 1880("First electric elevator," n.d.), convidado por uma exposição local, Werner von Siemens – que testava várias aplicações para a sua recente tecnologia: o dínamo – desenvolveu e instalou o primeiro elevador com um motor elétrico em Mannheim, Alemanha. Nesse mesmo ano o sistema foi patenteado por Werner von Siemens e Johann Georg Halske, os fundadores da empresa Siemens & Halske, que viria a dar origem à empresa Siemens.

Apesar desta aplicação, o foco de Siemens era outro, e, apesar da procura por elevadores elétricos, a empresa não seguiu esta área de negócio("First electric elevator," n.d.). Ao invés, seguiu-se a aplicação

de dínamos, movimentados por máquinas a vapor, em centrais elétricas em várias cidades da Alemanha("Timeline of the Elevator," n.d.), essenciais para a proliferação da energia elétrica.

Em 1889("History of Elevators," n.d., "Timeline of the Elevator," n.d.), Elisha Otis implementou, pela primeira vez, um motor de corrente alternada num sistema de elevador. Neste período, foram experimentados e aplicados motores de corrente alternada e contínua, uma vez que a energia elétrica estava em constante evolução e nenhum dos modelos tinha dominado. Um dos marcos desta mudança ocorreu em 1922, quando a empresa Edison – distribuidora de energia elétrica em Nova Iorque – alterou a distribuição de energia elétrica bifásica em corrente contínua para trifásica alternada. Esta mudança obrigou a diversas conversões de sistemas motrizes de elevadores.

A aplicação de motores elétricos trazia várias vantagens: maior suavidade, maior velocidade, permitia içar uma maior quantidade de bens – uma vez que o binário produzido por um motor elétrico é instantâneo e constante – dispensando engrenagens – utilizadas anteriormente para desmultiplicar o binário produzido pelas máquinas a vapor – e não exigiam sistemas de extração de fumos.

Ao mesmo tempo que a energia elétrica era aplicada em elevadores, esta também foi testada em escadas rolantes, guinchos e comboios. Neste mesmo período, começaram a aparecer os primeiros metropolitanos, nomeadamente em Nova Iorque, Londres, Chicago e Glasgow. Toda esta evolução permitiu a proliferação da energia elétrica, dos motores elétricos e do próprio elevador. Em 1907, as empresas Otis e Sprague, de Nova Iorque e New Jersey respetivamente, uniram-se para ganhar um contrato de 170 elevadores para o metropolitano de Londres, com uma capacidade individual de 70 pessoas.

Nesta fase, os elevadores eram controlados por um operador que permanecia no interior da cabina, dando instruções simples como: subir, descer, abrir ou fechar as portas, parar ou tocar o alarme. Exemplo deste sistema era o edifício Singer("Timeline of the Elevator," n.d.), em Nova Iorque, com 41 andares, em que cada operador de elevador possuía um telefone, no interior da cabina, respondendo às ordens de um supervisor do sistema, que controlava as movimentações e autorizava as partidas dos vários elevadores.

Em 1904("Timeline of the Elevator," n.d.), Joseph Richmond/Carey desenvolveu um sistema de controlo através de um botão para executar a chamada do elevador. Mais tarde, em 1907, Elisha Otis viria a implementar um sistema de controlo parecido. Em 1924, Otis implementou o primeiro sistema independente de operadores de elevador. Progressivamente, estes sistemas de controlo foram sendo instalados nos vários sistemas de elevadores(Bernard, 2014; Jarboe & O'Donoghue, 2007).

Haughton desenvolveu em 1926 – e patenteou em 1929 – um sistema de portas de elevador completamente automáticas.

Com a proliferação dos vários sistemas de elevadores, surgia a necessidade de regular a área. Em 1914 surge um regulamento na cidade de Boston e em 1918 surgem as primeiras regras em Nova Iorque, que limitavam a velocidade dos elevadores em 700 pés por minuto, aproximadamente 3,556 metros por segundo.

Em 1930 e 1932 surgem dois edifícios icónicos na cidade de Nova Iorque: o Chrysler Building e o Empire State Building, respetivamente. O segundo destronou o primeiro como sendo o edifício mais alto do mundo. Para permitir uma melhor mobilidade dentro do Empire State Building, as regras do regulamento para elevadores na cidade de Nova Iorque tiveram que ser alteradas, permitindo a velocidade de 1000 pés por minuto – aproximadamente 5,08 metros por segundo – aos 64 elevadores instalados no edifício.

1.2 O sistema atual

Existem vários sistemas de elevadores, quer em termos de origem da força motriz, quer em termos de transmissão da própria força motriz à cabina, tal como visto no ponto anterior.

A origem de força motriz mais comum é o motor elétrico – pelo seu binário, fiabilidade e suavidade – e a forma de transmissão mais comum é por tração de cabos.

Os sistemas hidráulicos são ainda aplicados – embora em menor número que os motores elétricos – por razões monetárias, facilidade de manutenção em algumas aplicações, necessidade de maior binário em detrimento da velocidade, dificuldade na receção de energia elétrica ou outro fator crucial,

Os motores elétricos poderão ser de corrente alternada ou contínua. A corrente alternada é a predominante por ser coincidente com a corrente que é distribuída pela rede elétrica, mas a corrente contínua apresenta algumas vantagens: separação do circuito elétrico do prédio (minimizando o impacte do aumento de carga na rede no arranque do motor) e por permitir um controlo superior da sua velocidade do motor através de moduladores de tensão(Janovský, 1999).

No ponto seguinte será analisado o sistema de tração de cabos, nas suas três vertentes, acoplado a motores elétricos, pela sua predominância.

1.2.1 Sistema por tração de cabos

Este sistema – tração de cabos – consiste, de forma básica, num sistema de roldanas. O elevador circula, para cima ou para baixo, num espaço denominado de poço. No topo desse poço – na

vulgarmente denominada “casa das máquinas” – é colocada uma roldana, acoplada a um motor elétrico que fornece o momento necessário para o movimento do sistema. Nessa roldana, circulam os cabos de aço que sustentam o sistema, assentes em gornes. O elevador está suspenso num dos extremos dos cabos e o contrapeso no outro extremo.

A inclusão de um contrapeso no sistema tem dois objetivos: assegurar a tração dos cabos e poupar energia. O primeiro objetivo é importante para manter o contacto máximo entre os cabos de tração e os gornes da roldana, garantindo o controlo sobre o sistema. O segundo objetivo é atingido uma vez que o motor elétrico apenas terá que fornecer a diferença entre o momento necessário para fazer subir um dos extremos e a força gravítica que o outro extremo exercerá por si só, independentemente do sentido do movimento da cabina.

Para atingir este objetivo, o contrapeso possui o peso da soma da cabina, dos respetivos acessórios com uma percentagem da capacidade de carga. Na generalidade dos casos, essa percentagem é de quarenta por cento(McCain, 2007) para elevadores com engrenagens, mas o valor dependerá do sistema de tração e do sistema de controlo aplicado.

Dentro dos sistemas de tração por cabos, os sistemas são subdivididos em três categorias: elevador de tração com engrenagens, elevador de tração sem engrenagens e elevador sem casa das máquinas(“Basic types of Elevators,” n.d., “Sobre elevadores,” n.d.; Janovský, 1999).

Os elevadores mais comuns são os de tração com engrenagem, uma vez que à saída do motor elétrico é colocado um sistema de engrenagem redutor, aumentando o binário produzido. Esta aplicação, apesar de limitar a velocidade, permite a utilização de um motor com menor potência, reduzindo o custo da instalação. A velocidade habitual de elevadores deste tipo é entre 1 a 2,5 metros por segundo, permitindo cargas até 1600 quilogramas. A paragem deste tipo de elevadores é feita através de um travão de controlo elétrico entre o motor e a engrenagem redutora(“Sobre elevadores,” n.d.). Recentemente, existem aplicações de tração com engrenagens que permitem velocidades superiores – de até 5 metros por segundo – através de engrenagens helicoidais duplas de alta eficiência acopladas a motores elétricos trifásicos com a variação da velocidade a ser conseguida através da manipulação da frequência da tensão(Janovský, 1999).

Os elevadores de tração sem engrenagens são a escolha típica para arranha-céus, pois permitem velocidades muito elevadas, acima dos 2,5 metros por segundo do sistema convencional acima referido.

O elevador sem casa das máquinas tem a grande vantagem de, como o próprio nome indica, não exigir o espaço de uma casa das máquinas convencional. O sistema consiste em componentes mais

compactos, colocados no topo do poço do elevador. Este sistema é aplicado em construções de até 30 andares(“Sobre elevadores,” n.d.). Na Figura 3, abaixo, é possível verificar a diferença entre os três sistemas de tração por cabos: com engrenagens (à esquerda), sem engrenagens (ao centro) e sem casa das máquinas (à direita)(“Sobre elevadores,” n.d.).

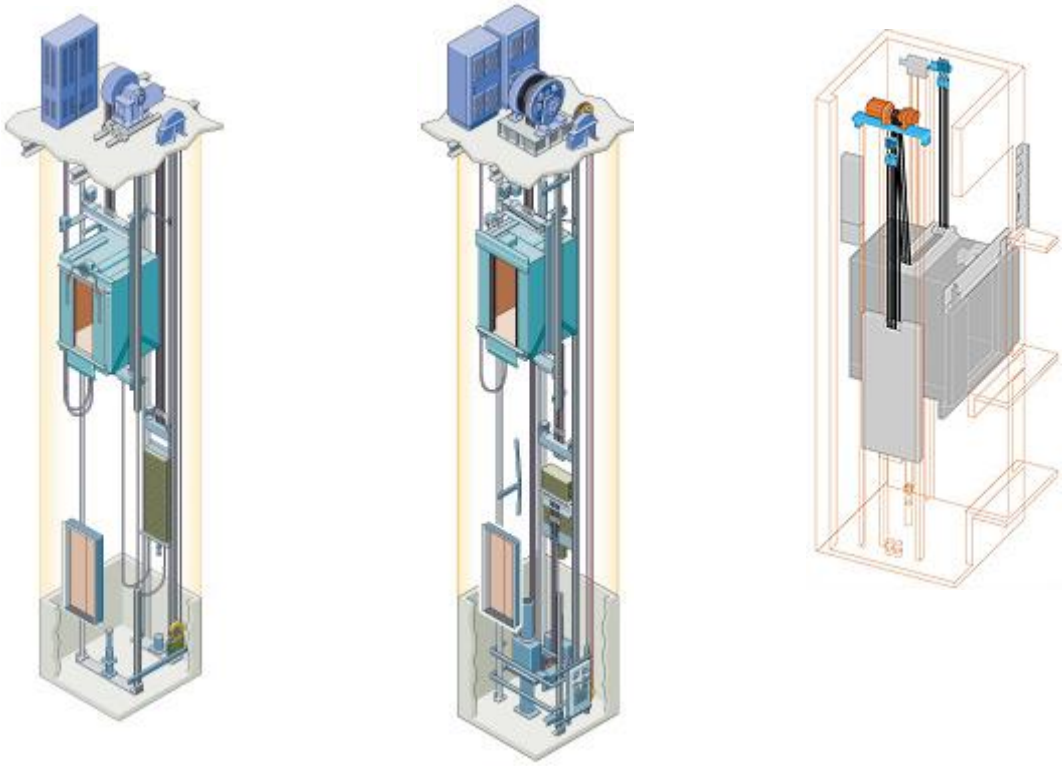


Figura 3 – Esquema simplificado do posicionamento dos componentes de um elevador nos três sistemas de tração por cabos

É possível verificar a poupança de espaço, no caso do sistema sem casa das máquinas, uma vez que todo o sistema de tração é instalado no topo do poço. Os outros dois sistemas, embora distintos, não apresentam um ganho tão significativo.

1.2.2 Segurança

Ao longo dos anos, foram adicionados e melhorados diversos sistemas de segurança para todo o sistema de elevadores.

Hoje, os elevadores possuem vários sistemas de segurança, quer ativos – que têm como objetivo evitar que o acidente ocorra – quer passivos – que têm como objetivo minimizar as consequências de um eventual incidente ou acidente.

Exemplos destes sistemas de segurança são(Craighead, 2009; Gray, 2002; Smith, 2011):

- Um dispositivo de auto-nivelamento da cabina que reduz significativamente os riscos de tropeço por parte dos utilizadores na entrada e saída do elevador, bem como o conforte geral da viagem;
- O poço e a cabina são totalmente fechados, impedindo, por exemplo, que as roupas, acessórios ou membros dos passageiros fiquem presos em algum dos componentes do sistema, ou numa das portas dos andares;
- Um sistema integral de bloqueio de portas impede que o elevador inicie o movimento a menos que todas as portas do poço desse elevador estejam fechadas e trancadas, incluindo as portas de cada andar e a porta da própria cabina;
- Um sistema fotoelétrico – normalmente por infravermelhos – responsável por detetar movimento à entrada e/ou obstrução do caminho das portas do elevador, que para o movimento das portas e as faz retornar à posição de totalmente abertas, impedindo o esmagamento de bens ou pessoas;
- Habitualmente, entre seis a oito cabos de aço a içar o elevador, em que cada um tem capacidade para suportar o peso total do elevador na sua capacidade máxima;
- Sensor de peso que impede o início do movimento caso a capacidade do elevador seja ultrapassada;
- Um governador centrífugo que impede o elevador de se movimentar demasiado rápido ou de entrar em queda livre. Este possui um cabo ligado diretamente à cabina e ao contrapeso. Caso o elevador exceda a velocidade para a qual foi concebido, o governador centrífugo acionará um interruptor de segurança que atuará um travão no motor do elevador, que, habitualmente, é suficiente para parar a cabina. Se este travão não for suficiente e a velocidade da cabina continuar a aumentar, o governador centrífugo irá tracionar o cabo que o liga à cabina, fazendo disparar um, ou mais, braços contra as próprias guias do elevador, que estão fixas à estrutura do edifício, parando a cabina. Este sistema é baseado no sistema apresentado por Elisha Otis em 1854;
- Em locais com um elevado risco sísmico, são adicionados dispositivos com o intuito de detetar movimentações do próprio edifício, desativando os elevadores. Ao sentir movimentações, os dispositivos enviam um sinal ao sistema de controlo de elevadores. O sistema de controlo de elevadores entra em modo de segurança e ordena a cada elevador para se dirigir ao andar mais próximo do sentido atual do movimento, abre as portas automaticamente e bloqueia qualquer movimento dos elevadores. Os elevadores só poderão voltar a ser utilizados após a

reinicialização dos dispositivos. Dependendo da Legislação local, após a probabilidade de réplicas, um indivíduo qualificado – um técnico de manutenção, técnico de elevadores ou o responsável pelo edifício – deverá inspecionar o poço do elevador, as cabinas, o contrapeso e a casa das máquinas em busca de possíveis danos ou acidentes antes de reiniciar os dispositivos de deteção de sismos e o sistema de controlo de elevadores. As falhas poderão ser: desalinhamento das guias do elevador ou contrapeso, empeno das portas do poço em cada andar, rutura ou desencaixe de um dos cabos de tração do elevador e contrapesa ou movimentação de equipamentos na casa das máquinas;

- No ponto mais baixo do poço, é colocado um mecanismo de amortecimento da cabina, para que, na eventualidade de uma queda livre, a queda da cabina seja amortecida, reduzindo os prejuízos. Este sistema é habitualmente constituído por molas, mas também poderá incluir borracha ou outro material com elevada deformação elástica. Este sistema foi inicialmente pensado por Otis Tufts.

Na casa das máquinas existem diversos comandos centrais que permitem às equipas de controlo, manutenção ou de salvamento controlar individualmente ou globalmente os elevadores do sistema. As denominações e funcionalidades poderão ser diferentes entre fabricante, modelos ou aplicação, mas de uma forma geral os controlos serão os seguintes(Craighead, 2009):

- Um interruptor de energia para o motor que permite desligar o fornecimento de energia ao sistema e, por consequência, bloquear a movimentação dos elevadores;
- Um dispositivo de operação manual que permitirá a seleção do andar de destino para o elevador, independentemente das solicitações. As portas serão automaticamente fechadas na posição atual, exceto se o sensor fotoelétrico detetar um objeto ou pessoa no caminho, e automaticamente abertas no destino, permanecendo nesse andar até que outro andar seja selecionado ou o funcionamento normal seja reposto, que habitualmente requer uma chave;
- Um botão para abrir a porta da cabina, independentemente da posição do elevador, mesmo que esteja entre andares. Esta funcionalidade poderá ser útil para resgatar pessoas em caso de falha do sistema ou outro incidente;
- Um interruptor para controlar o funcionamento da iluminação e ventilação dentro da cabina;
- Um interruptor para aceitar ou não o *input* do sensor fotoelétrico à entrada da cabina. Este sistema poderá ser útil para obrigar as portas a fecharem, de forma a iniciar o movimento, em casos em que de facto há um objeto em frente à cabina, mas que não represente perigo, ou avaria do próprio sensor.

Dentro da cabina do elevador, normalmente posicionados no painel de seleção do destino pretendido, existem diversos mecanismos e botões que permitem executar algumas ações, quer aos passageiros, quer às equipas de manutenção ou de salvamento. Estes mecanismos, botões e respetivas ações e particularidades estarão dependentes da Legislação local e do ano em que o sistema de elevadores foi instalado(Jarboe & O'Donoghue, 2007):

- Iluminação interior, independente da iluminação normal da cabina, que permitirá iluminar o painel da cabina durante até quatro horas, através de uma bateria instalada na própria cabina. Esta luz poderá estar montada no próprio painel ou no topo da cabina;
- Um mecanismo que permita parar o movimento do elevador, ligado diretamente ao motor na casa das máquinas. Esta função poderá ser ativada ou por um simples botão ou por uma chave;
- Um botão que feche as portas, podendo agilizar o processo de paragem do elevador;
- Um botão que abra as portas do elevador, para, por exemplo, impedir que estas choquem com um bem ou uma pessoa;
- Um botão que toca um alarme, de forma a alertar os responsáveis do edifício para um incidente dentro do elevador, que deverá possuir a sua própria fonte de energia com autonomia de até uma hora;
- Um intercomunicador, bidirecional, com uma bateria que permita uma autonomia de até quatro horas. Será da responsabilidade do gestor do edifício manter este intercomunicador ligado a um serviço de atendimento disponível durante todo o horário de laboração dos elevadores, com um tempo de resposta de até 30 segundos. Este sistema poderá estar ligado com o anterior, mas nunca deverá substituí-lo;
- Um painel que permita controlar o movimento da cabina bem como outras ações através de botões. Este painel é, habitualmente, oculto, e apenas deverá ser acedido por equipas de manutenção ou de resgate, através da abertura por chave.

Abaixo, na Figura 4, estão alguns exemplos dos painéis que poderão ser encontrados no interior da cabina.

Figura 4 – Exemplos de painéis de controlo, no interior da cabina("ElevatorTour's most interesting photos," n.d.)

Todos estes sistemas de segurança que evitam a ocorrência de acidentes e os mecanismos de controlo que ajudam no resgate de passageiros em alguns incidentes, tornam o elevador um dos transportes mais seguros no mundo. Algumas referências consideram-no mesmo a forma mais segura de transportar pessoas(Smith, 2011; Strakosch & Caporale, 2010), dando o mérito à constante evolução nas tecnologias de segurança e formação dada por fabricantes, sindicatos e técnicos de instalação e manutenção, bem como os fornecedores de componentes e sistemas.

É referido ainda que uma parte dos incidentes e fatalidades ocorridas em elevadores são durante a manutenção, algumas por erro do trabalhador. Ainda assim, existem casos onde o incidente ou fatalidade tem como causa uma falha num dos sistemas de segurança ou incorreta instalação do sistema(Smith, 2011).

1.3 Os sistemas de gestão

O sistema de gestão – ou de controlo – de elevadores tem como principal objetivo atribuir um destino ao elevador, tendo por base as chamadas dos passageiros que pretendem utilizar o elevador e os destinos solicitados pelos utilizadores do elevador. Este objetivo é independente do tipo de sistema de gestão e da aplicação.

Existem exceções, como são os sistemas de segurança, vistos no subcapítulo 1.2.2 Segurança, ou um sistema de prioridades, como é, por exemplo, um doente do serviço de urgência de um Hospital. Nestas situações, o sistema de gestão terá ordens e prioridades diferentes, podendo ignorar as chamadas ou os destinos atuais dos passageiros.

Como *inputs*, o sistema de gestão tem as chamadas e os destinos. As chamadas são realizadas por cada passageiro em cada andar, dando a informação ao elevador que deverá parar num determinado andar para receber passageiros, sendo as únicas exceções os pontos enumerados no parágrafo anterior e se a capacidade do elevador estiver perto do limite(Strakosch & Caporale, 2010). Os destinos são os andares em que o elevador deverá parar para permitir a saída de passageiros, que poderá ou não coincidir com as chamadas. As exceções vistas no parágrafo anterior mantêm-se.

Para obter os *inputs*, o sistema de gestão terá que incluir botoneiras. O tipo, quantidade e localização das botoneiras dependerá do sistema de gestão instalado e da Legislação aplicável a cada País e aplicação.

1.3.1 Botoneiras

A botoneira representa o método de *input* dos passageiros para o sistema de gestão e, como tal, desempenha um papel fundamental na eficácia e eficiência do sistema. Isto porque, os dados que o utilizador fornece ao sistema de gestão estão diretamente ligados com a paragem do elevador – para realizar a entrada ou saída de passageiros – e com a atribuição do elevador correto – no sentido da viagem do cliente e com paragem no destino pretendido. Todos estes processos têm como objetivo minimizar o tempo de espera (desde a chamada até à entrada no elevador) e o tempo de viagem (decorrido desde a entrada até a saída do elevador).

No mais básico dos sistemas, com apenas dois andares, o sistema apenas necessitará de um botão à entrada de cada andar, onde é rececionada a chamada do elevador. Se o elevador se encontrar no piso da chamada este abrirá as portas, se o elevador estiver no outro andar este fechará as portas, irá movimentar-se até ao andar onde a chamada teve origem e irá abrir as portas automaticamente. Depois do cliente entrar, este fechará as portas e dirigir-se-á ao outra andar, abrindo automaticamente as portas no destino. Exemplos deste sistema são os monta-cargas tradicionais ou os elevadores de cozinha. Na Figura 5, abaixo, é possível ver um exemplo real.

Figura 5 – Exemplo de um botão de chamada simples

Os sistemas mais comuns contêm dois momentos de *input* por cada passageiro: um à entrada, onde é realizada a chamada do elevador, e outro dentro da cabina, onde é introduzido o destino pretendido. A

botoneira à entrada poderá ter dois formatos: simples, com apenas um botão como mencionado no parágrafo anterior e visível na Figura 5, ou dupla, com um botão a apontar para cima e outro a apontar para baixo. Na Figura 6 abaixo é possível visualizar uma botoneira dupla (à esquerda) e o painel de seleção do destino por parte dos utilizadores no interior da cabina (à direita) ("Elevator control panels," n.d.).

Figura 6 – Exemplos reais de uma botoneira dupla e de um painel de seleção do destino no interior de uma cabina

No caso da botoneira simples, basta pressionar o botão único para realizar a chamada do elevador. Este sistema é vulgar em aplicações de um só elevador. Na botoneira dupla, o utilizador deverá pressionar o botão de acordo com o sentido da posição atual para o destino pretendido: para cima se desejar subir, ou para baixo se pretender descer. Esta abordagem é habitualmente implementada em aplicações com dois, ou mais, elevadores, mas não obrigatoriamente.

Esta diferenciação – para cima ou para baixo – é benéfica porque não só mantém o destino atual do elevador – evitando uma reversão do destino para os passageiros que já estão dentro da cabina – mas também porque diminui o tempo de viagem para o passageiro que pretende entrar, uma vez que o sistema de gestão irá atribuir-lhe o elevador que está a movimentar-se no sentido da sua chamada para o seu destino. O utilizador deverá confirmar o sentido do elevador quando uma das portas do andar se abrir, e visualizar – através do ecrã, normalmente colocado acima da porta do elevador – qual o sentido dessa cabina. Na Figura 7 abaixo é possível ver dois exemplos reais da indicação do sentido que o elevador irá tomar após receber os passageiros, normalmente posicionados acima da porta de cada elevador("Elevator Dial Floor Indicator," n.d.).

Figura 7 – Exemplos reais da indicação do sentido que o elevador irá tomar

Se o passageiro cometer um erro na botoneira dupla, clicando para baixo quando o sentido pretendido é para cima, por exemplo, estiver no terceiro andar e pretender viajar até ao quinto andar, este entrará no elevador e pressionará o quinto andar, mas o elevador – que atualmente possui o sentido descendente – irá realizar o total de descidas até atingir o destino ou a chamada mais baixa, obrigando o passageiro que cometeu o erro a viajar no interior da cabina até esse andar mais baixo e a realizar novamente todo o percurso, desta vez ascendente, até ao seu destino: o quinto andar. Nesse percurso, o elevador poderá parar em vários andares, dependendo do número de chamadas e destinos, aumentando significativamente o tempo de viagem deste utilizador. Além disto, este tipo de erros aumenta o peso a transportar – refletindo-se no consumo energético do motor – e irá representar um lugar ocupado, podendo este ser ocupado por alguém que realmente queira tomar esse sentido.

Estes dois sistemas – de botoneira simples e dupla – exigem um segundo *input*, dentro da cabina, onde o utilizador irá selecionar o destino pretendido.

Para fundir estes dois *inputs*, foram desenvolvidas botoneiras que contêm todos os andares possíveis, ou apenas um teclado numérico, que permite selecionar o andar pretendido. Em sistemas mais avançados, a botoneira poderá ser substituída por um ecrã tátil. Estas botoneiras são colocadas em cada andar, junto às portas dos vários elevadores, permitindo aos passageiros selecionarem o andar pretendido antes de entrar na cabina(“Analysis of Elevator Performance and Passenger Demand with Destination Control,” n.d.). Neste caso, o sistema de gestão irá interpretar o sentido do movimento – tendo em conta o andar onde surgiu o pedido e o destino pretendido – e irá alocar a cabina adequada. Para orientar os passageiros, habitualmente, é atribuída uma letra a cada elevador, e o utilizador recebe o feedback do sistema de gestão num visor colocado na própria botoneira, ou no ecrã tátil. Também poderão ser utilizados números, mas as letras permitem uma diferenciação em relação aos

números dos andares, sendo mais fácil a compreensão para novos utilizadores do sistema. Adicionalmente, o sistema poderá indicar ao passageiro a direção que deve seguir para encontrar a cabina que lhe foi atribuída.

Abaixo, na Figura 8, estão exemplos reais de sistemas de introdução do destino antes de entrar na cabina: botoneira numérica, com atribuição do elevador correto (à esquerda); ecrã tátil, para selecionar o destino (ao centro) e atribuição da cabina correta (à direita)("ThyssenKrupp Destination Dispatch Elevator," n.d.).

Figura 8 – Exemplos reais de sistemas de introdução do destino

1.3.2 Algoritmo de controlo

A função básica de um sistema de gestão de elevadores é gerir as paragens dos elevadores, mediante as chamadas e os destinos selecionados. Este objetivo é independente do tipo de aplicação ou número de elevadores.

Além desta função, o sistema de gestão de elevadores tem outras funções, que podem ser divididas por camadas de importância, ou prioridade, mas sempre com um objetivo comum: melhorar a eficiência do elevador, mantendo a eficácia(Barney & Lutfi Al-Sharif, 2015). O algoritmo é o responsável por estabelecer o próprio sistema de controlo e definir prioridades, tomando decisões, tendo por base os *inputs*, e dando ordens através de *outputs*.

Um algoritmo simples, de um sistema de gestão de apenas um elevador, poderá ser descrito da seguinte forma(Setchi, 2010):

- Movimentar-se num determinado sentido, parando em todos os andares onde existam chamadas ou destinos;
- Alterar o sentido quando atinge os andares de topo – superior ou inferior – ou quando não tem nenhuma chamada ou destino para andares subsequentes no atual sentido;
- Parar, caso não existam chamadas ou destinos.

O caso dos elevadores, quer em termos de algoritmo, quer em termos de controlo e instrumentação é vulgarmente utilizado como caso-estudo em livros, teses ou aulas teóricas e práticas(Alagar & Periyasamy, 2011; Forero, 2010). Com algumas simplificações – nomeadamente ignorar a redundância ou a inclusão de apenas um elevador – o sistema de gestão de elevadores é um caso muito interessante para estudo, pois envolve *inputs*, tomada de decisão e *outputs*.

A redundância de chamadas é uma das funções principais dos sistemas de gestão de um grupo de elevadores – adequável à maioria das aplicações. Exemplo disto é a ocorrência do mesmo tipo de chamadas no mesmo andar: se num determinado andar ocorre uma chamada para subir, o sistema deverá verificar se já foi atribuído um elevador a esse tipo de pedido (mesmo andar, mesmo sentido), ou se um elevador já tem esse andar como destino e irá seguir o mesmo sentido que o solicitado. Esta gestão de recursos permite diminuir o tempo de viagem, uma vez que apenas um elevador irá parar no andar onde ocorreu a solicitação, ao invés de vários elevadores.

Um sistema de gestão de um grupo de elevadores tem como objetivo comandar um grupo de elevadores, tendo acesso a todos os *inputs* e podendo controlar todos os *outputs*, incluindo todos os elevadores. O objetivo é melhorar a eficiência global do sistema.

A capacidade de movimentação de um grupo de elevadores com um sistema de gestão de grupo é superior à soma da capacidade de movimentação de cada elevador a atuar individualmente(Barney & Lutfi Al-Sharif, 2015). Por esta razão, é altamente recomendado que os elevadores sejam agrupados e instalados numa área central do edifício, ao invés de serem espalhados irmãmente pelo edifício e atuarem de forma singular.

Além da redundância, mencionada anteriormente, outra vantagem desta abordagem é a possibilidade de retirar um dos elevadores de serviço – para manutenção, poupança de energia ou imobilização – mantendo a acessibilidade aos outros andares, muitas vezes sem os utilizadores se aperceberem.

1.3.3 Sistemas baseados no destino

Um dos sistemas de gestão de grupo de elevadores mais avançados, e que se prolifera gradualmente, é o sistema baseado no destino dos passageiros, em Inglês: *destination dispatch*. Este sistema exige o uso de botoneiras que permitam inserir o destino pretendido antes de entrar na cabina, como os exemplos da Figura 8.

Existem diversas abordagens a este sistema, mas o foco de todas elas é diminuir o impacte que a adição de um novo passageiro irá ter no sistema.

Há abordagens em que o foco é agrupar os passageiros em setores de andares, alcançando o objetivo mencionado anteriormente indiretamente, Outras abordagens calculam o valor real desse impacte e atribuem ao novo passageiro o elevador com o menor impacte. No subcapítulo seguinte – 1.3.4 Sistema ETD – será demonstrado o método de cálculo.

Em Portugal, este sistema pode ser encontrado no Centro Hospitalar e Universitário de Coimbra, EPE (CHUC), no bloco central do edifício principal. A intervenção consistiu no *retrofitting* deste sistema de gestão no sistema de tração que estava instalado. Este sistema de gestão, associado à adição de controladores – que permitem ajustar a velocidade dos motores anteriormente instalados – permitirão uma economia energética estimada em 201 MWh por ano("Prémio EDP: Energia Elétrica e Ambiente 2011," 2010).

Esta mudança, bem como outras políticas de redução do consumo de energia e do impacte ambiental, permitiram ao CHUC conquistar a vitória na Categoria B – Serviços do Prémio EDP: Energia Elétrica e Ambiente de 2011, juntamente com loja de Loures da empresa IKEA Portugal.

1.3.4 Sistema ETD

Como exemplo, será analisado o sistema ETD ("*Estimated Time to Destination*", ou Tempo Estimado até ao Destino) da empresa TKE® (ThyssenKrupp® Elevator)(Peters & Smith, 2009).

O sistema ETD foi baseado no sistema ETA ("*Estimated Time of Arrival*", Tempo Estimado de Chegada), introduzido em 1985 pela empresa Dover Elevator®, adquirida mais tarde pela ThyssenKrupp®, formando a TKE®. O ETA tinha como objetivo reduzir o tempo de espera, selecionando o elevador do grupo que poderia responder à chamada no menor tempo possível. Isto é, quando um cliente faz uma chamada, o sistema calcula o tempo aproximado que cada elevador irá levar para chegar ao andar onde a chamada teve origem. O elevador com o menor tempo é, normalmente, o elevador que irá ficar encarregue de responder a esta chamada.

O sistema ETD tem em conta o tempo total da viagem: o tempo de espera (em que o cliente aguarda pela chegada do elevador correto ao seu andar) somado com o tempo de viagem (que o cliente passa dentro da cabina até chegar ao seu destino). Além disto, o sistema tem em conta o tempo que cada elevador irá levar para atender cada chamada (ETA) e o impacte que cada nova alocação irá ter no tempo de cada passageiro que já se encontra dentro da cabina.

Cada vez que um passageiro realiza uma chamada, o sistema irá calcular o ETD, em segundos, para a viagem pretendida em cada elevador. Como exemplo: o passageiro y realiza uma chamada no andar zero (rés-do-chão) e, por consequência, o sistema de gestão irá calcular o ETD do passageiro y se

viajasse em cada elevador. O ETD do elevador x é calculado através da soma do ETA desse elevador x – desde a posição onde este se encontra atualmente até ao andar zero, onde o passageiro y realizou a chamada, tendo em conta todas as paragens que este irá realizar durante esse percurso até à chegada ao andar zero – com o tempo que o elevador x irá levar até ao destino pretendido pelo passageiro y, tendo também em conta todas as paragens que este irá realizar durante o percurso até ao destino.

Adicionalmente, o sistema também irá calcular o SDF ("*System Degradation Factor*", ou Fator de Degradação do Sistema) que a alocação do novo passageiro irá causar em todos os passageiros do sistema. O $SDF_{x,z}$ é o atraso que a alocação do passageiro y irá trazer para o passageiro z dentro do elevador x, com z a percorrer os n passageiros dentro do elevador x. O $SDF_{x,z}$ é obtido subtraindo o tempo de viagem até ao destino do passageiro z com o novo passageiro y ao tempo de viagem até ao destino do passageiro z sem o novo passageiro y.

O TC ("*Total Cost*", ou Custo Total) da alocação do novo passageiro y ao elevador x é igual o resultado da soma do somatório do $SDF_{x,z}$ com o ETD_x. Abaixo, na Equação 1, é possível ver a fórmula matemática deste cálculo.

Equação 1 – Cálculo do custo total da adição de um novo passageiro no sistema ETD(Peters & Smith, 2009)

$$\boldsymbol{TC_x = \sum_{z=1}^{n} SDF_{x,z} + ETD_x}$$

O sistema de gestão irá alocar o novo passageiro ao elevador com o menor Custo Total.

1.3.5 Sistemas com legado

Os sistemas de gestão de grupos de elevadores com legado têm como objetivo adaptar a oferta à procura, de uma forma constante. Podendo fazer ajustes ao algoritmo atual ou alterá-lo por completo.

A origem deste sistema esteve na ideia de colocar vários elevadores no piso de entrada – seja o rés-do-chão ou o parque de estacionamento subterrâneo do edifício – permitindo a entrada imediata de passageiros na cabina, de forma a diminuir o tempo de espera(Barney & Lutfi Al-Sharif, 2015).

A determinação do algoritmo apropriado – pelas equipas de instalação – deverá ser baseada no padrão e intensidade do tráfego que o sistema de controlo enfrenta todos os dias.

Na análise de tráfego será importante compreender os picos e a hora a que estes ocorrem. Os picos habituais são:

- Subida: tipicamente de manhã, onde existem várias solicitações do rés-do-chão para subir para os respetivos andares;

- Descida: tipicamente no final do dia de trabalho, onde é registado um elevado número de pedidos para descer dos respetivos andares para o rés-do-chão;
- Misto: tipicamente durante as horas de almoço, onde as pausas podem cruzar-se temporalmente, registando-se uma procura mista entre subidas e descidas.

As restantes movimentações, embora importantes para a determinação dos algoritmos apropriados, não são consideradas picos.

A seleção do algoritmo apropriado pode ser feita de forma manual ou automática. A abordagem manual exige que um operador esteja disponível e selecione o algoritmo apropriado, normalmente com uma chave no painel de controlo da casa das máquinas. A seleção automática permite duas abordagens: por horário, onde o algoritmo é alterado com base em horas pré-definidas; ou por um sistema de análise, que avalia o tráfego atual – através da carga dos elevadores, da quantidade de chamadas ou do padrão das chamadas, sentidos e destinos – e, ou faz ajustes ao algoritmo atual, ou altera por completo o algoritmo a utilizar(Barney & Lutfi Al-Sharif, 2015).

1.4 Estado da arte

O crescimento de vários Países emergentes tem mudado as aspirações de várias pessoas e o governo de muitas empresas em várias indústrias: tecnológica, comunicações, automóvel, construção e até no transporte vertical. Uma das mudanças é o "paisagem urbana", onde Nações se enchem de orgulho em relação à majestosidade das suas cidades, sendo Nova Iorque um exemplo mundial(Rao, 2012).

Os arranha-céus nascem e evoluem até ganharem simbolista e terem impacte positivo na economia local. Vários arranha-céus ao redor do Globo são autênticos pontos turísticos adorados e visitados por milhares de pessoas, como o Empire State Building, em Nova Iorque; as Torres Petronas, em Kuala Lumpur, Malásia; e o Burj Khalifa, no Dubai.

Até por volta do ano 2000, a indústria do transporte vertical era considerada por alguns uma indústria parada, sem sofisticação tecnológica, sem desafios, com fraca estética e com funções pouco amigas do cliente(Rao, 2012).

O avanço tecnológico – através do poder computacional acessível, da simulação, e da instrumentação necessária – permitiu à indústria do transporte vertical evoluir. A crescente sensibilização e consciência ambiental também despoletaram um interesse maior em soluções mais eficientes, e com menor impacte ambiental.

Hoje, através dos sistemas de gestão de grupos de elevadores baseados no destino, é possível diferenciar os elevadores atribuídos, permitindo instalar elevadores de menor velocidade – e menor

consumo energético – em conjunto com elevadores com maior velocidade. Os elevadores de menor velocidade irão transportar os passageiros até os andares intermédios, e os de maior velocidade até os andares de topo(Rao, 2012).

Este sistema de gestão, juntamente com o sistema com legado, permite retirar do serviço alguns elevadores nas horas em vazio, regressando ao serviço nas horas de pico.

Estas duas mudanças permitem reduzir os custos energéticos e as horas de trabalho, reduzindo a manutenção do sistema.

Em janeiro de 2016("Hoorah! Shanghai Tower crowned world's second tallest skyscraper," n.d.), foi inaugurado a Shanghai Tower, na China, com 121 andares e 632 metros de altura, tornando-se o edifício mais alto da China e o segundo mais alto do mundo – depois do Burj Khalifa, no Dubai, com 828 metros de altura. Este edifício conta com o elevador mais rápido do mundo.

Instalado pela empresa Mitsubishi Electric, o elevador pode atingir velocidades de até 1080 metros por minuto (ou 18 metros por segundo, ou 64,8 quilómetros por hora)("Mitsubishi Electric's new elevator is the fastest on Earth," n.d.).

Três elevadores irão viajar diretamente do segunda andar da cave até ao andar do observatório no centésimo décimo nono andar. Para atender o hotel, existe um grupo de quatro elevadores de dois andares que irão viajar diretamente do rés-do-chão até ao centésimo primeiro andar, onde fica a entrada do hotel do edifício. O elevador de emergência irá ser o elevador que percorrerá a maior distância, do terceiro andar da cave até ao centésimo vigésimo primeiro andar, numa distância total de 578,5 metros. Contando com todos estes anteriormente mencionados, o edifício contém um total de 106 elevadores.

Os sete elevadores que irão atingir a velocidade mencionada, irão incorporar guias de cabos ativas, para reduzir a vibração e garantir a tensão necessária; coberturas do tejadilho das cabinas aerodinâmicas, permitindo a velocidade máxima e reduzindo as vibrações e ruídos na cabina; e controlo pneumático para compensar pela mudança na pressão atmosférica ao longo do percurso("Mitsubishi Electric to Install World's Fastest Elevators in Shanghai Tower," n.d.).

1.4.1 Simulação

Os modelos mais recentes de sistemas de gestão de grupo de elevadores – como são a alocação de passageiros por destino e os sistemas com legado tiveram – tiveram na sua base testes e resultados de simuladores.

Um simulador que se destaca é o *software* Elevate®(Barney & Lutfi Al-Sharif, 2015), que permite simular e analisar o tráfego em elevadores, com suporte para várias configurações e aplicações: elevadores com dois andares, diferentes velocidades, diferentes destinos ou capacidades("About Elevate," n.d.). Este *software* corre em ambiente Windows™ e foi desenvolvido pela empresa Peters Research, de Londres.

Outra inovação desta empresa é o Elevate Live™ que permite, em tempo real, verificar o estado do sistema de elevadores("About Elevate Live," n.d.).

Este sistema não é a única ferramenta utilizada na indústria, mas é uma das mais promovidas. Mas, tendo em conta a disponibilidade de partilha e a defesa da propriedade intelectual das empresas deste setor, a informação sobre os *softwares* e as ferramentas utilizadas é escassa.

Mas a necessidade e a aplicação de simulação nesta área é clara(Barney & Lutfi Al-Sharif, 2015; "Current Technology and Future Developments in Elevator Simulation," n.d.; Henri Hakonen & Siikonen, 2009; Zhang & Zong, 2014), pois os modelos atingem graus elevados de complexidade, como é o exemplo da Shangai Tower – anteriormente referido – onde centenas de elevadores circulam, com várias restrições e propósitos diferentes.

1.4.2 O futuro

Uma referência(Barney & Lutfi Al-Sharif, 2015) que aponta como possibilidade futura o reconhecimento facial, que permitirá, por exemplo, confirmar a entrada no edifício de um sujeito conhecido, reconhecer o seu padrão de utilização, preparar um elevador para este utilizar e indicar-lhe o elevador adequado com uma saudação: "Bom dia Senhor João, queira utilizar o elevador A".

2. SIMULAÇÃO

Este capítulo será inteiramente dedicado à simulação realizada no âmbito da presente Dissertação, utilizando o *software* Simio®.

O primeiro subcapítulo será dedicado à descrição do *software* utilizado e os restantes à simulação propriamente dita.

2.1 O *software* Simio®

O *software* Simio®, utilizado na presente Dissertação, é desenvolvido pela empresa americana Simio® LLC. e deve o seu nome à expressão: "*Simulation Modeling framework based on Intelligent Objects*", ou estrutura de Simulação de Modelos baseados em Objetos Inteligentes("About Simio," n.d.).

A empresa Simio LLC. foi fundada em 2005 por C. Dennis Pegden, fundador e ex-CEO da empresa Systems Modeling Corporation, desenvolvedora do software Arena®, e adquirida pela empresa Rockwell Automation® em 2000, Pegden liderou o desenvolvimento da linguagem de simulação SLAM®, da linguagem SIMAN®, do sistema de animação Cinema® e dos simuladores Arena® e Simio®("C. Dennis Pegden," n.d., "Management Team," n.d.). David Sturrock, Vice-Presidente de Operações da Simio® LLC., foi outra peça importante do desenvolvimento do *software* Simio®, fazendo parte da bibliografia de apoio à manipulação do software e membro ativo das comunidades de apoio do software, nomeadamente o fórum oficial("Simio Software Discussion Forum," n.d.).

A empresa Simio® LLC. é muito jovem no seio do mercado competitivo da simulação, mas com um elevado foco na melhoria. Durante o último ano foram libertadas quatro atualizações, sendo que a primeira e a última foram de grande dimensão, evoluindo da versão 6 para a 7 em janeiro de 2015, e para a versão 8 em dezembro de 2015.

O sistema de modelação utilizado pelo Simio® foi desenvolvido para simplificar a construção de modelos promovendo uma mudança no paradigma da modelação de orientação a processos para uma orientação a objetos. Os objetos podem ser construídos pelos modeladores da empresa – e libertados ou melhorados numa atualização – ou o utilizador poderá construir os seus próprios objetos, podendo reutilizá-los em vários modelos ou dentro de outro objeto. Apesar da estrutura do Simio® ser orientada a objetos, este também suporta um modelo integrado com vários paradigmas de modelação, incluindo: eventos, processos ou baseada em agentes; em sistemas discretos ou contínuos.

2.1.1 Objetos

Um principiante poderá preferir utilizar os objetos presentes na biblioteca do *software*, mas este poderá optar por construir o seu próprio objeto, adequado às suas necessidades, com inteligência, de forma a ser utilizado num modelo com hierarquias.

Um objeto poderá ser uma máquina, um robot, avião, cliente, médico, tanque de guerra, autocarro, barco, ou outra coisa que possa ser necessária ao modelo. Um modelo é construído combinando objetos que representem os componentes físicos do sistema, com o intuito de se assemelhar com o sistema real. A lógica e a animação poderão ser combinadas num só passo, de forma a permitir que a lógica do objeto – a mudança de um estado, por exemplo – seja visível na animação.

Os objetos são construídos utilizando o próprio conceito de orientação a objetos, mas – ao contrário de outras aplicações – é feita em ambiente gráfico, não exigindo qualquer programação.

O ambiente de construção de um objeto no Simio® é idêntico ao de construção de um modelo, sendo central no *design* do *software*, e é referido como o princípio da equivalência. Ao construir um modelo, este é, por definição, um objeto, que poderá ser incorporado noutro modelo. Por exemplo, ao adicionar duas máquinas e um robot a um modelo de uma célula de trabalho, o próprio modelo da célula de trabalho é um objeto que poderá ser adicionado o número de vezes necessárias num outro modelo. A célula de trabalho será um objeto neste novo modelo, tal como as máquinas e o robot foram objetos no modelo que é foi a célula de trabalho.

Os objetos presentes na biblioteca do *software* dividem-se em cinco tipos(Pegden & Sturrock, 2014):

- *Fixed*: um objeto que possui uma localização específica e fixa, como poderá ser uma máquina;
- *Link*: fornece um caminho pelo qual entidades se podem mover, como são os *paths*;
- *Node*: representa uma interceção entre um, ou mais, *links* de entrada e/ou de saída. Os *nodes* poderão também estar associados a objetos fixos (*fixed*), de forma a permitir a entrada e saída de entidades do interior do objeto;
- *Entity*: doravante referida como entidade, é a definição de objetos dinâmicos que poderão ser criados e destruídos ao longo da simulação, mover-se numa rede de *links* e *nodes*, e entrar e sair de objetos fixos (*fixed*) através dos *nodes* associados a esse objeto;
- *Trasporter*: define um tipo especial de entidade que poderá recolher e entregar outras entidades em *nodes*. Exemplo deste tipo de objeto é o veículo.

Estes cinco tipos representam definições genéricas do *software*, podendo o utilizador transformar uma entidade em *transporter*, como será feito na simulação analisada no subcapítulo 2.5 Entidade da presente Dissertação.

2.1.2 Processos

Os processos são os responsáveis pela lógica que permite aos objetos agirem de forma "inteligente".

Os processos podem ser executados no início ou no fim da simulação (*run*), no fim do aquecimento inicial ou antes do aquecimento final (*warm up period*), ao entrar ou ao sair de um no *node*, por ser disparado por outro processo, ao quando são criadas ou destruídas entidades, entre outros. Esta flexibilidade permite criar um processo no momento exato em que faz sentido. Abaixo, na Figura 9, são exibidos dois exemplos de possibilidades para despoletar processos, em *nodes* (à esquerda) e em veículos (à direita).

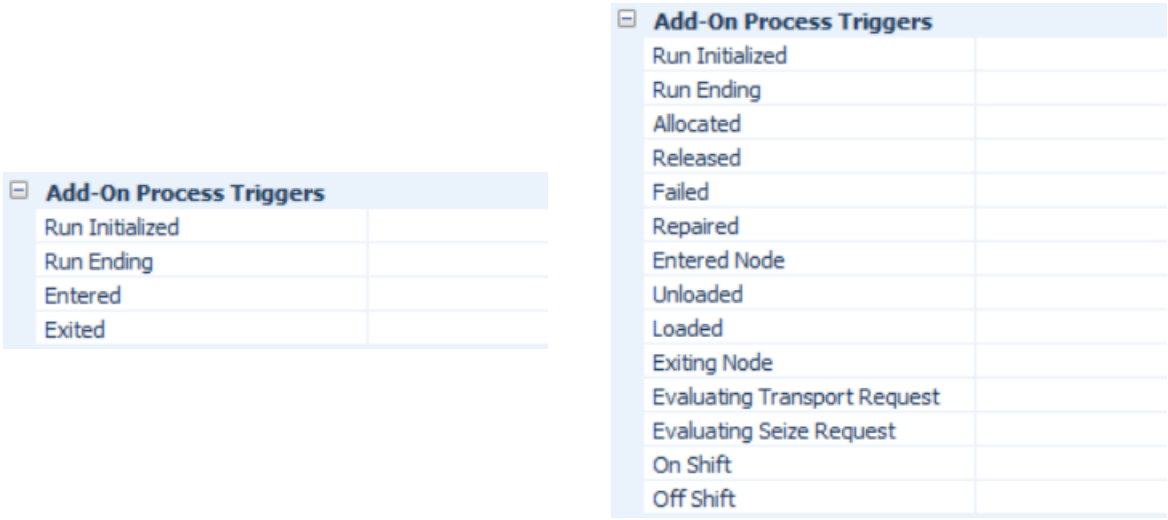


Figura 9 – Exemplos de possibilidades para despoletar o início de processos

Os processos são executados por *tokens*. Cada entidade poderá ter mais do que um *token*, mas cada *token* representa apenas uma entidade. Isto é, um *token* é criado para uma entidade específica no momento em que o processo é despoletado, mas uma entidade poderá criar vários *tokens* por ter despoletado mais do que um processo.

Um único processo poderá ter vários *tokens* a executá-lo, inclusive, de entidades diferentes.

Os processos podem ser utilizados para vários fins: criar ou destruir entidades, requisitar recursos, atribuir um *node* como destino a uma entidade, transferir entidades, manipular variáveis, avaliar possibilidades, entre outros. Para alcançar esta lógica, os processos são criados utilizando passos (*steps*), em que cada passo permite realizar uma tarefa diferente, e, dentro de cada passo, há diferentes possibilidades.

Na Tabela 1, abaixo, é possível ver os vários passos utilizados nos processos das simulações da presente Dissertação, referindo o seu nome, um exemplo gráfico e uma brave descrição da função de cada um. De forma a facilitar a identificação de cada passo e melhorar a visualização dos processos, foi atribuída uma cor diferente a cada passo.

Tabela 1 – Resumo dos passos (*steps*) utilizados nos vários processos das simulações: denominação, exemplo gráfico e descrição da sua função

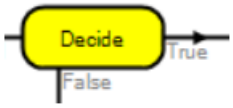
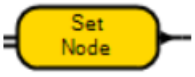
Denominação	Exemplo gráfico	Descrição da função
Begin		Indica o ponto de início do processo
Assign		Permite atribuir um valor a uma variável do sistema, através de um valor específico ou expressão lógica
Create		Permite criar objetos específicos do tipo entidade (*output created*) ou copiar entidades já existentes (*outputs original* e *created*)
Decide		Através de probabilidade ou expressão lógica, o *token* irá sair do output apropriado: *true* ou *false*, isto é, verdadeiro ou falso
Delay		Permite adicionar uma espera temporal ao *token*
Execute		É utilizado para ordenar a execução de um processo específico
Find		É útil para procurar um valor, segundo uma expressão, num ou mais indexantes, por exemplo, o valor mínimo de uma matriz
Fire		Permite disparar um evento
Set Node		É utilizado para definir o *node* de destino de uma entidade
Tally		Grava o valor observado na expressão introduzida quando o *token* passa por este passo

Tabela 1 – Resumo dos passos (*steps*) utilizados nos vários processos das simulações: denominação, exemplo gráfico e descrição da sua função

Denominação	Exemplo gráfico	Descrição da função
Transfer		Permite transferir objetos de uma localização para outra, por exemplo, de um *transfer node* para uma *station*
Wait		Permite parar o *token* até que o evento definido seja disparado
End		Representa o final do processo

As diferentes possibilidades, dentro do mesmo passo, poderá ser visualizada no passo *Decide*, por exemplo, permitindo utilizar probabilidade ou uma condição lógica para determinar a saída do *token* desse passo.

2.1.3 Eventos

Os processos, quando comparados com programação, têm a vantagem de ser possível dar-lhes a noção de tempo, quer pelo passo *wait*, quer pelo passo *delay*.

O passo *delay* permite fazer esperar o *token* uma determinada quantidade de tempo. O *token* só será libertado para o passo seguinte depois de esgotado esse tempo. Para tal, será necessária a atribuição de um espaço temporal exato, como: cinco segundos, sete minutos ou duas horas

O passo *wait* permite parar o *token* até que um determinado evento seja disparado, através do passo *fire*. Este passo é especialmente importante, pois, aliado à possibilidade de vários *tokens*, permite melhorar a simulação de forma significativa.

Exemplos da aplicabilidade dos eventos e do passo *wait* são: aguardar até que o tanque fique cheio, apenas solicitar uma palete quando a quantidade mínima de caixas for atingida ou aguardar neste *node* até que o elevador chegue(Pegden & Sturrock, 2014).

2.2 Veículo vs. Entidade teórica

Para elucidar da necessidade da comparação, é necessário enquadrar as duas abordagens.

A abordagem veículo irá utilizar o objeto pré-concebido pela equipa do *software* Simio® chamado *vehicle*. Por ser um objeto pré-concebido pela própria empresa, o utilizador não terá acesso aos seus

processos, eventos, variáveis e outros parâmetros essenciais para a sua "inteligência", isto é, tomada de decisão.

Por se tratar de um objeto pré-concebido, o utilizador não será obrigado a construir processos complexos de decisão, uma vez que os desenvolvedores do *software* o fizeram antecipadamente e dotaram-no de propriedades alteráveis pelo utilizador, tais como: capacidade de carga, tempo de descarregamento, método de seleção da tarefa, entre outros.

Ao mesmo tempo, por estar bloqueado o acesso a tais parâmetros e lógica, o utilizador estará limitado, ficando refém dos processos genéricos, que desconhece completamente. Este facto poderá impedir a concretização do objetivo inicial: simular o comportamento de um elevador.

Ainda assim, sendo um veículo, o seu intuito principal é transportar entidades da sua origem até ao seu destino. Isto é, essencialmente, o objetivo do elevador: transportar um cliente do andar onde realizou a chamada (origem) até ao andar onde pretende ir (destino).

Em suma: a utilização de um veículo encurta o esforço inicial, mas limita o utilizador na "inteligência" e nas funcionalidades adicionais do objeto.

Por contraste, a utilização de uma entidade como elevador exigirá a construção completa da lógica de um elevador. Isto é, o utilizador terá que desenvolver processos específicos para a atividade a simular. Como exemplo, o utilizador terá que criar um processo específico para criar o objeto e coloca-lo no *node* inicial, atividades que com um objeto da biblioteca são realizadas num ambiente gráfico.

A vantagem desta abordagem é a flexibilidade do objeto e a adaptação completa à atividade proposta, pois o utilizador, como desenvolvedor do seu próprio modelo, terá acesso total a todos os parâmetros e lógica, podendo moldá-los de acordo com o algoritmo pretendida.

A decisão entre utilizar um objeto veículo ou uma entidade não é óbvia ou direta. Em cada aplicação, tendo em conta os objetivos, parâmetros e necessidades; é necessário simular e estudar as duas possibilidades, a fim de compreender as vantagens e limitações de cada um dos modelos. É esta a análise central do presente capítulo.

2.3 Transversalidade dos modelos

Ambos os modelos pretendem simular o mesmo: o comportamento de um elevador e, portanto, partilham vários pontos. Este subcapítulo pretende analisar essas semelhanças, ao mesmo tempo que enquadra a simulação, revelando o modelo base.

O modelo contém sete andares, identificados por andar 1, andar 2, andar 3, andar 4, andar 5, andar 6 e andar 7. A fim de permitir uma direta relação entre expressões lógicas e andares correspondentes,

evitou-se a utilização de denominações como andar 0 ou andar menos 1. Esta decisão permite também uma melhor compreensão dos processos e do modelo a novos participantes.

Em cada andar existe um *source*, responsável pela criação das entidades do tipo cliente, que representam os passageiros na simulação. Cada cliente terá um destino, e deverá ser destruído no *sink* do andar correspondente ao seu destino.

Cada andar possui um *path* que liga o *source* do respetivo andar até ao *node* onde cada cliente irá realizar a sua chamada e aguardar pelo elevador. Os *paths* não permitem ultrapassagem – e consequente sobreposição – com o intuito de melhorar a imagem da simulação.

Quando o elevador chega ao andar correspondente, ocorrerão dois tipos de transferência de clientes. Uma de receção, onde os clientes que aguardam o elevador são transferidos do *node* onde esperavam para a fila (ou *ride station*) do elevador, veículo ou entidade. A outra transferência será a de entrega, onde os clientes que têm como destino o andar onde o elevador parou são transferidos da fila do elevador para o *node out* correspondente.

Os clientes que entrarem no elevador ficarão na fila do mesmo até alcançarem o andar que desejam.

Ao saírem do elevador, os clientes irão seguir o *path* que liga o *node out* para onde foram transferidos até ao respetivo *sink*, onde serão destruídos.

Este processo é transversal às duas simulações e representa a entrada, transporte e saída de passageiros num sistema de um elevador.

2.4 Veículo

Sendo um objeto pré-concebido, o veículo apresenta alguma facilidade inicial na conceção deste modelo, como mencionado no subcapítulo 2.2 Veículo vs. Entidade teórica. Isto é, uma vez que o objeto veículo já possui os seus próprios processos, cabe ao utilizador dotar a entidade cliente de "inteligência".

A entidade cliente precisa de ser criada, ser-lhe atribuído um destino, esperar pelo elevador, realizar a transferência do *node* para a fila do elevador, sair do elevador no andar correspondente e ir até ao *sink* do andar de destino.

2.4.1 Processos

A criação da entidade é feita ao mesmo tempo que lhe é atribuído um destino, de forma aleatória, através de uma função específica que o *software* disponibiliza no *source*, visível na Figura 10 abaixo.

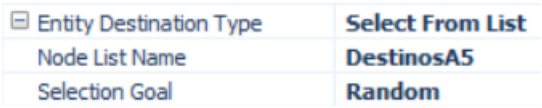


Figura 10 – Seleção do destino do cliente no *source* através de listas, na simulação com o objeto veículo

A lista, referenciada no *source*, é definida pelo utilizador. Desta forma, são introduzidos todos os *nodes* referentes aos andares da simulação, exceto o andar do *source* onde o cliente teve origem. Uma das listas é visível na Figura 11 abaixo.

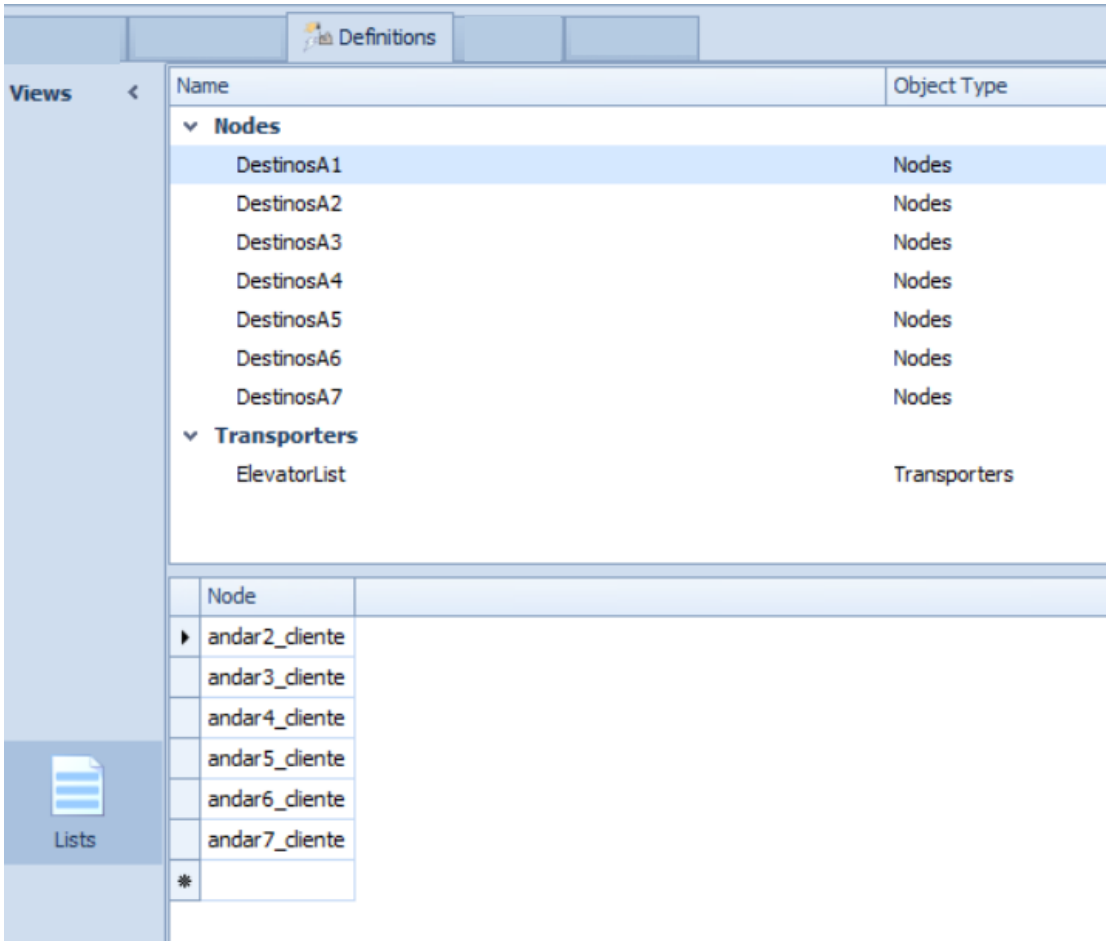


Figura 11 – Definição de uma lista com os andares de destino dos clientes, na simulação com o objeto veículo

Ao sair do *source*, a entidade cliente despoletará um processo. A configuração deste despoletar no *source* – tal como a configuração enunciada na Figura 10 acima – é a visível na Figura 12 abaixo.

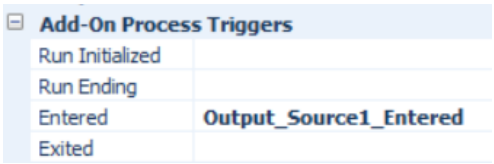


Figura 12 – Configuração para despoletar o processo "*Output_Source1_Entered*" no *source*, na simulação com o objeto veículo

O processo despoletado no *source* tem como intuito atribuir à entidade cliente o valor correspondente ao andar de origem na variável "origem" da entidade "cliente". A expressão, portanto, é a seguinte: "Cliente.origem". O processo e a expressão lógica são os visíveis na Figura 13 abaixo.

Figura 13 – Processo "*Output_Source1_Entered*" e expressão lógica utilizada, na simulação com o objeto veículo

O valor 1, atribuído à expressão "Cliente.origem", refere-se ao andar 1. Nos restantes andares o processo é semelhante, atribuindo o valor correspondente ao andar. Esta variável será utilizada mais à frente na simulação.

Em seguida, a entidade cliente irá percorrer o *path* que a levará até ao *node* onde esperará pelo elevador. Para obrigar a entidade cliente a esperar pelo elevador, será necessário ativar a opção *Ride on transport* (ou Embarcar no veículo) em todos os *node* de entrada. Adicionalmente será necessário escolher que transporte será escolhido (específico ou de uma lista) e a lógica de seleção do transporte. Na presente simulação, como apenas existe um elevador, é esse que está selecionado. Abaixo, na Figura 14, é possível confirmar as opções escolhidas.

Ride On Transporter	**True**
Transporter Type	Specific
Transporter Name	**Vehicle1**
Reservation Method	Reserve Closest
Selection Goal	Preferred Order
Selection Condition	

Figura 14 – Opção *Ride on transport*, no *node* de entrada, para obrigar a entidade cliente a esperar pelo elevador, na simulação com o objeto veículo

A entidade irá, portanto, esperar pela chegada do elevador *Vehicle1*.

Na simulação com o objeto veículo, os *nodes* onde as entidades cliente aguardam pelo elevador são coincidentes com aqueles onde o elevador passa, tal como visível na Figura 15 abaixo.

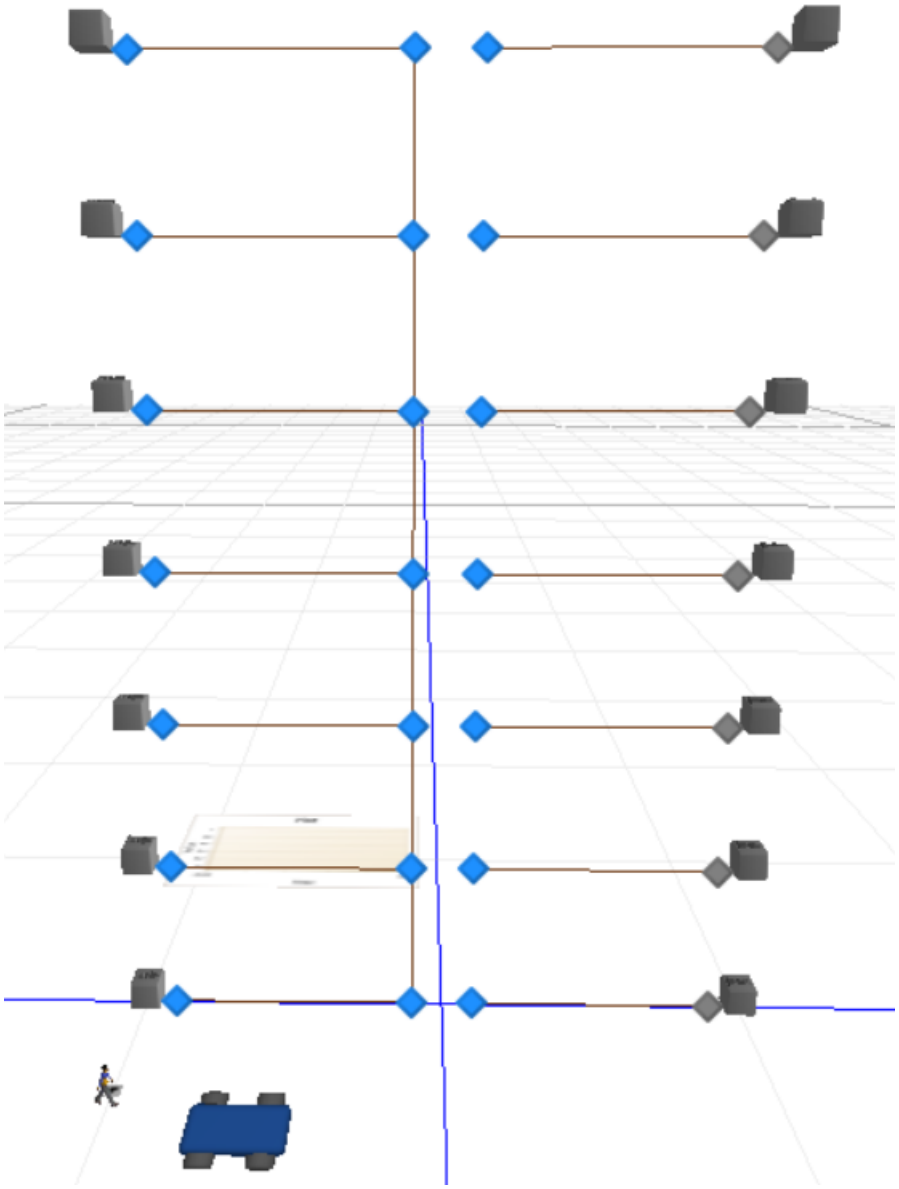


Figura 15 – Modelo de simulação do elevador, com o objeto veículo, estando a simulação parada

Esta coincidência de *nodes* permite tirar partido da componente pré-concebida do objeto veículo, mantendo o foco no contraste entre a simulação do elevador como veículo – objeto pré-concebido – e entidade – objeto modelado pelo utilizador.

A vantagem desta abordagem é a transferência automática do *node* para o veículo e do veículo para o *node*. Isto é alcançado porque o cliente reservou o veículo (visível na Figura 14) – transferindo-se do *node* para a fila do elevador automaticamente – e o *node* de destino atribuído na lista do *source* (visível na Figura 11) é coincidente com os *nodes* onde o veículo circula – transferindo-se da fila do elevador para o *node* de destino de forma automática. Os passos *transfer* responsáveis por estas transferências estarão nos processo do veículo, ocultos ao utilizador, visto tratarem-se de objetos da biblioteca do *software*.

A entidade cliente, ao sair do elevador, terá que ser finalmente transferida para o *node out* do respetivo andar. Isto é atingido, despoletando um processo aquando da entrada de uma entidade no respetivo *node* – visível na Figura 16, abaixo – e o processo associado – enunciado na Figura 17, também abaixo.

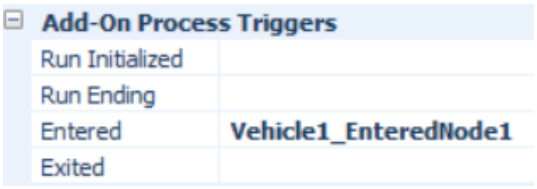


Figura 16 – Configuração para despoletar o processo "*Vehicle1_EnteredNode1*" no *node* de entrada, na simulação com o objeto veículo

O processo associado é o apresentado na Figura 17 abaixo, consistindo em dois passos *decide* e um *transfer*.

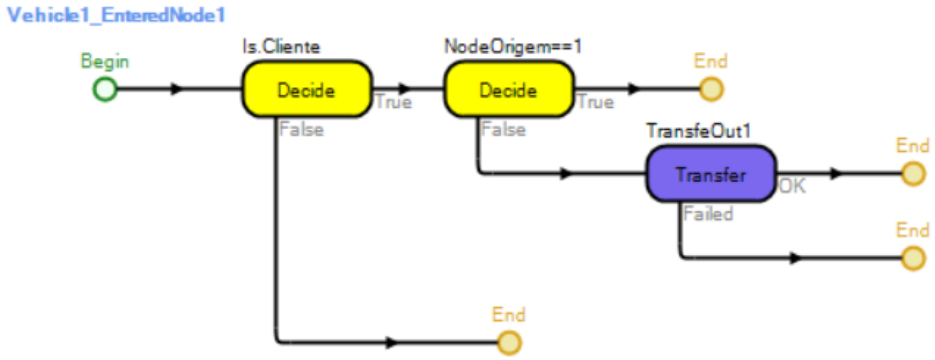


Figura 17 – Processo "*Vehicle1_EnteredNode1*", responsável pela saída das entidades no andar de destino, na simulação com o objeto veículo

O primeiro passo *decide* consiste numa proteção, executando a validação de que o *token* executante pertence a uma entidade do tipo cliente, excluindo um potencial *token* do veículo, visto que este também circula nos *nodes* de entrada.

O segundo *decide* utiliza a variável atribuída no processo exposto na Figura 13 – e despoletado no *source* de origem, como configurado na Figura 12 – para realizar o seguinte teste: a origem do cliente foi no mesmo andar do *node* atual?

Se sim (*true*), significará que o cliente possui um destino diferente do andar do *node* atual, e deverá aguardar pelo elevador, que o levará ao seu destino final. O processo é então terminado (*end*).

Caso o teste seja negativo (*false*), significará que o cliente foi deixado pelo elevador, e o presente andar é o seu andar de destino. O *token* é então dirigido para um passo *transfer*, que irá transferir o cliente do *node* atual para o *node out* do andar correspondente. Abaixo, na Figura 18, está representada a lógica deste passo no processo.

Basic Logic	
From	CurrentNode
To	Node
Node Name	Out1

Figura 18 – Lógica do passo *transfer* para a saída dos clientes no andar pretendido, na simulação com o objeto veículo

Ao ser transferido para o *node out* correspondente, as entidades cliente irão percorrer o *path* de saída do andar de destino e terminar no *sink*, onde serão destruídas.

Abaixo, na Figura 19, são visíveis os vários estágios deste processo: criação de clientes e posterior caminhada no *path* de entrada, espera pelo elevador, entrada no elevador, saída do elevador e caminhada final no *path* até ao *sink*.

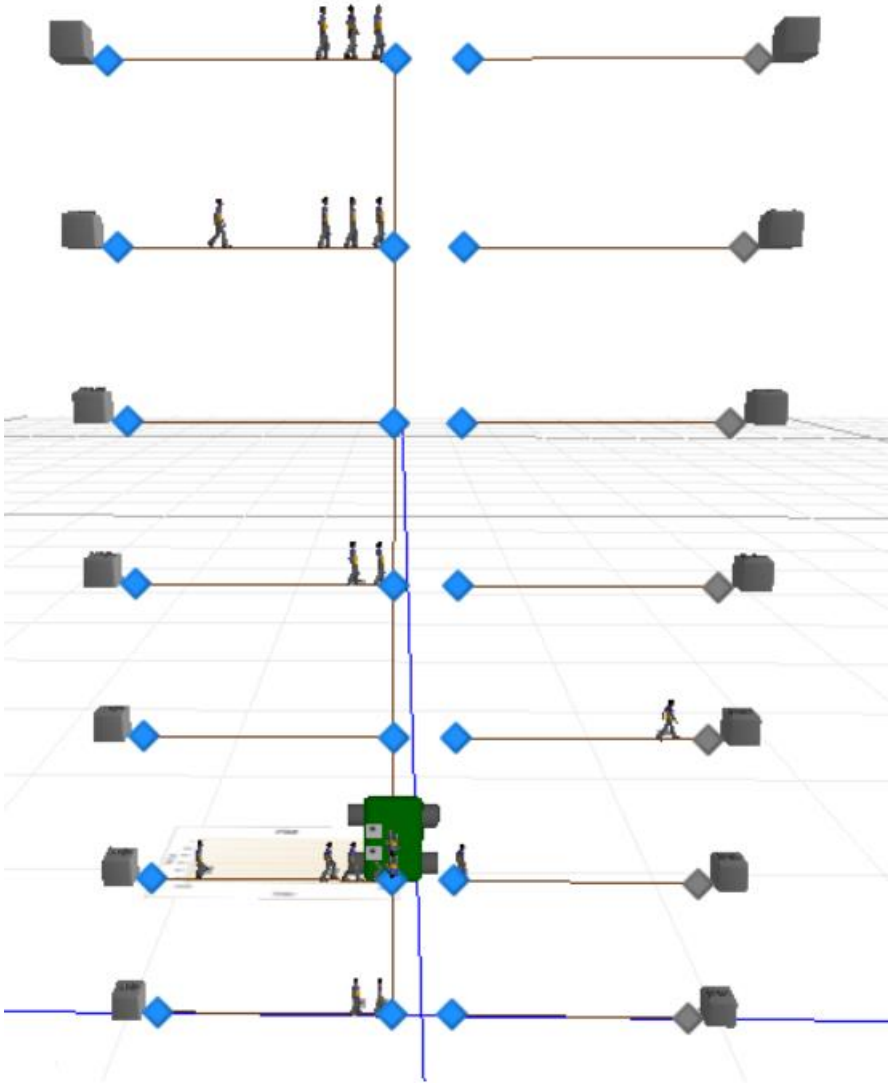


Figura 19 – Modelo de simulação do elevador, com o objeto veículo, a decorrer

O objeto veículo possui vários parâmetros que necessitam de ser definidos. Abaixo, na Figura 20, são visíveis os parâmetros definidos para esta simulação.

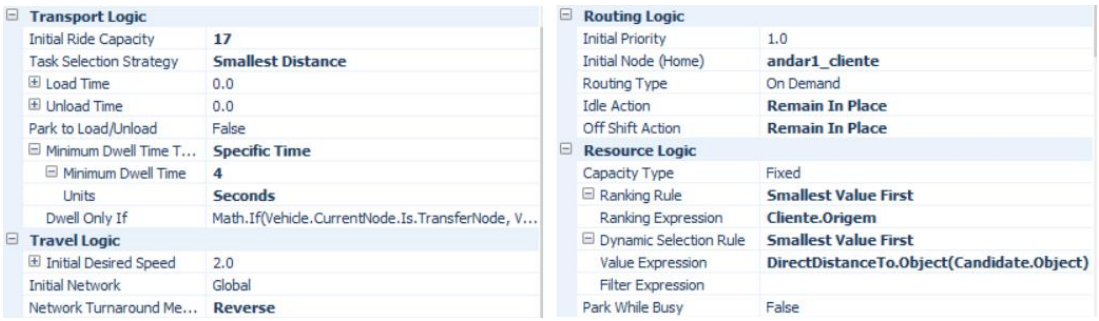


Figura 20 – Parâmetros definidos para o objeto veículo

Os valores a negrito destacam os valores alterados pelo utilizador, mantendo no estilo normal os restantes parâmetros que possuem os valores pré-concebidos pelos desenvolvedores do *software*.

O valor do parâmetro *Initial ride capacity* refere-se à capacidade da fila do elevador. O valor escolhido foi 17 para não prejudicar o desenrolar da simulação, com vista a não ser uma limitação à própria simulação. A opção *Task selection strategy* visa definir a estratégia que o objeto terá para selecionar a próxima tarefa para receber ou libertar um cliente. O parâmetro *Dwell time* é utilizado para definir o mínimo de tempo que o veículo ficará parado no *node* onde receberá ou libertará um cliente. O valor selecionado é de quatro segundos. A condição pré-concebida – *Dwell only if* – é utilizada para apenas creditar o *dwell time* se o elevador estiver num *transfer node* (o tipo de *node* que permite transferências de entidades) e se a atual fila do elevador ainda permitir entradas.

O *Network turnaround method* define a forma como a animação do objeto irá atuar no momento de inversão do sentido após a saída de um *node*. *Reverse* é a animação mais apropriada para recriar um elevador, isto é, o objeto irá circular em frente e, ao inverter o sentido, irá fazer algo semelhante à marcha atrás nos veículos.

O *Initial node (dome)* define o local onde o objeto irá iniciar a simulação, que, este caso, será o andar mais baixo do modelo. A *Idle action* e *Off shift action* são definidas para *Remain in place*, de forma a manter o veículo em posição durante os momentos em que não tem requisições ou sai de turno.

O parâmetro *Routing type* permanece em *On demand*, pois é essencial para o desenvolvimento dos parâmetros do *Resource logic*: *Routing rule* e *Dynamic selection rule*. A outra opção é uma rota fixa, determinada pelo utilizador. *On demand* permitirá adaptar a rota do veículo às solicitações.

Ambos os parâmetros *Routing rule* e *Dynamic selection rule* têm como regra selecionar o valor mínimo das respetivas expressões: *Ranking expression* e *Value expression*. O primeiro parâmetro, *Routing rule*, visa ordenar os pedidos de requisição do veículo. Para tal, foi selecionada a expressão "Cliente.origem", de forma a colocar todos os pedidos por ordem do andar onde a requisição do veículo ocorreu. O parâmetro *Dynamic selection rule* permite definir a forma como o veículo irá selecionar o próximo pedido, da lista anteriormente ordenada, quando está livre de pedidos. A

expressão selecionada é "DirectDistanceTo.Object(Candidate.Object)". Uma vez que o candidato selecionado será o de valor menor, o veículo irá selecionar o candidato cliente mais próximo. Nas expressões *Ranking expression* e *Value expression* deverá utilizar-se a denominação *Candidate*, ao invés de *Entity*, uma vez que a expressão irá realizar um *ping* à fila de requisições do veículo.

2.4.2 Conclusão

O modelo com o elevador a ser recriado por um objeto veículo é, em suma, simples de conceber. Apesar de requerer conhecimento do *software* e dos vários parâmetros que permitem a utilização automática de um veículo (visíveis nas seguintes figuras: Figura 12, Figura 14 e Figura 20), os processos concebidos pelo utilizador são simples e de com um ou três passos.

Para desenvolver a simulação foram utilizados catorze processos: sete processos no *source*, com um passo cada, de forma a atribuir o valor origem a cada cliente (visível na Figura 13); e sete processos, um em cada *node* de entrada de cada andar, para filtrar os clientes que querem utilizar o elevador daqueles que saem do elevador e têm o respetivo andar como destino (processo visível na Figura 17).

Os parâmetros *Routing rule* e *Dynamic selection rule*, bem como as suas respetivas expressões: *Ranking expression* e *Value expression* representam o cerne do controlo do veículo, manipulando a forma como este irá definir a sua rota. Mas, no modelo enunciado acima, foram evidentes as limitações destes. O veículo respondia apenas às solicitações dos clientes que estavam na sua fila, apenas rececionando novos clientes quando parava num *node* para libertar um passageiro.

Este facto invalida totalmente a simulação de um elevador utilizando o objeto veículo.

2.5 Entidade

A simulação através do objeto entidade obrigará à construção total da lógica de um elevador. Isto é, o utilizador terá que desenvolver processos específicos para a atividade a simular.

Isto torna a simulação complexa, podendo dificultar o processo de compreensão da lógica desenvolvida. Para evitar tal acontecimento, serão relembrados os objetivos centrais de um elevador ao mesmo tempo que será feito o enquadramento desses objetivos com os processos concebidos.

2.5.1 Enquadramento

Tal como visto no subcapítulo 1.3.2 Algoritmo de controlo, o algoritmo simples, de um sistema de controlo de apenas um elevador, poderá ser descrito da seguinte forma(Setchi, 2010):

- Movimentar-se num determinado sentido, parando em todos os andares onde existam chamadas ou destinos;
- Alterar o sentido quando atinge os andares de topo – superior ou inferior – ou quando não tem nenhuma chamada ou destino para andares subsequentes no atual sentido;
- Parar, caso não existam chamadas ou destinos.

O primeiro ponto é dividido em duas vertentes: manter o sentido atual e parar em todos os andares, que contêm chamadas ou destinos. A primeira vertente – manter o sentido atual – é alcançada através de um passo *decide* denominado "Descer ou Subir?", visível no Anexo I – Processo "*OnRunInitialized*" do objeto entidade, que, através de uma condição que verifica se o *node* de destino é superior ou inferior ao atual, divide o processo em dois: processo de subida e processo de descida. Após esta bifurcação, é cumprida a segunda vertente, em que o *token* do elevador irá percorrer, através de um *find*, todos os andares subsequentes – acima se estiver a subir, ou abaixo se estiver a descer – verificando se existem chamadas. As chamadas estão guardadas num vetor, cujo número de posições é igual ao número de andares. O valor é comutado entre 1 e 0, conforme existem ou não chamadas no andar do índice. Abaixo, na Figura 21, está um pormenor do processo "*OnRunInitialized*", que exibe a bifurcação.

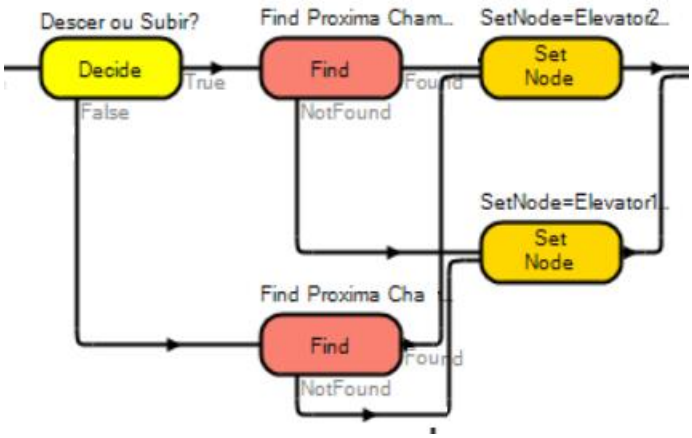


Figura 21 – Pormenor do processo "*OnRunInitialized*", exibindo a bifurcação do sentido de subida e descida, na simulação com a entidade

É também através desta bifurcação que é cumprido o segundo ponto do algoritmo simples relembrado acima, uma vez que, ao atingir o *node* do topo superior ou inferior, o *decide* denominado de Subir ou Descer? irá contemplar esse momento.

A garantia de que os destinos dos clientes que estão dentro do elevador são atendidos é conseguida através de outro processo, denominado de "Cheguei", visível no Anexo II – Processo "Cheguei" do objeto entidade. Este processo é iniciado sempre que a entidade entra num *node* central, uma vez que

é nesse momento que é possível deixar sair os passageiros. O processo "Cheguei" irá parar o elevador se uma das duas condições for verdadeira: o elevador estiver no seu destino (definido no *set node* da bifurcação vista anteriormente), uma vez que existe uma chamada nesse *node*; ou se houver um cliente no interior do elevador que tem como destino o andar do respetivo *node*. Abaixo, na Figura 22, é destacado um pormenor do processo "Cheguei", onde é possível verificar o teste de destino.

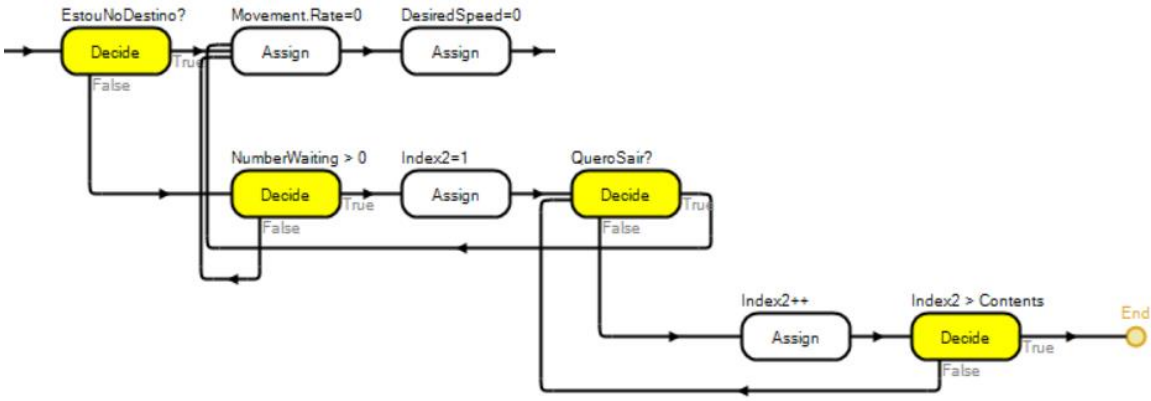


Figura 22 – Pormenor do processo "Cheguei", exibindo o método de deteção de destino

O primeiro *decide* analisa se o *node* atual é o pretendido, isto é, se o *node* atual contém chamadas. Caso a condição seja verdadeira (*true*), o processo irá parar o elevador, colocando as variáveis *Movement.rate* e *DesiredSpeed* a zero. Esta paragem irá cumprir o terceiro ponto do algoritmo simples revisto acima

Se a condição for falsa (*false*), o *token* seguirá para a verificação de destino das entidades no interior de elevador, analisada abaixo.

De forma a incluir as situações onde não existem chamadas e o elevador está vazio – como é o caso do início da simulação, eventuais momentos em que o elevador está vazio e por segurança – o *token* irá encontrar outro *decide* que irá verificar se existem clientes no interior do elevador, testando a expressão Elevador.Ride.Contents.NumberWaiting > 0. Caso o resultado seja falso, o *token* seguirá para o ramo superior do processo, onde o elevador será parado, pois não existem chamadas ou destinos, visto que não existem clientes dentro do elevador.

Se o resultado for verdadeiro (*true*) – significando que existem clientes dentro do elevador – o *token* seguirá para um processo de verificação de destino dos vários clientes dentro do elevador. Através da variável Index2, inicializada a 1 no primeiro passo *assign*, é feito um teste lógico, utilizando um passo *decide*: Elevador.Ride.Contents.ItemAtIndex(Elevador.Index2).Cliente.Destino == Elevador.IDNodeAtual, isto é, o destino do cliente na posição de Index2 na fila do elevador é correspondente com o *node* atual? Se a resposta a esta pergunta for sim (*true*), o *token* seguirá para o ramo superior do processo, onde o elevador será parado, pois existe, pelo menos, um cliente cujo destino é o *node* atual.

Quando a resposta é não (*false*), o processo incrementará uma unidade à variável Index2, de forma a percorrer os restantes clientes da fila do elevador. Em seguida, é feito um teste lógico para determinar se todos os clientes da fila do elevador foram contemplados pelo teste de confirmação do destino. Em caso negativo (*false*), o *token* irá regressar ao teste lógico de igualdade do destino com o *node* atual. Em caso afirmativo, o processo irá terminar (*end*), uma vez que não existem chamadas, nem destinos, no *node* atual. Nesta situação, o elevador irá seguir viagem para o *node* seguinte.

Através destes passos, nos dois processos executados pela entidade elevador, são satisfeitos todos os pontos do algoritmo simples de um elevador.

Ainda assim, os processos mencionados não foram examinados na totalidade, nem todos os processos deste modelo foram mencionados. O seguinte subcapítulo irá analisar os restantes passos e processos do modelo de elevador através de uma entidade.

No modelo de elevador através de uma entidade existem onze processos: oito executados pelos clientes e três pela entidade, que representa o elevador.

2.5.2 Processos cliente

Assim que cada cliente é criado é executado um processo composto por dois passos: um para atribuir o andar onde a entidade foi criada à variável destino; e outro para, de forma aleatória, atribuir o destino à entidade. Na Figura 23, abaixo, é possível ver este processo, que é replicado para cada andar, totalizando sete processos.

Figura 23 – Processo "*Output_Source1_Entered1*", realizado pelo cliente quando é criado, na simulação com o objeto entidade

O cliente executa outro processo, denominado por "Chama", visível no Anexo III – Processo "Chama" do objeto entidade. Este processo é executado por um *token* do cliente assim que este chega ao *node* de entrada do andar de origem, acompanhando-o até sair do elevador para o *path* de saída.

O primeiro *assign* para o cliente, atribuindo à variável DesiredSpeed o valor de zero. De seguida é realizada um teste lógico, num passo *decide*, denominado de "ElevadorEstaAqui?": Elevador1[1].IDNodeAtual == Andares[Cliente.Origem].FloorID, com o intuito de confirmar se o elevador se encontra no *node* do cliente. Para facilitar esta comparação, foi desenvolvida uma tabela

de correspondência, que irá permitir comparar os *nodes* do elevador com o do cliente. Esta tabela é demonstrada abaixo, na Figura 24.

Andares

	Floor ID	Floor Node	To Path	Transfer Out
▸ 1	1	andar1	Enter1	Out1
2	2	andar2	Enter2	Out2
3	3	andar3	Enter3	Out3
4	4	andar4	Enter4	Out4
5	5	andar5	Enter5	Out5
6	6	andar6	Enter6	Out6
7	7	andar7	Enter7	Out7

Figura 24 – Tabela de correspondência entre *node* de entrada, *node* de saída e *paths*, na simulação com o objeto entidade

Esta correspondência é necessária visto que os *nodes* não coincidem fisicamente, por obrigatoriedade do *software*. A diferença entre os vários *nodes*: de entrada, do elevador e de saída é visível abaixo, na Figura 25.

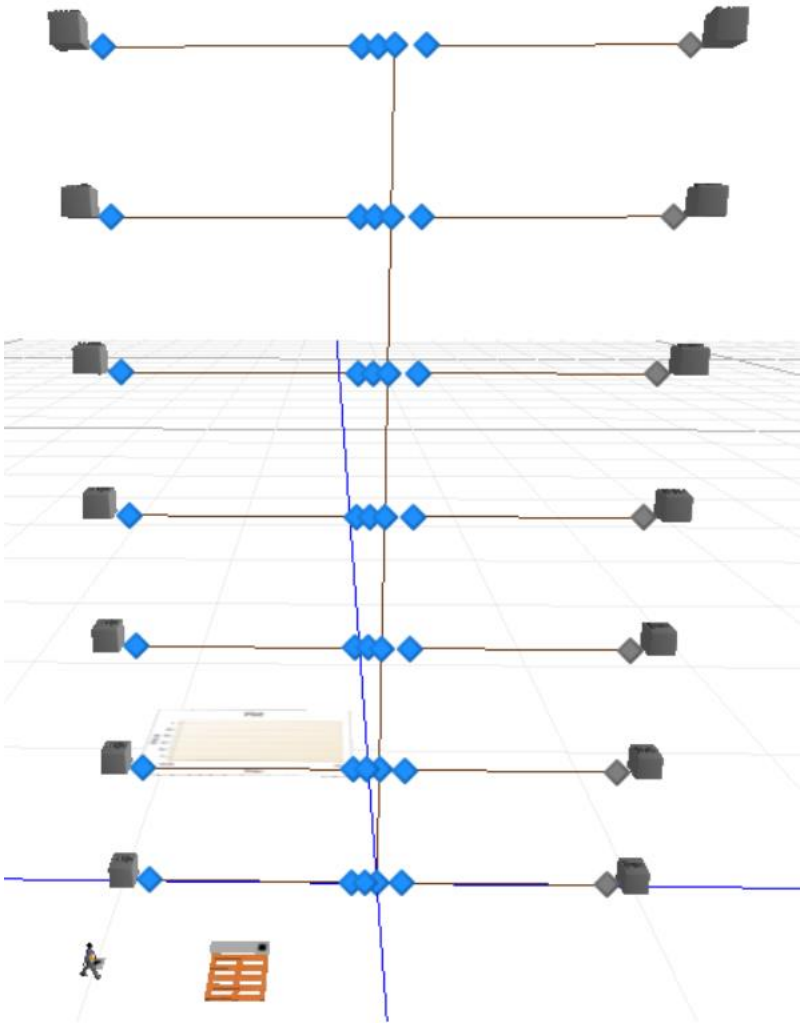


Figura 25 – Modelo de simulação do elevador, com o objeto entidade, estando a simulação parada

Caso os *nodes* coincidissem, a entidade seguiria para o *node* de destino, uma vez que iria conseguir alcança-lo na mesma *network* de *nodes*, seguindo os *paths* de cliente e os *paths* do elevador. Adicionalmente, foi colocado um terceiro *node*, entre o *node* de entrada do cliente e o *node* do elevador. Este terceiro *node* tem como intuito impedir a destruição do cliente, que o *software* realiza caso não tenha um *path* à sua frente, apesar de ter sida dada a ordem para este parar.

Regressando ao processo "Chama", no passo "ElevadorEstaAqui?", caso este seja *false*, o *token* seguirá para um ramo específico, que terá como intuito: disparar uma chamada e gravar o instante temporal. Este ramo é mostrado abaixo, na Figura 26.

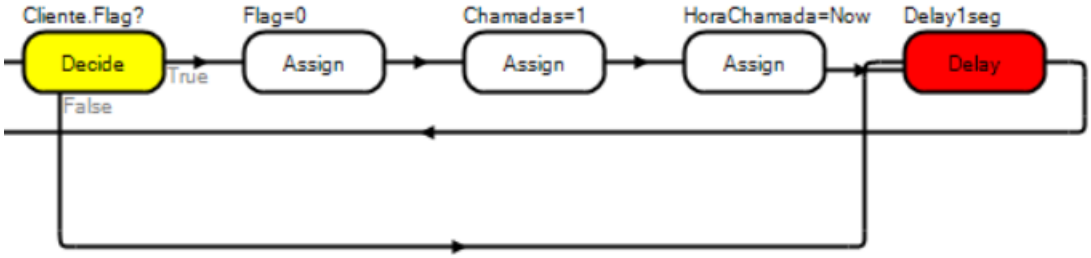


Figura 26 – Pormenor do processo "Chama", na simulação com a entidade

A chamada é despoletada colocando o seu valor igual a 1, através de um passo *assign* denominado Chamadas=1. As chamadas são tratadas por um vetor, com o número de posições igual ao número de andares, e o andar respetivo é apontado através da variável Cliente.origem. Em seguida, é gravado o instante temporal em que a chamada ocorreu, através de um *assign*. Este valor permitirá verificar qual das chamadas do sistema ocorreu há mais tempo, presente na tomada de decisão do elevador.

A fim de evitar a reescrita do instante temporal em que a chamada foi registada – inutilizando os dados – foi adicionada a variável Flag. Esta é, por defeito, igual a 1, sendo colocada a 0 no ramo em análise. Isto fará com que, após este ramo ser percorrido por um *token*, este não o irá percorrer novamente, mantendo a integridade dos dados.

No final, para evitar o sobre carregamento da simulação, é adicionado um *delay* de um segundo.

Retornando ao passo *decide* ElevadorEstaAqui?, em caso de resposta afirmativa (*true*), o cliente será transferido para o interior do elevador.

Estando no interior do veículo, a chamada é desligada – colocando-a a zero – e a hora de chamada é colocada a infinito, de forma a não influenciar a decisão do elevador no futuro. Através do passo *wait*, o *token* irá aguardar pelo disparo do evento "*GetOut*", que será disparado pelo elevador em todos os andares. Para filtrar o andar pretendido pelo cliente do andar onde o elevador parou, é adicionado um passo *decide*, que irá verificar o estado da variável do cliente VouSair. Esta variável será manipulada e o evento "*GetOut*" será disparado no processo "VerificarDestinos".

O processo "VerificarDestinos" é disparado pelo elevador sempre que para num *node*, com o intuito de percorrer todos os clientes dentro da fila do elevador e dar-lhes a ordem de saída. Como mencionado anteriormente, a ordem de saída e dada quer pela variável do cliente VouSair, quer pelo disparo do evento "*GetOut*". O processo "VerificarDestinos" é representado abaixo.

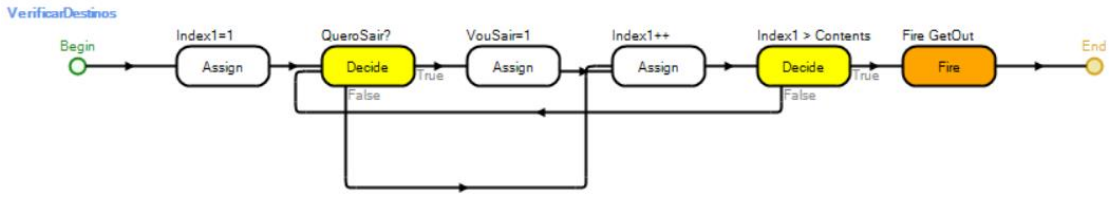


Figura 27 – Processo "VerificarDestinos", na simulação com o objeto entidade

Para tal, este processo utiliza a variável Index1 que percorrerá todos os clientes da fila do elevador. Em seguida, através de um *decide*, é feito um teste lógico que verificará se o cliente da fila no índice Index1 tem como destino o *node* atual. Se sim (*true*), a variável VouSair desse cliente é colocada a 1; se não (*false*), é ignorada a mudança da variável e é creditada uma unidade ao índice Index1. Para determinar o fim do *loop*, é colocado um *decide* que comparará o índice atual com a quantidade de clientes no interior do elevador. Caso os clientes não tenham sido completamente percorridos, o *loop* continuará; caso todos os clientes tenham sido testados, é disparado o evento "*GetOut*" através do passo *fire*, permitindo ao cliente sair do elevador no processo "Chama".

Quando ambas as condições são cumpridas: a variável "VouSair" é igual a 1 e o evento "*GetOut*" foi disparado, o processo "Chama" prosseguirá; o que significa que o cliente deverá sair do elevador. Um passo *transfer* irá transferir o cliente para o *node out* correspondente, recorrente também à tabela apresentada na Figura 24. Em seguida, é atribuída velocidade ao cliente através dos parâmetros *Movement.Rate* e *DesiredSpeed*, de forma a permitir ao cliente percorrer o *path* final, encaminhando-o para o *sink*.

2.5.3 Processos elevador

No subcapítulo 2.5.1 Enquadramento foram analisados pormenores dos dois processos que a entidade elevador executa, confrontando-os com o algoritmo básico de um elevador. Neste subcapítulo serão analisados os restantes passos.

O processo "*OnRunInitialized*", visível no Anexo I – Processo "*OnRunInitialized*" do objeto entidade, é iniciado assim que a simulação inicia. Este tem como intuito criar o objeto elevador, posicioná-lo e gerir a tomada de decisão do elevador em relação ao próximo destino a rumar. Este processo é um *loop* infinito.

Os primeiros dois passos do processo são *create* e *transfer*, que irão, respetivamente, criar o objeto elevador e posicioná-lo no andar um, o *node* inicial.

Em seguida, é colocado um passo *find* que procurará por chamadas no sistema. Caso estas não existam (*not found*), o *token* seguirá para um ramo que visa entregar os clientes que se encontram no interior do elevador. Este ramo é visível abaixo, na Figura 28.

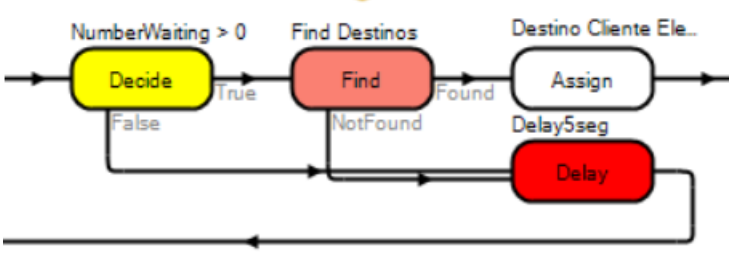


Figura 28 – Pormenor do processo "*OnRunInitialized*", na simulação com a entidade

O primeiro passo consiste num *decide* que prevê a possibilidade da não de passageiros, para além de não existirem chamadas. Para tal, este executa um teste lógico para verificar se a fila do elevador contém clientes. Caso não contenha (*false*), o *token* é levado para um *delay* que está ligado ao *find* inicial, que procura chamadas no sistema. Caso a fila do elevador contenha clientes, o *token* seguirá para um *find* que irá encontrar o destino mais próximo, percorrendo as posições da fila do elevador. O *assign* que se segue atribuirá a posição do cliente com o destino mais próximo na fila do elevador à variável utilizada no *set node* do elevador.

Retomando ao ramo superior, caso o resultado do primeiro *find* seja *found*, traduzindo-se na existência de chamadas, o *token* é levado para outro *find*. Este irá procurar pelo instante mais baixo em que foi realizada uma chamada, isto é, a chamada que ocorreu mais cedo e, portante, que contém o cliente que está há mais tempo à espera. Este valor foi gravado no processo "Chama".

A fim de evitar movimentações desnecessárias e incorreções foi adicionada uma proteção. Esta, através de um *decide*, verifica se o *node* obtido anteriormente é diferente do atual. Caso não o seja (*false*), isto é, seja igual, o *token* segue para um *delay* e reinicia o processo no *find* de chamadas. Caso seja (*true*), o *token* seguirá o processo.

Em seguida, o processo chega a uma bifurcação. Esta é visível e foi analisada na Página 41.

Depois da decisão do sentido a tomar e do *node* de destino nesse sentido, é utilizado o passo *set node* para definir o *node* de destino que o elevador irá tomar. É também a este *set node* que o *token* do ramo inferior irá desaguar. Em seguida, é atribuída velocidade ao elevador através dos parâmetros *Movement.Rate* e *DesiredSpeed*. No final do processo *loop*, o *token* irá aguardar pelo evento "IniciarMarcha" – através de um passo *wait* – que será disparado no final do processo em análise seguidamente.

O processo "Cheguei" – visível no Anexo II – Processo "Cheguei" do objeto entidade – irá ser executado à entrada de cada *node* que o elevador percorra, a fim de tomar a decisão de parar no *node*. Caso pare, é este processo que irá dar ordem para a execução do processo "VerificarDestinos", fazendo sair os clientes da fila do elevador; e permite aos clientes entrarem no elevador.

O primeiro passo deste processo é um *assign*, atribuindo à variável IDNodeAtual o valor do *node* atual, necessário para a confirmação dos destinos dos clientes na fila do elevador no ramo inferior deste processo; e para a decisão do sentido a tomar no processo "*OnRunInitialized*".

Em seguida o *token* segue para um *decide* que testa se o elevador está no *node* que lhe foi atribuído no *set node* do processo "*OnRunInitialized*". Se coincidir, significará que existem chamadas, e o elevador será parado (ramo superior). Caso não sejam coincidentes, é necessário percorrer todos os clientes na fila do elevador para confirmar se algum deles possui o andar atual como destino (ramo inferior). Este ponto foi analisado anteriormente, no subcapítulo 2.5.1 Enquadramento, na Página 42, e visível na Figura 22.

Caso não existam chamadas, nem nenhum dos clientes tenha como destino o *node* atual, o processo terminará, permitindo ao elevador seguir para o *node* seguinte. Caso hajam chamadas ou destinos, o elevador é parado através da colocação dos parâmetros *Movement.Rate* e *DesiredSpeed* a zero.

Estando o elevador parado, o processo irá verificar se existem clientes no interior do veículo. Caso existam, este irá executar o processo "VerificarDestinos". Caso não hajam, o *token* irá ignorar este passo e realizar um *delay*. Este *delay* permite não só a entrada de passageiros, mas também a saída dos clientes na fila do elevador.

A entrada de passageiros irá ocorrer de forma independente deste processo, pois é realizada no processo "Chama".

No término do processo "Cheguei", é disparado o evento "IniciarMarcha", que permitirá ao elevador recalcular o destino.

2.5.4 Conclusão

A complexidade do modelo é evidenciada pelo tamanho dos processos, a interoperacionalidade dos processos e pela necessidade de tabelas de correspondência.

O processo "*OnRunInitialized*" é um exemplo de complexidade, com vários passos *find* a procurarem a melhor solução para o elevador e vários ramos para abranger todas as possibilidades na simulação. Os processos são interoperacionais, na medida em que um processo pode executar outro (passo *execute*), e um processo poderá ficar a aguardar pelo disparo de um evento por parte de outro processo (passo

wait a aguardar por um *fire*). A tabela de correspondência, visível na Figura 24, reflete a complexidade das expressões utilizadas, uma vez que a comparação não é possível de forma direta, necessitando desta tabela para realizar a conversão.

Mas, o modelo é forte, respondendo a todas as solicitações de clientes, quer dentro, quer fora do elevador. Abaixo, na Figura 29, é visível um exemplo da simulação a decorrer, onde são visíveis clientes a entrarem e a saírem do elevador no andar 2.

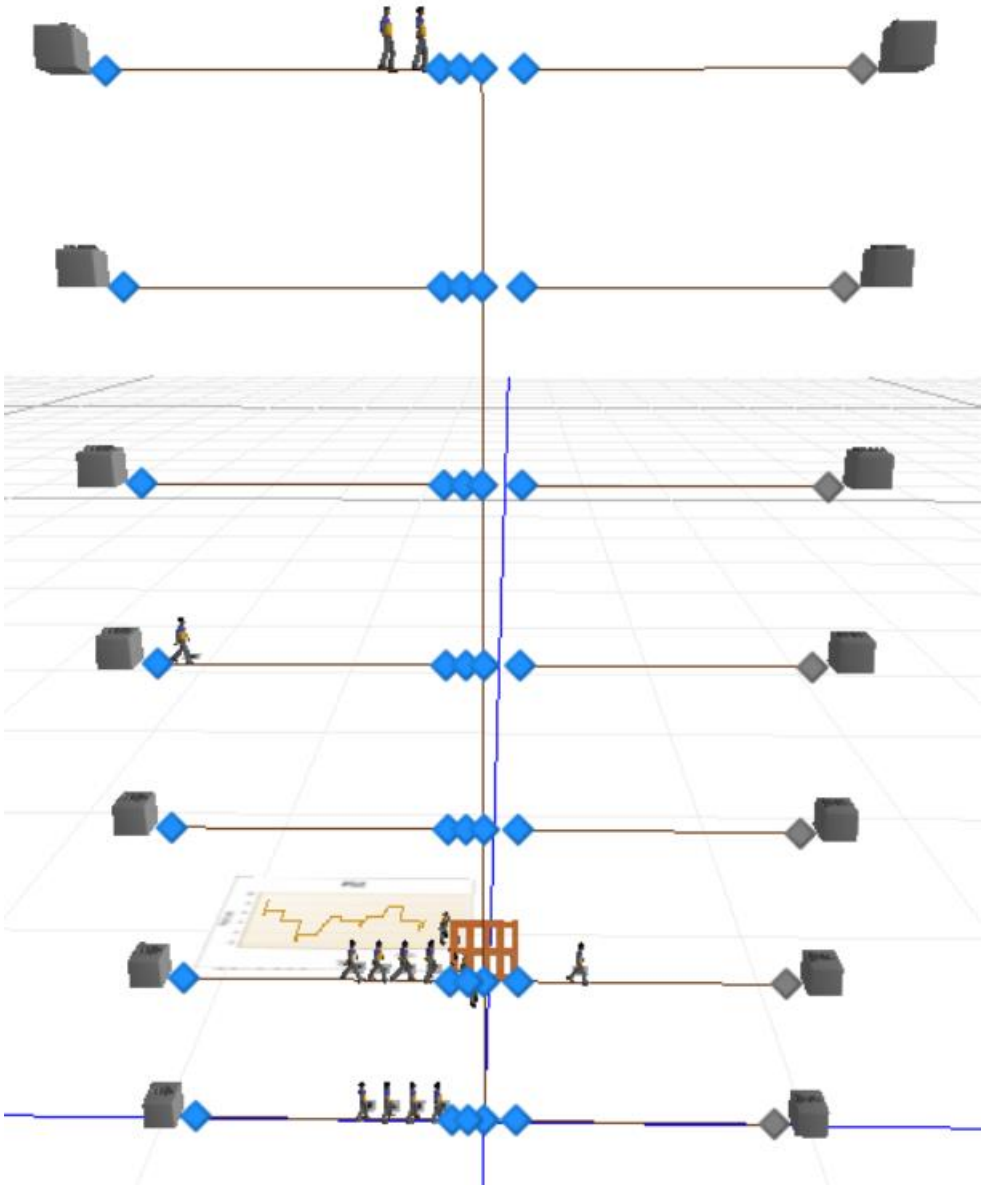


Figura 29 – Modelo de simulação do elevador, com o objeto entidade, a decorrer

2.6 Veículo vs. Entidade prática

A abordagem ao elevador também foi provada como sendo a da utilização de uma entidade, uma vez que o modelo de simulação com o veículo apresenta limitações. A mais evidente é a na não paragem

para rececionar clientes que requisitaram o veículo, dando prioridade ao destino das entidades dentro do veículo. As únicas situações em que o veículo receciona clientes são: paragem para deixar sair um cliente do interior do veículo e caso o veículo esteja vazio, respondendo de forma prioritária às requisições mais próximas.

O modelo de simulação com o objeto entidade como elevador é superior e torna-se evidente, pois as fragilidades do veículo são demasiadas, não permitindo simular um elevador no seu algoritmo base. Num algoritmo mais avançado, todas as fragilidades seriam aumentadas.

O facto de os seus processos estarem ocultos é uma fraqueza, uma vez que não é permitida a manipulação dos mesmo, consequente adaptação e acesso a algumas variáveis importantes, como é a verificação dos candidatos.

3. ANÁLISE

Propõem-se analisar o impacte que a variação do tempo de abertura de portas do elevador tem no tempo total, ou de atravessamento, dos clientes no sistema. O tempo total é o resultado da soma do tempo de espera com o tempo de viagem. Isto é, o resultado da soma do tempo que o cliente está à espera que um elevador pare no seu andar de origem e o deixe entrar, e o tempo que o cliente levará até chegar ao seu destino.

A variação do tempo de abertura de portas é particularmente interessante, uma vez que o tempo de abertura de portas tem impacte em todos os clientes do elevador. Isto porque, um aumento de abertura de portas permite uma entrada superior de passageiros por cada paragem – devido ao aumento da janela temporal – mas, por contraste, tem como consequência o aumento do tempo de travessia entre andares, uma vez que o elevador estará mais tempo parado num determinado andar.

No modelo de elevador utilizando uma entidade, este tempo é replicado no processo "Cheguei", uma vez que é este o processo despoletado na chegada do elevador a cada *node* e eventual paragem do elevado no andar. Este processo poderá ser visto no Anexo II – Processo "Cheguei" do objeto entidade. Abaixo, na Figura 30, é possível ver o pormenor do processo onde é incorporado este tempo, através de um *delay*.

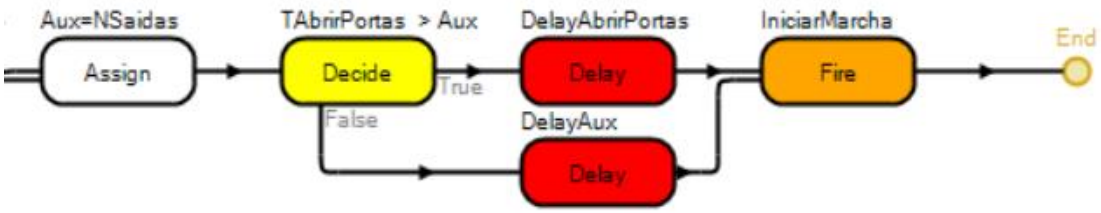


Figura 30 – Pormenor do *delay* responsável pelo tempo de abertura de portas no processo "Cheguei"

Este *delay* está ramificado de forma a reproduzir uma de duas situações: ou o tempo de abertura de portas definido, ou o tempo de saída dos passageiros; prevalecendo o de maior magnitude. Desta forma, é possível adicionar uma penalização realista para os tempos de abertura de portas reduzidos, onde, embora o tempo de abertura de portas seja pequeno, o elevador é obrigado a permitir a saída dos passageiros. Para tal, é calculada uma variável auxiliar, apelidada de Aux, que irá resultar da multiplicação do número de clientes que irá sair do elevador – calculado no processo "VerificarDestinos" – por 0,6 segundos. De seguida, é utilizado um passo *decide* para comparar ambos os tempos: o *token* seguirá o ramo *true* caso o tempo de abertura de portas seja superior à variável Aux, ou o ramo *false* caso a penalização pela saída de clientes seja superior ao tempo de abertura de

portas. Em cada ramo é executado um *delay*, resultante dos valores acima mencionados, que será responsável pela permanência do elevador num determinado andar. Após o período de espera, o *token* seguirá para o passo *fire*, onde será disparado o evento "Iniciar Marcha".

3.1 Preparação

Para recolher os dados necessários à análise proposta, é necessário adaptar o modelo. Esta adaptação consiste na adição de passos *assigns*, responsáveis pelo incremento ou decremento de variáveis, e *tallies*, responsáveis pelo registo dos valores pretendidos.

De forma a aferir o tempo total, ou de atravessamento, é necessário medir o tempo decorrido entre o instante da criação do cliente e o instante de chegada do cliente ao seu destino, individualmente.

A decisão de registar o instante da criação do cliente, ao invés do instante da chamada, advém da própria criação do modelo, uma vez que apenas o cliente na frente é que realiza a chamada do elevador, e os restantes clientes aguardam a sua vez no *path*, visto que todos os *path* de entrada não permitem ultrapassagem, não só para permitir uma melhor visualização da simulação – tendo visualmente presente a quantidade de clientes à espera do elevador – mas também porque iria causar dificuldades na análise do modelo, visto que todos os clientes seriam transferidos para a fila do elevador de uma forma quase instantânea, dando origem a erros nos passos *transfer* em situações limite, como seria o elevador estar cheio; dificultando a medição de outras medidas de desempenho.

Esta diferença entre o registo do instante de criação e a real chamada será transversal a todos os andares e cenários possíveis, atenuando as diferenças.

Para aferir este instante de criação foi criado o processo "*TimeStamp"*. Este processo é executado por todos os processos "*Output_Source_Entered*" de cada *source* através de um passo *execute* no final desse processo, ou seja, é realizado por todos os clientes aquando da sua criação. Na Figura 31, abaixo, é visível a adição do passo *execute* ao processo "*Output_Source1_Entered1*", que despoletará o processo "*TimeStamp*". Este processo inicia com um passo *assign* que irá atribuir à variável "InstanteCriacao" o instante atual através da instrução "*TimeNow*".

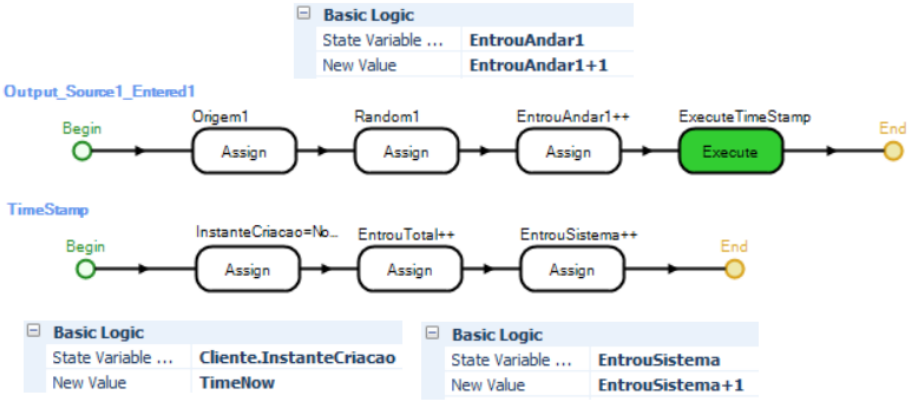


Figura 31 – Alterações ao processo "*Output_Source1_Entered1*", de forma a incluir a execução do processo "*TimeStamp*"

No processo "*Output_Source_Entered"* de cada andar foi ainda adicionado um passo *assign* que soma um valor à variável "EsperaAndar" do respetivo andar, permitindo contabilizar o número de pessoas em espera de forma individual. No processo "*TimeStamp*" foram ainda adicionados dois *assigns*, um para incrementar a variável "EntrouTotal", que medirá o número de clientes criados – permitindo obter as métricas: média de clientes por hora e total de clientes que entraram no sistema – e outro para incrementar a variável "EntrouSistema", responsável por aferir o número de clientes que estão no interior do sistema.

De forma a registar cada observação do tempo total, foi adicionado um passo *tally* ao processo "Chama". Este *tally* é colocado neste processo, depois do segundo passo *transfer*, pois é nesse exato momento que o cliente sai da fila do elevador em direção ao *node* do andar que possui como destino e respetivo *sink*, momento em que o tempo de viagem termina. O *tally* possui uma expressão que subtrai ao instante atual ("*TimeNow*") o valor anteriormente gravado na variável "InstanteCriacao", obtendo o tempo decorrido entre a criação do cliente e a sua entrega no andar de destino, isto é, o tempo de atravessamento do sistema, que é referido ao longo da presente Dissertação como tempo total. Abaixo, na Figura 32, é possível verificar a adição do passo *tally* ao processo "Chama".

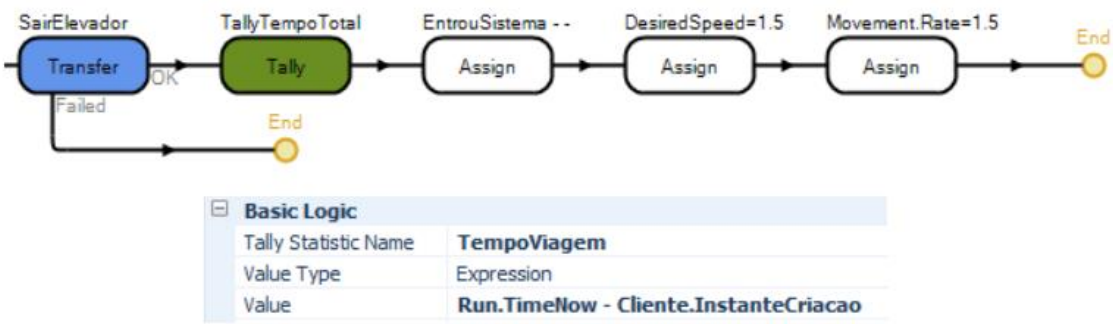
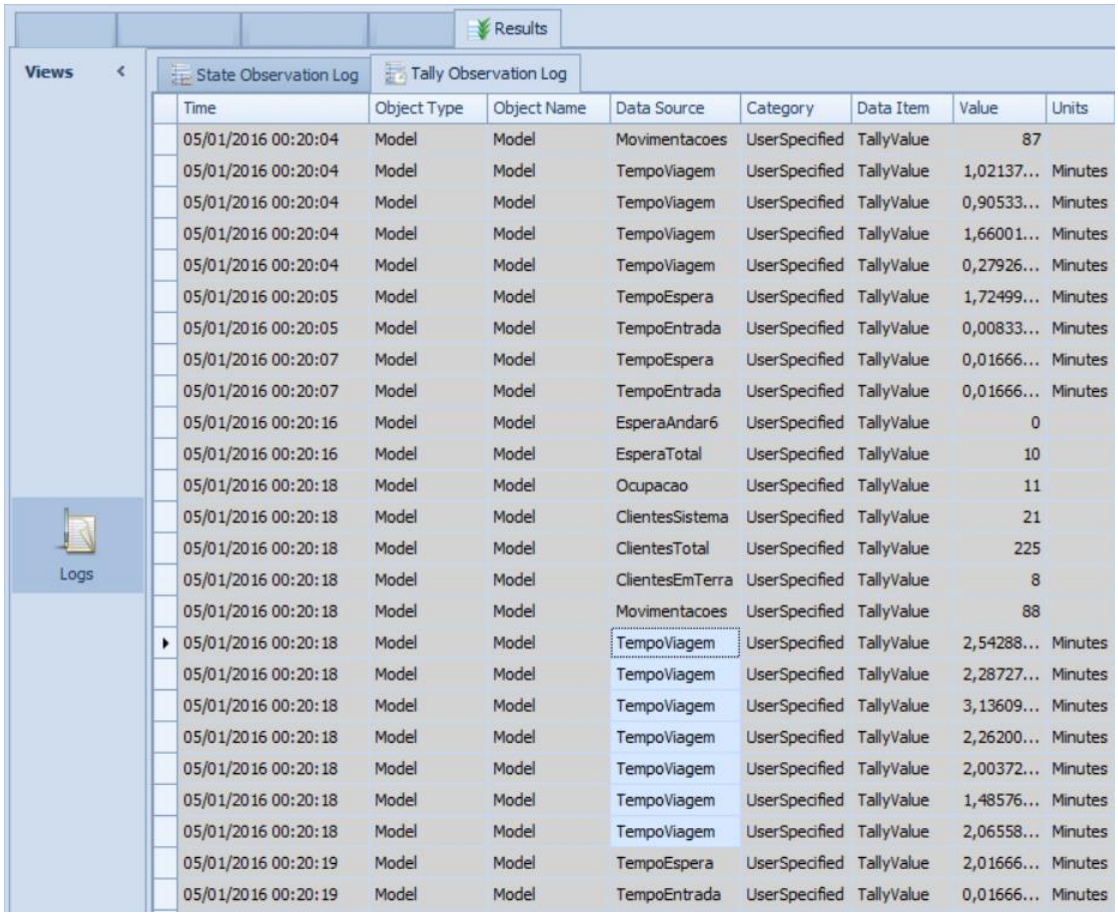


Figura 32 – Pormenor do processo "Chama", onde foi adicionado o passo *tally*

É nesse instante, após o cliente ter saído do elevador, que é decrementada em uma unidade a variável "EntrouSistema", uma vez que o cliente foi servido pelo sistema e encontra-se no seu destino.

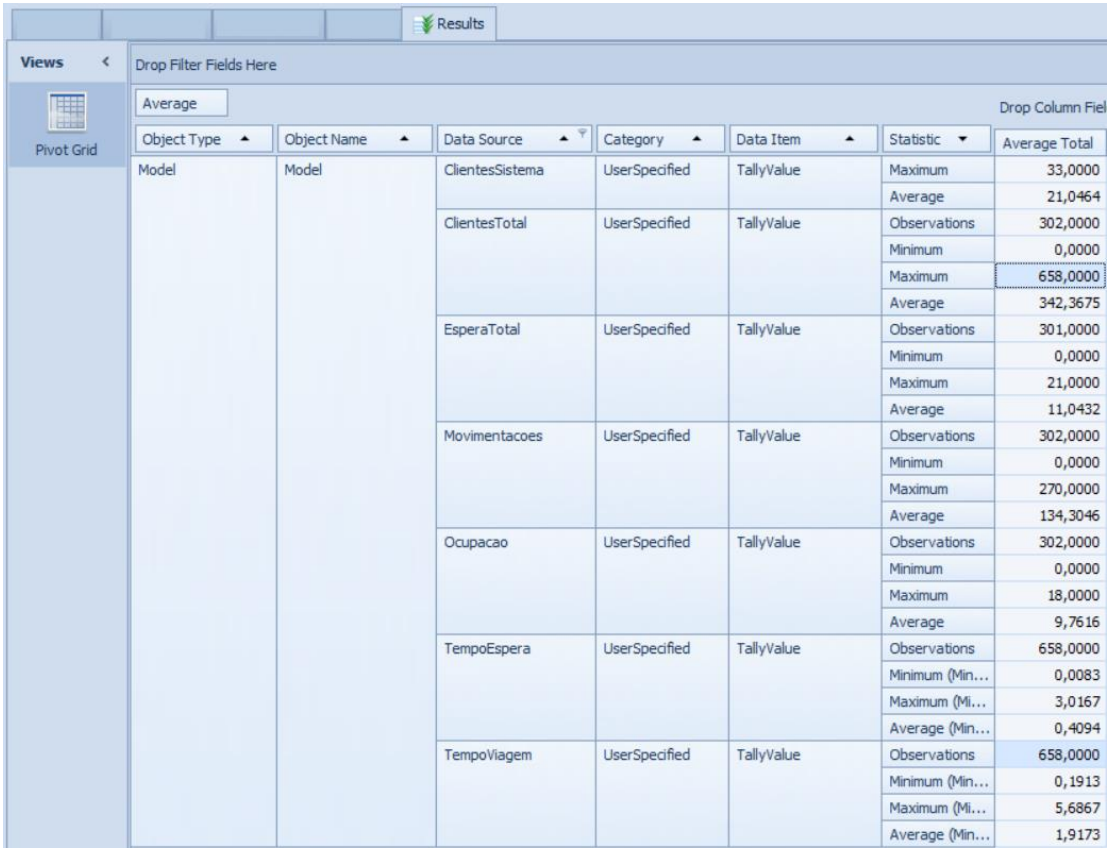
O *tally* observa o valor da subtração e regista-o no elemento "TempoViagem", um tipo especial de variável do *software*, uma vez que é responsável pelo registo das observações. Todas as observações a esta variável podem ser consultadas nos *logs* do separador *Results* do *software*. Abaixo, na Figura 33, é possível visualizar um exemplo ilustrativo dos registos das observações dos vários *tallies* adicionados ao modelo, no *software*.



Time	Object Type	Object Name	Data Source	Category	Data Item	Value	Units
05/01/2016 00:20:04	Model	Model	Movimentacoes	UserSpecified	TallyValue	87	
05/01/2016 00:20:04	Model	Model	TempoViagem	UserSpecified	TallyValue	1,02137...	Minutes
05/01/2016 00:20:04	Model	Model	TempoViagem	UserSpecified	TallyValue	0,90533...	Minutes
05/01/2016 00:20:04	Model	Model	TempoViagem	UserSpecified	TallyValue	1,66001...	Minutes
05/01/2016 00:20:04	Model	Model	TempoViagem	UserSpecified	TallyValue	0,27926...	Minutes
05/01/2016 00:20:05	Model	Model	TempoEspera	UserSpecified	TallyValue	1,72499...	Minutes
05/01/2016 00:20:05	Model	Model	TempoEntrada	UserSpecified	TallyValue	0,00833...	Minutes
05/01/2016 00:20:07	Model	Model	TempoEspera	UserSpecified	TallyValue	0,01666...	Minutes
05/01/2016 00:20:07	Model	Model	TempoEntrada	UserSpecified	TallyValue	0,01666...	Minutes
05/01/2016 00:20:16	Model	Model	EsperaAndar6	UserSpecified	TallyValue	0	
05/01/2016 00:20:16	Model	Model	EsperaTotal	UserSpecified	TallyValue	10	
05/01/2016 00:20:18	Model	Model	Ocupacao	UserSpecified	TallyValue	11	
05/01/2016 00:20:18	Model	Model	ClientesSistema	UserSpecified	TallyValue	21	
05/01/2016 00:20:18	Model	Model	ClientesTotal	UserSpecified	TallyValue	225	
05/01/2016 00:20:18	Model	Model	ClientesEmTerra	UserSpecified	TallyValue	8	
05/01/2016 00:20:18	Model	Model	Movimentacoes	UserSpecified	TallyValue	88	
05/01/2016 00:20:18	Model	Model	TempoViagem	UserSpecified	TallyValue	2,54288...	Minutes
05/01/2016 00:20:18	Model	Model	TempoViagem	UserSpecified	TallyValue	2,28727...	Minutes
05/01/2016 00:20:18	Model	Model	TempoViagem	UserSpecified	TallyValue	3,13609...	Minutes
05/01/2016 00:20:18	Model	Model	TempoViagem	UserSpecified	TallyValue	2,26200...	Minutes
05/01/2016 00:20:18	Model	Model	TempoViagem	UserSpecified	TallyValue	2,00372...	Minutes
05/01/2016 00:20:18	Model	Model	TempoViagem	UserSpecified	TallyValue	1,48576...	Minutes
05/01/2016 00:20:18	Model	Model	TempoViagem	UserSpecified	TallyValue	2,06558...	Minutes
05/01/2016 00:20:19	Model	Model	TempoEspera	UserSpecified	TallyValue	2,01666...	Minutes
05/01/2016 00:20:19	Model	Model	TempoEntrada	UserSpecified	TallyValue	0,01666...	Minutes

Figura 33 – Exemplo dos registos (*logs*) dos vários *tallies* adicionados ao modelo

Estes dados podem ser exportados para um ficheiro com a extensão CSV – que poderá ser aberto com ferramentas de análise de dados, como é o Microsoft Excel® – ou utilizando as próprias ferramentas de *report* e de análise do *software*. Abaixo, na Figura 34, encontra-se um exemplo ilustrativo de uma análise sumária dos dados obtidos.



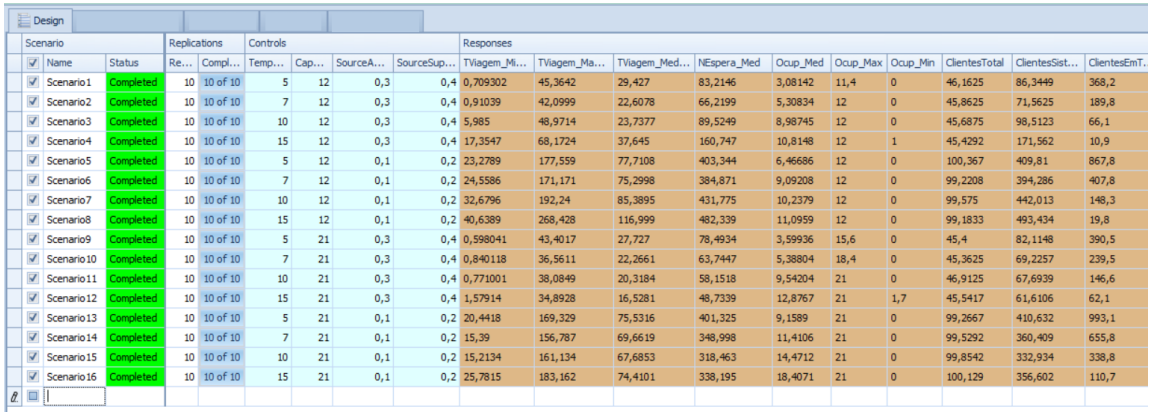
Object Type	Object Name	Data Source	Category	Data Item	Statistic	Average Total
Model	Model	ClientesSistema	UserSpecified	TallyValue	Maximum	33,0000
					Average	21,0464
		ClientesTotal	UserSpecified	TallyValue	Observations	302,0000
					Minimum	0,0000
					Maximum	658,0000
					Average	342,3675
		EsperaTotal	UserSpecified	TallyValue	Observations	301,0000
					Minimum	0,0000
					Maximum	21,0000
					Average	11,0432
		Movimentacoes	UserSpecified	TallyValue	Observations	302,0000
					Minimum	0,0000
					Maximum	270,0000
					Average	134,3046
		Ocupacao	UserSpecified	TallyValue	Observations	302,0000
					Minimum	0,0000
					Maximum	18,0000
					Average	9,7616
		TempoEspera	UserSpecified	TallyValue	Observations	658,0000
					Minimum (Min...	0,0083
					Maximum (Mi...	3,0167
					Average (Min...	0,4094
		TempoViagem	UserSpecified	TallyValue	Observations	658,0000
					Minimum (Min...	0,1913
					Maximum (Mi...	5,6867
					Average (Min...	1,9173

Figura 34 – Exemplo da ferramenta de análise de dados do *software* Simio®

Os valores visíveis na Figura 33 e Figura 34 são meramente ilustrativos, resultantes de uma simulação teste realizada com o propósito de exibir as e funcionalidades e potencialidade do *software*.

Para a análise presente no subcapítulo seguinte, recorreu-se à funcionalidade Experiências do *software* Simio®. Esta ferramenta é especialmente importante, pois permite produzir vários testes (apelidados de cenários) através da variação das propriedades definidas no modelo, como: o tempo de abertura de portas, a capacidade do elevador e o volume de criação de clientes nos *sources*. Esta ferramenta permite também definir o número de replicações a realizar para cada cenário e obter as várias observações dos *tallies* inseridos no modelo, podendo obter a média, o máximo ou o mínimo registado pelos *tallies* de cada cenário, calculando a média das replicações.

Abaixo, na Figura 35, é possível visualizar um exemplo de uma experiência com diferentes tempos de abertura de portas, diferentes capacidades do elevador e diferentes exponenciais de criação de clientes no *source* do andar um e nos *sources* dos andares superiores.



Design

Scenario		Replications		Controls				Responses								
Name	Status	Re...	Compl...	Temp...	Cap...	SourceA...	SourceSup...	TViagem_Mi...	TViagem_Ma...	TViagem_Med...	NEspera_Med	Ocup_Med	Ocup_Max	Ocup_Min	ClientesTotal	ClientesSist...	ClientesEmT.
Scenario1	Completed	10	10 of 10	5	12	0,3	0,4	0,709302	45,3642	29,427	83,2146	3,08142	11,4	0	46,1625	86,3449	368,2
Scenario2	Completed	10	10 of 10	7	12	0,3	0,4	0,91039	42,0999	22,6078	66,2199	5,30834	12	0	45,8625	71,5625	189,8
Scenario3	Completed	10	10 of 10	10	12	0,3	0,4	5,985	48,9714	23,7377	89,5249	8,98745	12	0	45,6875	98,5123	66,1
Scenario4	Completed	10	10 of 10	15	12	0,3	0,4	17,3547	68,1724	37,645	160,747	10,8148	12	1	45,4292	171,562	10,9
Scenario5	Completed	10	10 of 10	5	12	0,1	0,2	23,2789	177,559	77,7108	403,344	6,46686	12	0	100,367	409,81	867,8
Scenario6	Completed	10	10 of 10	7	12	0,1	0,2	24,5586	171,171	75,2998	384,871	9,09208	12	0	99,2208	394,286	407,8
Scenario7	Completed	10	10 of 10	10	12	0,1	0,2	32,6796	192,24	85,3895	431,775	10,2379	12	0	99,575	442,013	148,3
Scenario8	Completed	10	10 of 10	15	12	0,1	0,2	40,6389	268,428	116,999	482,339	11,0959	12	0	99,1833	493,434	19,8
Scenario9	Completed	10	10 of 10	5	21	0,3	0,4	0,598041	43,4017	27,727	78,4934	3,59936	15,6	0	45,4	82,1148	390,5
Scenario10	Completed	10	10 of 10	7	21	0,3	0,4	0,840118	36,5611	22,2661	63,7447	5,38804	18,4	0	45,3625	69,2257	239,5
Scenario11	Completed	10	10 of 10	10	21	0,3	0,4	0,771001	38,0849	20,3184	58,1518	9,54204	21	0	46,9125	67,6939	146,6
Scenario12	Completed	10	10 of 10	15	21	0,3	0,4	1,57914	34,8928	16,5281	48,7339	12,8767	21	1,7	45,5417	61,6106	62,1
Scenario13	Completed	10	10 of 10	5	21	0,1	0,2	20,4418	169,329	75,5316	401,325	9,1589	21	0	99,2667	410,632	993,1
Scenario14	Completed	10	10 of 10	7	21	0,1	0,2	15,39	156,787	69,6619	348,998	11,4106	21	0	99,5292	360,409	655,8
Scenario15	Completed	10	10 of 10	10	21	0,1	0,2	15,2134	161,134	67,6853	318,463	14,4712	21	0	99,8542	332,934	338,8
Scenario16	Completed	10	10 of 10	15	21	0,1	0,2	25,7815	183,162	74,4101	338,195	18,4071	21	0	100,129	356,602	110,7

Figura 35 – Exemplo da ferramenta Experiências do software Simio®

Os dados resultantes da experiência – que constam nas colunas do lado direito da Figura 35 acima – podem ser extraídos do *software* para uma ferramenta de tratamento de dados, como, por exemplo, o Microsoft Excel®, para serem tratados.

3.2 Propriedades

De forma a obter os diferentes cenários, é necessário criar propriedades do modelo. Isto é, variáveis que possam ser controladas nos cenários executados na ferramenta Experiências, de forma a obter as observações dos vários *tallies* do modelo à medida que uma, ou mais, propriedades são alteradas.

As propriedades definidas no modelo de simulação com o objeto entidade são:

- Tempo de abertura de portas, em segundos: irá determinar o valor do *delay* do processo "Cheguei", visível na Figura 30, representando o tempo mínimo de abertura de portas de um elevador;
- Capacidade do elevador: irá definir o tamanho da fila (*ride*) do veículo entidade, isto é, o número máximo de clientes que o elevador poderá transportar;
- Criação de clientes no *source* do andar um: valor da exponencial aleatória responsável pela criação de clientes no andar um;
- Criação de clientes nos *sources* dos andares superiores: valor da exponencial aleatória responsável pela criação de clientes nos andares dois a sete;

Estes últimos dois pontos – criação de clientes nos *sources* do andar um e superiores – tem como objetivo aumentar e diminuir a criação de entidades, simulando os diferentes picos: subida, descida ou misto nas experiências, estando diretamente relacionados com a solicitação ao elevador.

O *output* do *source* do andar um irá refletir a quantidade de pessoas que pretendem entrar no edifício, utilizando o elevador para subir. Este valor será muito elevado no pico de subida – refletindo, por

exemplo, a entrada ao serviço dos funcionários de uma empresa; baixo no pico de descida – recriando o fim do horário laboral de uma empresa e consequente saída do edifício; e ligeiramente superior aos restantes antares no movimento misto – uma vez que, por se tratar do andar de entrada do edifício, haverá maior necessidade de o utilizar, quer pelos próprios funcionários de uma empresa, quer por terceiros.

O *output* dos *sources* dos andares superiores replicará os clientes que se encontram nos andares superiores ao andar um: do andar dois ao sete. Este valor será mais elevado nos picos de descida – refletindo a saída do edifício no final de um dia de trabalho; baixo nos picos de subida – simulando o momento de entrada no edifício dos funcionários de uma empresa; e ligeiramente inferior quando comparado com o andar um no movimento misto – representando as movimentações no interior do edifício, mas tendo em conta que o andar um é o andar de entrada e saída do edifício.

Estes dois parâmetros são manipuláveis nos *sources* dos respetivos andares, variando o coeficiente da exponencial aleatória que regula a criação de novos clientes. Na Figura 36, abaixo, é visível este parâmetro nos menus do *software*.

Figura 36 – Definição do parâmetro de criação de clientes nos *sources*

A propriedade tempo de abertura de portas, denominada no software por "TempoAbrirPortas" representa o tempo mínimo de abertura de portas do elevador, em segundos. Este tempo de abertura poderá ser superior devido ao tempo de saída de passageiros. O valor desta propriedade irá influenciar o *delay* do processo "Cheguei" – analisado na Página 51 e visível no pormenor da Figura 30 – fazendo com o que *token* deste processo pare e só siga, para disparar o evento "IniciarMarcha", findo o tempo atribuído ao respetivo *delay*. Este tempo tem impacte direto não só na receção de passageiros, uma vez que um menor tempo de abertura de portas poderá deixar clientes à espera; mas também aumentar o tempo de paragem do elevador, aumentando o tempo de espera dos clientes nos restantes andares.

A capacidade do elevador foi adicionada às propriedades do modelo, pois permite ao modelo ser utilizado para fins de dimensionamento, analisando o impacte da colocação de uma cabina maior ou menor no desempenho do sistema. Uma cabina maior permite transportar um número superior de passageiros, diminuindo a possibilidade de ficarem clientes em terra por ter sido atingido o limite de passageiros. Por contraste, uma cabina com maior capacidade exige mais espaço – algo importante no

mercado imobiliário – e um maior consumo energético, quer quando viaja cheio, quer vazio, uma vez que o contrapeso irá influenciar o gasto energético.

3.3 Experiências

O foco central das experiências realizadas foi o tempo de abertura de portas, uma vez que esta propriedade tem influência direta na maioria das medidas de desempenho de um elevador: tempo de espera, tempo de viagem, movimentações do elevador, ocupação (ou carga) do elevador e clientes em espera.

Esta propriedade irá variar entre um e vinte segundos, retratando as várias possibilidades de abertura de portas, inclusive situações limite como é a de um e dois segundos.

Por se tratar de um sistema de apenas um elevador, optou-se por atribuir os valores de 0,5 e 0,6 às exponenciais aleatórias dos *sources* do andar um e andares superiores, respetivamente. Desta forma, os movimentos em vazio são de pequena expressão, mantém-se uma solicitação constante e evita-se o sobre carregamento do sistema, que influenciaria negativamente os dados. Por hora, o sistema recebe cerca de 30 clientes que pretendem utilizar o elevador.

A capacidade do elevador tem influência nas várias medidas de desempenho, especialmente no tempo de viagem, pois uma menor capacidade irá limitar a entrada de clientes, criando longas filas de espera, aumentando o tempo de espera em situações de elevada solicitação, prejudicando o tempo total (ou de atravessamento) do sistema. De forma a eliminar a influência negativa da capacidade do elevador no sistema – e uma vez que o modelo inclui apenas um elevador – optou-se por atribuir o valor de 21 pessoas.

A experiência realizada decorre no período de vinte e quatro horas, com uma criação de clientes contínua e estável durante esse período, com um aquecimento (*warm-up period*) de uma hora e dez replicações para cada cenário.

3.3.1 Dados

Como medidas de desempenho, foram estabelecidas as seguintes variáveis:

- Tempo total (ou de atravessamento), médio, mínimo e máximo: tempo decorrido desde a criação de um cliente até à sua entrega no andar de destino pretendido;
- Ocupação (ou carga) do elevador, média, mínima e máxima: quantidade de clientes no interior do elevador;
- Movimentações do elevador: quantidade de movimentos entre andares que o elevador realizou;

- Clientes no sistema: quantidade de clientes no interior do sistema, isto é, clientes criados que ainda não chegaram ao seu destino;
- Total de clientes em espera: o total de clientes à espera do elevador em todos os andares;
- Tempo de espera médio: tempo de espera médio pelo elevador.

No Anexo IV – Tabela resumo de dados, é visível uma tabela resumo com todos os dados recolhidos das medidas de desempenho mencionadas acima. Esta tabela apresenta uma graduação de cores de verde a vermelho, passando pelo amarelo, representando a variação do melhor para o pior valor registado. Nas colunas dos dados da ocupação – mínima, máxima e média – os valores sobrepõem-se a uma barra que representa a evolução da ocupação nos vários cenários, onde 100% representa o valor máximo registado na coluna em análise.

A variação do tempo total (ou de atravessamento) do sistema é visível na Tabela 2, abaixo.

Tabela 2 – Evolução do tempo total, com o aumento do tempo de abertura de portas

Tempo de abertura de portas [segundos]	Tempo total mínimo [minutos]	Tempo total máxima [minutos]	Tempo total médio [minutos]
1	1,2	15,9	8,4
2	1,3	21,2	11,0
3	0,9	22,8	14,8
4	0,7	23,0	14,0
5	0,7	12,8	7,3
6	0,7	10,3	5,4
7	0,6	7,1	3,6
8	0,8	8,0	3,8
9	0,8	6,7	3,6
10	0,8	6,4	3,4
11	0,8	5,5	3,0
12	1,0	6,0	3,1
13	1,1	5,8	3,2
14	0,9	6,2	3,3
15	0,9	6,3	3,3
17	0,9	8,6	4,6
20	1,0	12,0	5,8

Através da tabela acima é possível visualizar que os piores desempenhos do tempo total são registados nos tempos de abertura de portas mais baixos. No tempo total mínimo, os valores mais elevados são registados na abertura de portas durante um e dois segundos. No tempo total máximo e médio, os valores mais elevados são registados no registo com três e quatro segundos de tempo de abertura de portas. No registo mais elevado, de vinte segundos, todos os registos de tempo total voltam a subir.

Abaixo, no Gráfico 1, é possível visualizar esta evolução graficamente.

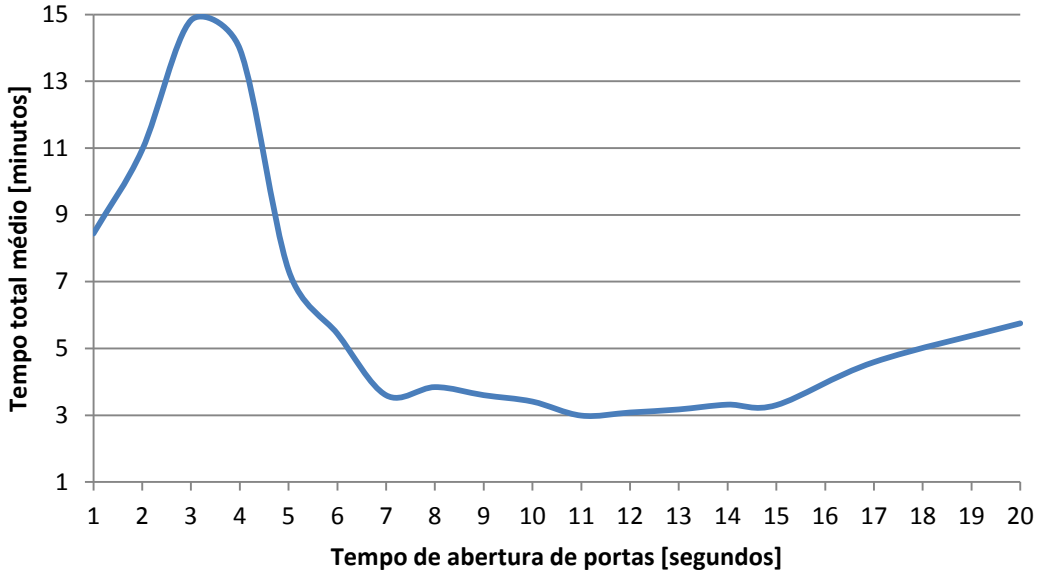


Gráfico 1 – Evolução do tempo total médio, em minutos

É possível visualizar uma faixa, entre os nove e os quinze segundos de abertura de porta, onde os registos do tempo total são mais baixos. Para aumentar a resolução, abaixo no Gráfico 2 é possível visualizar essa faixa em pormenor.

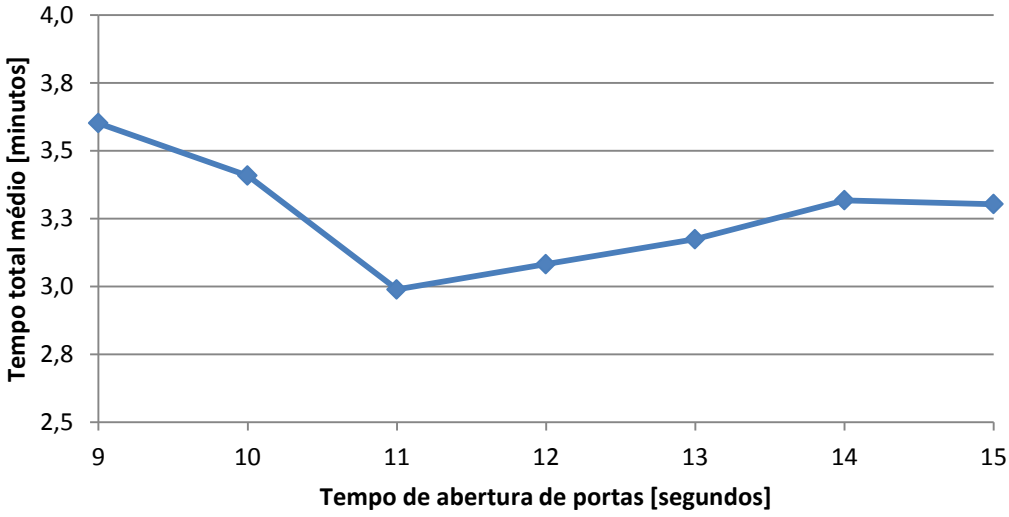


Gráfico 2 – Evolução do tempo total médio, em minutos, em pormenor

O registo mais baixo ocorre no cenário com onze segundos de tempo de abertura de portas. Este ponto coincide com o ponto onde o tempo total máximo também atinge o seu valor mais baixo, visível no Gráfico 3 abaixo, o que revela o grande impacte que o tempo total máximo tem no tempo total médio.

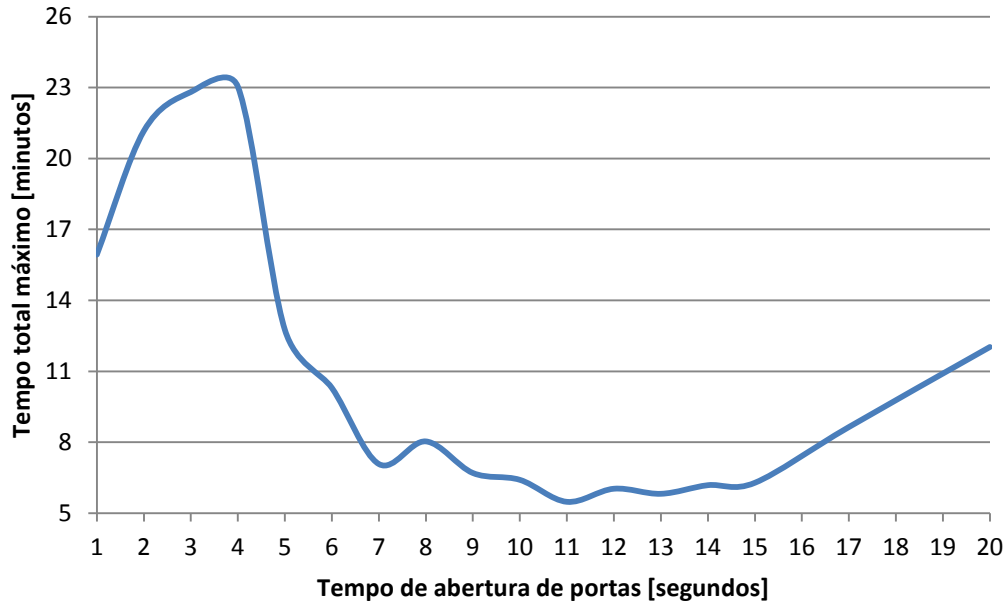


Gráfico 3 – Evolução do tempo total máximo, em minutos

O valor da variável Clientes no sistema está ligado com o tempo total, uma vez que – sendo o valor da criação de clientes idêntico – se o valor do tempo total diminui, a quantidade de clientes no interior do sistema terá que diminuir, uma vez que o elevador estará a deixar os clientes no seu destino. No Gráfico 4, abaixo, é possível confirmar isso mesmo.

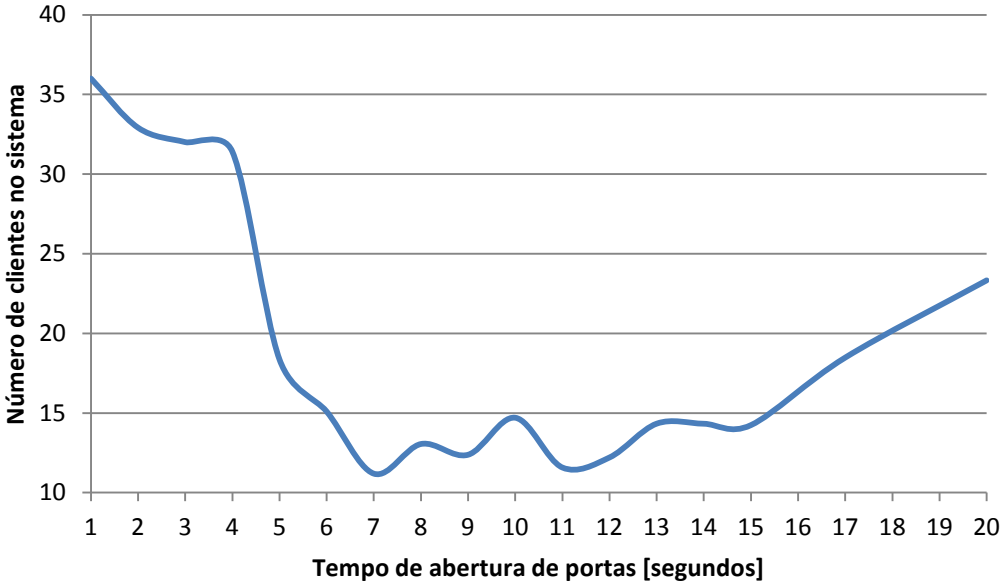


Gráfico 4 – Evolução do número de clientes no interior do sistema

O valor registado no tempo de abertura de portas de onze segundos é de 11.6, um pouco acima do valor mínimo, de 11.2, registado no tempo de abertura de portas de sete segundos. Ainda assim, a curva é muito parecida com a obtida na evolução do tempo total, mas com um declive mais acentuado nos extremos, onde o valor do tempo de abertura de portas é mínimo e máximo.

A quantidade de clientes no sistema é influenciada de forma significativa pela quantidade de clientes em espera, uma vez que a quantidade de clientes no sistema é o resultado da soma da quantidade de clientes em espera com a quantidade de clientes no interior do veículo. Portanto, a curva do Gráfico 5 abaixo tem uma forma muito parecida com a curva do Gráfico 4 acima.

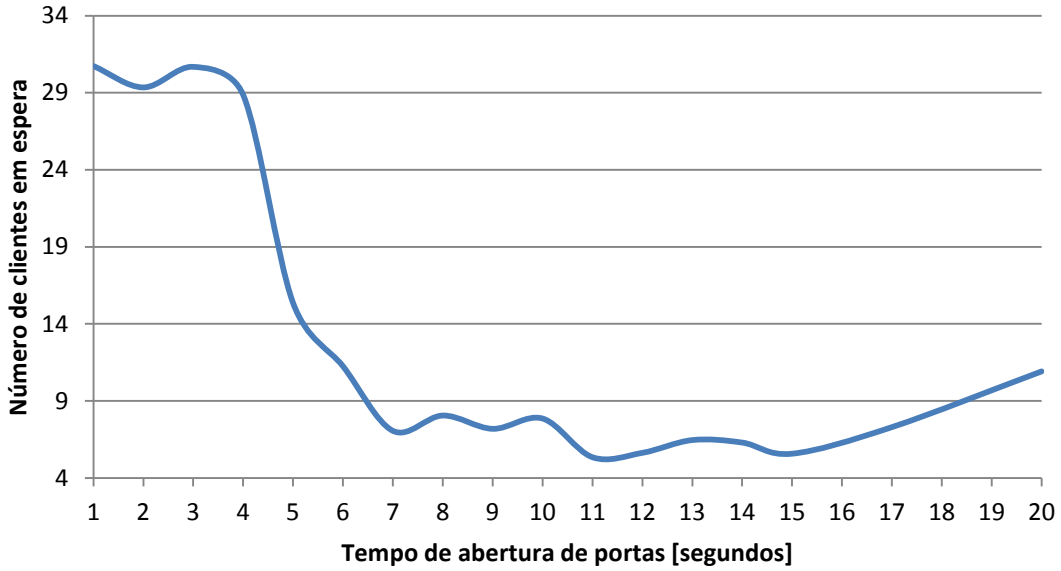


Gráfico 5 – Evolução do número de clientes em espera

A evolução do tempo de espera dos clientes é pouco conclusiva, não permitindo retirar resultados inequívocos sobre as suas causas. O Gráfico 6, abaixo, demonstra essa evolução.

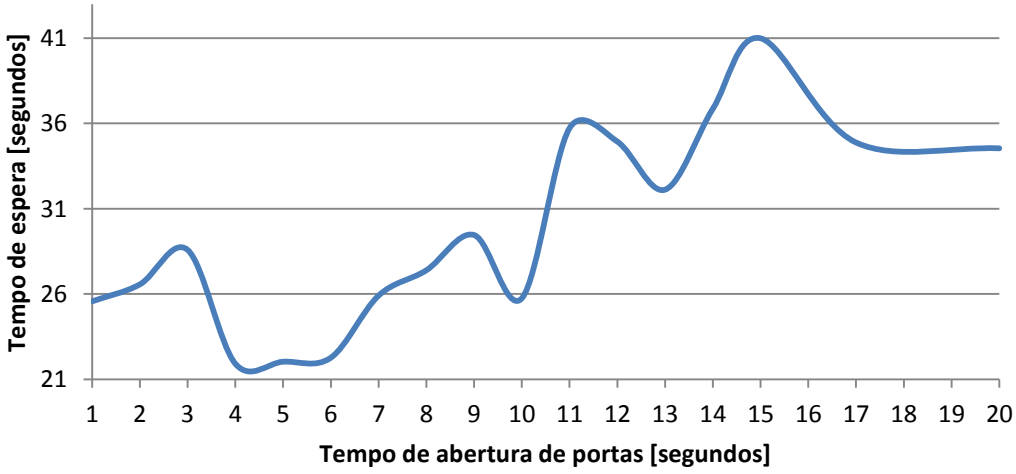


Gráfico 6 – Evolução do tempo de espera, em segundos

Para permitir melhores conclusões sobre a espera foi criado o coeficiente de espera. Este coeficiente consiste na multiplicação do número de pessoas em espera pelo tempo de espera. Desta forma, o coeficiente será elevado se um dos fatores for elevado, podendo evidenciar cenários onde, apesar de um dos fatores diminuir, o outro sobe, revelando problemas no sistema, uma vez que uma grande fila de espera ou um grande tempo de espera poderão significar um mau desempenho do sistema. O Gráfico 7 abaixo revela a evolução desse coeficiente à medida que o tempo de abertura de portas aumenta.

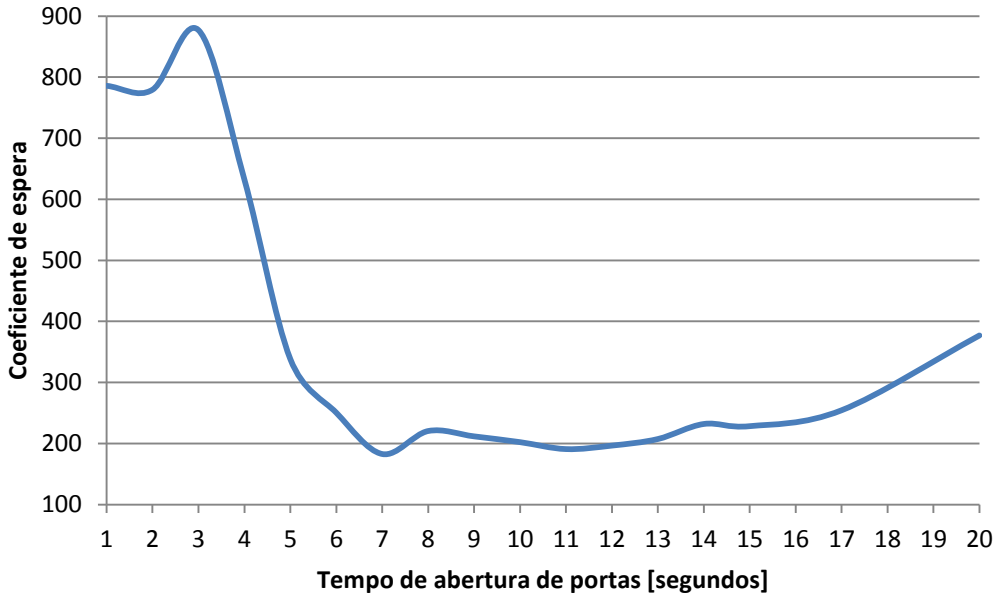


Gráfico 7 – Evolução do coeficiente de espera

É possível verificar que a faixa entre os sete e os quinze segundos registam o coeficiente mais baixo. Isto significa que é nesta faixa onde se regista o melhor desempenho do sistema em termos de espera dos clientes, sendo coincidente com a faixa onde são registados os menores valores de tempo total (ou de atravessamento) dos clientes.

Recorrendo aos gráficos anteriores, é possível verificar que nesta faixa, apesar do terem sido registados valores elevados de tempo de espera, o tempo total regista os valores mais baixos. Isto significa que o tempo de viagem – no interior do elevador – é reduzido.

A evolução da ocupação do elevador regista um decréscimo nos dois primeiros cenários, mas inicia uma escalada quase linear à medida que o tempo de abertura de portas aumenta, sendo visível a similaridade entre as três curvas: ocupação mínima, média e máxima. As três curvas são visíveis no Gráfico 8 abaixo.

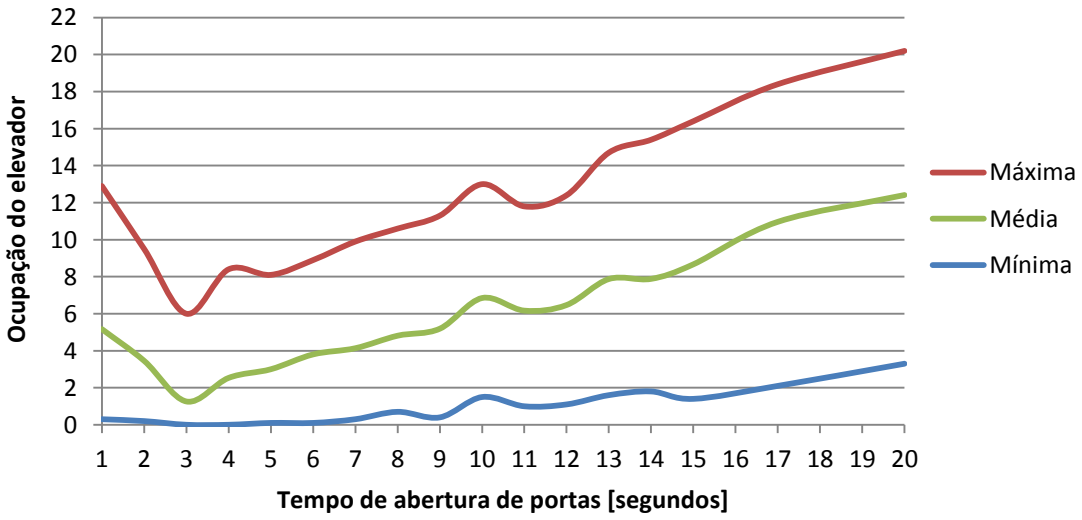


Gráfico 8 – Evolução da ocupação do elevador

Este gráfico revela a ligação da capacidade do elevador com o tempo de abertura de portas, em que um tempo elevado de abertura de portas revela a necessidade de um elevador com uma maior capacidade.

As movimentações do elevador, isto é, a quantidade de movimentos entre andares que o elevador executa, registam uma diminuição à medida que o tempo de abertura de portas aumenta, evidenciando o aumento da percentagem de tempo em que o elevador está parado, ao invés de estar em movimento a transportar clientes. Esta evolução é visível no Gráfico 9 abaixo.

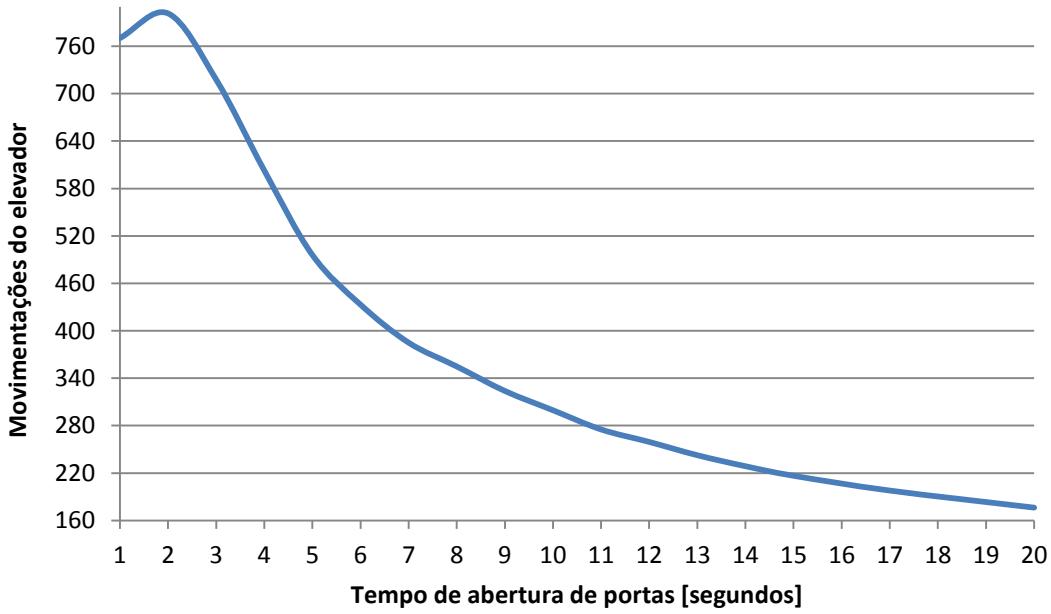


Gráfico 9 – Evolução das movimentações do elevador

O consumo energético, uma consciencialização crescente do ponto de vista ambiental, mas também uma preocupação constante devido ao seu custo, especialmente quando o serviço prestado é gratuito. Nesta avaliação, há duas medidas importantes: a ocupação do elevador – uma vez que uma ocupação muito inferior ao dimensionado ou muito próxima do limite irão exigir maior binário do motor elétrico, uma vez que a diferença entre o peso da cabina com os passageiros no seu interior será muito diferente do peso do contrapeso, e, consequentemente, um consumo superior de energia elétrica – e o número de movimentação – um maior número de movimentos corresponderá a um maior consumo de energia, especialmente tendo em conta que o instante do arranque e a aceleração até à velocidade máxima são os momentos de maior consumo de energia.

De forma a visualizar melhor a relação entre estas duas medidas, o Gráfico 10 abaixo coloca a evolução de ambas lado a lado, estando a evolução do número de movimentos.

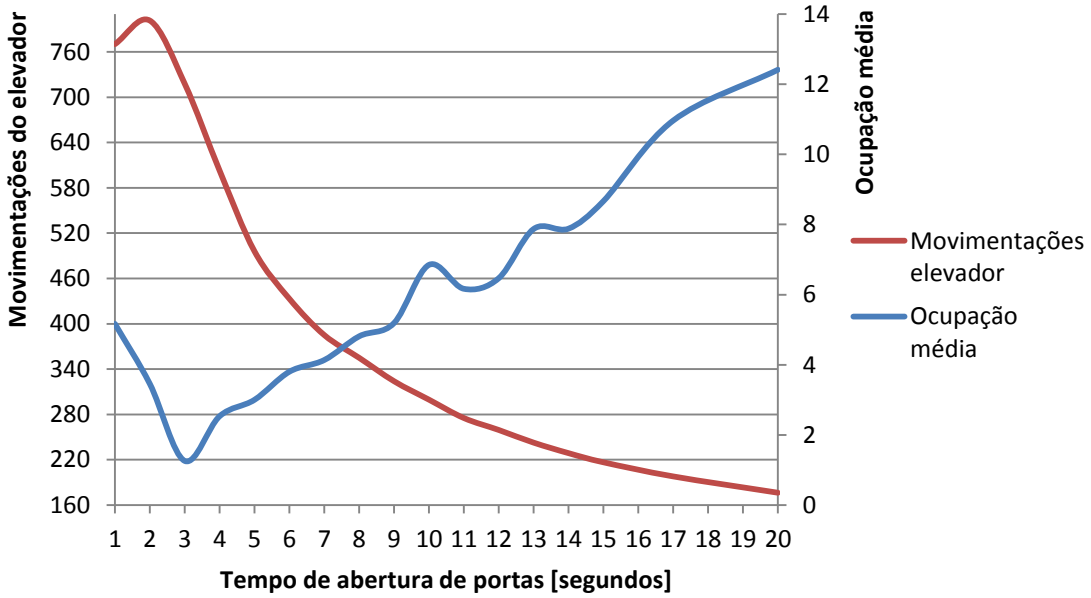


Gráfico 10 – Evolução do número de movimentações comparada com a evolução da ocupação

É possível verificar que, embora o número de movimentações diminua com o aumento do tempo de abertura de portas, a ocupação média do elevador sobe. O ponto de menor consumo energético estará no centro do gráfico acima, mas, por falta de dados – nomeadamente, os valores de consumo energético por unidade de carga e por iniciação da marcha – não é possível determinar o ponto exato.

3.3.2 Conclusão

É visível o impacte que o aumento do tempo de abertura de portas tem em todos as medidas de desempenho analisadas neste subcapítulo. É, portanto, crucial definir um valor saudável para o sistema. Isto é, um valor que permita um bom tempo de atravessamento dos clientes, mas que, ao mesmo tempo, não tenha um impacte negativo nas outras medidas, tais como: número de clientes à

espera, ocupação e movimentações do elevador; respeitando as características do sistema: número de solicitações e capacidade máxima da cabina.

Neste tipo de decisões existem dois lados: o tempo total, ou de atravessamento, dos clientes no sistema – representando o desempenho efetivo do sistema e a perceção dos utilizadores para com o sistema e a empresa a que serve – e o consumo energético – uma fatura importante na exploração deste tipo de sistemas, especialmente quando são gratuitos.

4. CONCLUSÕES

Neste capítulo serão apresentados os objetivos atingidos e as conclusões retiradas da realização da presente Dissertação.

4.1 Simulação de elevadores no Simio®

A simulação de elevadores no *software* Simio® fica provada como possível.

Esta obedece a um algoritmo simples, que pode evoluir e simular um sistema baseado no destino do cliente, com vários elevadores a servirem um edifício.

A utilização do elevador como entidade torna-se evidente, pela sua flexibilidade e especificidade.

O modelo de simulação com o veículo apresenta limitações. A mais evidente é a maior prioridade à entrega das entidades que estão na fila do elevador em detrimento da receção de novos clientes. O veículo apenas rececionará clientes quando: realiza uma paragem para libertar um cliente da sua fila e caso esteja vazio, respondendo primeiramente às solicitações mais próximas.

O modelo de simulação com o objeto entidade como elevador revelou-se mais adequado à aplicação e torna-se natural a sua escolha, uma vez que o veículo apresenta sérias limitações, não permitindo simular um elevador no seu algoritmo mais básico. Em aplicações mais avançadas e complexas, com a necessidade de algoritmos robustos, as fragilidades seriam ampliadas.

O carácter oculto dos seus processos também é um ponto a desfavor, pois poderá ser uma limitação no futuro, podendo criar um processo que irá contra um já existente, mas não acessível ao utilizador.

A utilização do elevador como entidade torna-se evidente, pela sua flexibilidade e possível especificidade.

4.2 Análise

O grande objetivo da simulação é a retirada de dados. A possibilidade de simular antes de alterar, conceber e implementar é algo valorizado, uma vez que permite verificar os benefícios antes do investimento, permitindo tomar uma decisão baseada em dados.

No caso dos elevadores, a troca do sistema ou a simples alteração do algoritmo tem custos avultados e obriga à inoperacionalização do sistema por um determinado espaço de tempo. A tomada de decisão baseada em dados irá certamente ajudar os proprietários dos elevadores a decidirem o sistema a

implementar e os parâmetros a definir. Um destes parâmetros será o tempo de abertura das portas do elevador.

Um tempo de abertura de portas baixo irá diminuir o tempo que o elevador está parado, regressando mais rapidamente ao movimento. Mas, ao mesmo tempo, também irá bloquear a entrada a alguns clientes, obrigando-os a esperar por outro elevador, aumentando drasticamente o tempo de espera.

Por contraste, um tempo de abertura de portas maior permitirá a entrada de mais passageiros, através de uma maior janela temporal, mas terá impacte direto na diminuição do número de viagens realizadas e consequentes entregas de clientes ao seu destino, resultando em tempos de viagem superiores.

Com os dados retirados das simulações realizadas, os gráficos obtidos e as leituras efetuadas, é possível afirmar: o tempo de abertura de portas aconselhável para a situação simulada é de onze segundos. Este ponto apresenta não só o menor tempo de atravessamento do sistema, como também regista valores baixos de quantidade de clientes em espera, clientes no interior do sistema, ocupação média do elevador e número de movimentações na parte inferior da curva.

Isto resulta num bom desempenho do sistema do ponto do vista do cliente – com baixas filas de espera, tempos de espera moderados e baixos tempos de atravessamento – mas também em consumos energéticos moderados – quando são comparados as medidas número de movimentações e taxa de ocupação deste cenário com os restantes.

4.3 Balanço do trabalho

A presente Dissertação representou um desafio. Interessante e desafiante: realizar a simulação de um sistema de elevadores num *software* onde nunca tinha sido feito à escala pretendida: sete andares.

Foi, portanto, uma jornada difícil, que requereu várias horas de teste e tentativas falhadas.

Esta falta de comparação levou a um tempo maior do que o previsto para desenvolver o modelo e respetivo algoritmo.

Apesar disto, o modelo desenvolvido é robusto e, se aplicado em situações reais, irá trazer dados sólidos, que permitiram tomar melhores decisões.

4.4 Futuras melhorias e implementações

O modelo atual, pela sua flexibilidade, poderá ser utilizado para testar diferentes algoritmos, como é, por exemplo, a *milk run*. Isto é, um elevador que pare em todos os andares. Estes diferentes algoritmos terão diferentes desempenhos, que, quando aliados a dados sólidos de movimentação de passageiros no interior de um edifício, poderá ser uma peça chave para a tomada de decisão.

Com o modelo atual, utilizando a capacidade do Simio® de transformar um modelo em objeto e reutilizá-lo num novo modelo, será possível replicar o elevador atual e implementar um sistema de vários elevadores. Tal sistema necessitaria de um ponto de divisão, isto é, um ponto anterior ao *node* de transferência para o interior do elevador, onde seria tomada a decisão de que elevador o cliente iria utilizar.

Nesse ponto, vários algoritmos poderão ser aplicados, como é o exemplo do Sistema ETD – visto no subcapítulo 1.3.4 na página 21 – que é baseado na Equação 1; ou um algoritmo mais simples, que encaminhará o cliente para o elevador mais próximo do sentido pretendido.

A aferição de gastos energéticos por parte do sistema de tração do elevador irá permitir determinar os pontos de menor consumo energético tendo em conta as medidas ocupação e movimentações. Este estudo seria proveitoso, especialmente quando conjugado com sistemas de vários elevadores, onde um, ou mais, elevadores do grupo pode ser desligado quando a afluência de passageiros é menor.

Todas estas possibilidades, aliadas a uma boa recolha de dados, poderão dar bons resultados e ter implicações reais e mensuráveis.

REFERÊNCIAS BIBLIOGRÁFICAS

About Elevate. (n.d.). Retrieved January 23, 2016, from https://www.peters-research.com/index.php/elevate/elevate-elevate-express

About Elevate Live. (n.d.). Retrieved January 23, 2016, from https://www.peters-research.com/index.php/elevate-live/more-about-elevate-live

About Simio. (n.d.). Retrieved January 23, 2016, from http://www.simio.com/about-simio/

Alagar, V. S., & Periyasamy, K. (2011). *Specification of Software Systems*. Springer Science & Business Media. Retrieved from https://books.google.com/books?id=xgwKrcy0TzAC&pgis=1

Analysis of Elevator Performance and Passenger Demand with Destination Control. (n.d.). Retrieved January 23, 2016, from https://www.peters-research.com/index.php/support/articles-and-papers/41-analysis-of-elevator-performance-and-passenger-demand-with-destination-control

Barney, G., & Lutfi Al-Sharif. (2015). *Elevator Traffic Handbook: Theory and Practice*. (Routledge, Ed.) (2nd ed.). Routledge. Retrieved from https://books.google.com/books?id=799zCgAAQBAJ&pgis=1

Basic types of Elevators. (n.d.). Retrieved January 23, 2016, from http://www.otis.com/site/us/pages/AboutElevators.aspx?menuID=2

Bernard, A. (2014). *Lifted: A Cultural History of the Elevator*. NYU Press. Retrieved from https://books.google.com/books?id=ryegAgAAQBAJ&pgis=1

C. Dennis Pegden. (n.d.). Retrieved January 23, 2016, from http://d.lib.ncsu.edu/computer-simulation/people/c-dennis-pegden-ph-d

Craighead, G. (2009). *High-Rise Security and Fire Life Safety*. Butterworth-Heinemann. Retrieved from https://books.google.com/books?id=4BWyBELDQIwC&pgis=1

Current Technology and Future Developments in Elevator Simulation. (n.d.). Retrieved January 23, 2016, from https://www.peters-research.com/index.php/support/articles-and-papers/47-current-technology-and-future-developments-in-elevator-simulation

Elevator control panels. (n.d.). Retrieved January 23, 2016, from http://www2.isye.gatech.edu/~jjb/misc/elevators/elevators.html

Elevator Dial Floor Indicator. (n.d.). Retrieved January 23, 2016, from http://www.innovationind.com/index.php?submenu=position&src=gendocs&ref=Elevator-Position-Indicators &category=fixtures

ElevatorTour's most interesting photos. (n.d.). Retrieved January 23, 2016, from

http://www.flickriver.com/photos/elevatortour/popular-interesting/

Elisha Graves Otis. (n.d.). Retrieved January 23, 2016, from http://www.britannica.com/biography/Elisha-Graves-Otis

First electric elevator. (n.d.). Retrieved from https://www.siemens.com/history/en/news/1043_elevator.htm

Forero, A. (2010, December 1). *Estudo e simulação de técnicas de controle de tráfego de grupo de elevadores usando automação industrial*. Retrieved from http://repositorio.unb.br/handle/10482/8893?mode=full

Gray, L. E. (2002). *From Ascending Rooms to Express Elevators: A History of the Passenger Elevator in the 19th Century*. Elevator World Inc. Retrieved from https://books.google.com/books?id=Qq4gVSkfnPcC&pgis=1

Henri Hakonen, & Siikonen, M.-L. (2009). Elevator Traffic Simulation Procedure. *Lift Report*, (5). Retrieved from http://www.lift-report.de/index.php?mact=News,cntnt01,print,0&cntnt01articleid=259&cntnt01showtemplate=false&cntnt01returnid=360

History of Elevators. (n.d.). Retrieved January 23, 2016, from http://www.vertechselevators.com/about-vertechs/vertechs-articles/history-of-elevators/

History of the Elevator. (n.d.). Retrieved January 23, 2016, from http://www.mitsubishielectric.com/elevator/overview/elevators/history.html

Hoorah! Shanghai Tower crowned world's second tallest skyscraper. (n.d.). Retrieved January 23, 2016, from http://shanghaiist.com/2016/01/12/shanghai_tower_number_2.php

Janovský, L. (1999). *Elevator Mechanical Design*. Elevator World Inc. Retrieved from https://books.google.com/books?id=b_FutCkJvbUC&pgis=1

Jarboe, T. L., & O'Donoghue, J. (2007). *Elevator and Escalator Rescue: A Comprehensive Guide*. Fire Engineering Books. Retrieved from https://books.google.com/books?id=rzTSE9rvAS0C&pgis=1

Management Team. (n.d.). Retrieved January 23, 2016, from http://www.simio.com/about-simio/the-simio-mgmt-team/

McCain, Z. (2007). *Elevators 101* (2nd Editio). Elevator World Inc. Retrieved from https://books.google.com/books?id=0yWTIo-CXXsC&pgis=1

Mitsubishi Electric to Install World's Fastest Elevators in Shanghai Tower. (n.d.). Retrieved January 23, 2016, from http://www.mitsubishielectric.com/news/2011/0928.html

Mitsubishi Electric's new elevator is the fastest on Earth. (n.d.). Retrieved January 23, 2016, from https://www.techinasia.com/mitsubishi-electric-worlds-fastest-elevator-shanghai-tower

Molyneaux, M. (2006). Real life experiments that reveal the ancient art and techniques of building Egyptian pyramids. Retrieved January 23, 2016, from http://www.catchpenny.org/mmbuild.html

National Inventors Hall of Fame. (n.d.). Retrieved January 23, 2016, from http://invent.org/inductee-detail/?IID=115

Otis History. (n.d.). Retrieved January 23, 2016, from http://www.otis.com/site/us/pages/OtisHistory.aspx?menuID=6

Pegden, C. D., & Sturrock, D. T. (2014). *Rapid Modeling Solutions: Introduction to Simulation and Simio*. Retrieved from http://www.simio.com/downloads/public/documents/RapidModelingSolutions_Color.pdf

Peters, R., & Smith, R. (2009). ETD Algorithm with Destination Dispatch and Booster Options. *Peters Research Ltd*. Retrieved from https://www.peters-research.com/index.php/support/articles-and-papers/42-etd-algorithm-with-destination-dispatch-and-booster-options

Prémio EDP: Energia Elétrica e Ambiente 2011. (2010). Retrieved January 23, 2016, from https://www.google.pt/url?sa=t&rct=j&q=&esrc=s&source=web&cd=5&cad=rja&uact=8&ved=0ahUKEwiLjeLb773KAhXKQiYKHatfA-IQFgg3MAQ&url=https://www.edpdistribuicao.pt/pt/infocenter/noticias/2011/EDP%20Documents/Br_Premio_EDP2010_15-2_nov.pdf&us

Rao, P. (2012). Benefits of Destination-Dispatch Systems, 70–73. Retrieved from https://www.google.pt/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0ahUKEwiEqJmEmcDKAhULWBoKHdvjBikQFggjMAA&url=http://www.elevatorworld.com/pdf/EWI_2q12_Feature.pdf&usg=AFQjCNHUuFn9DVI_UaNTB7MIY1_-EL6F-A&sig2=ka-HPTARPrsRG7O6Pc6g

Setchi, R. (2010). Knowledge-Based and Intelligent Information and Engineering Systems. In R. Setchi & I. Jordanov (Eds.), *Knowledge-Based and Intelligent Information and Engineering Systems* (Vol. 1, pp. 133–134). Cardiff, UK: Springer Science & Business Media. Retrieved from https://books.google.com/books?id=8-vR1Nawg6AC&pgis=1

Simio Software Discussion Forum. (n.d.). Retrieved January 23, 2016, from http://www.simio.com/forums/

Smith, D. L. (2011). *Environmental Issues for Architecture*. John Wiley & Sons. Retrieved from

https://books.google.com/books?id=deH-4fRljnAC&pgis=1

Sobre elevadores. (n.d.). Retrieved January 23, 2016, from http://www.otis.com/site/pt/pages/sobreelevadores.aspx

Strakosch, G. R., & Caporale, R. S. (2010). *The Vertical Transportation Handbook*. (G. R. Strakosch & R. S. Caporale, Eds.) (4th Editio). John Wiley & Sons. Retrieved from https://books.google.com/books?id=jMRc3Sy8qakC&pgis=1

The Inventors. (n.d.-a). Retrieved January 23, 2016, from http://www.theelevatormuseum.org/e/E-5.htm

The Inventors. (n.d.-b). Retrieved January 23, 2016, from http://www.theelevatormuseum.org/e/e-2.htm

The Other Elevator Inventor. (n.d.). Retrieved January 23, 2016, from http://www.pbs.org/wgbh/nova/tech/elevator-inventor.html

ThyssenKrupp Destination Dispatch Elevator. (n.d.). Retrieved January 23, 2016, from https://www.youtube.com/watch?v=Azgmx2F0lvw

Timeline of the Elevator. (n.d.). Retrieved January 23, 2016, from http://www.theelevatormuseum.org/timeline.php

Who invented the elevator? (n.d.-a). Retrieved January 23, 2016, from http://www.history.com/news/ask-history/who-invented-the-elevator

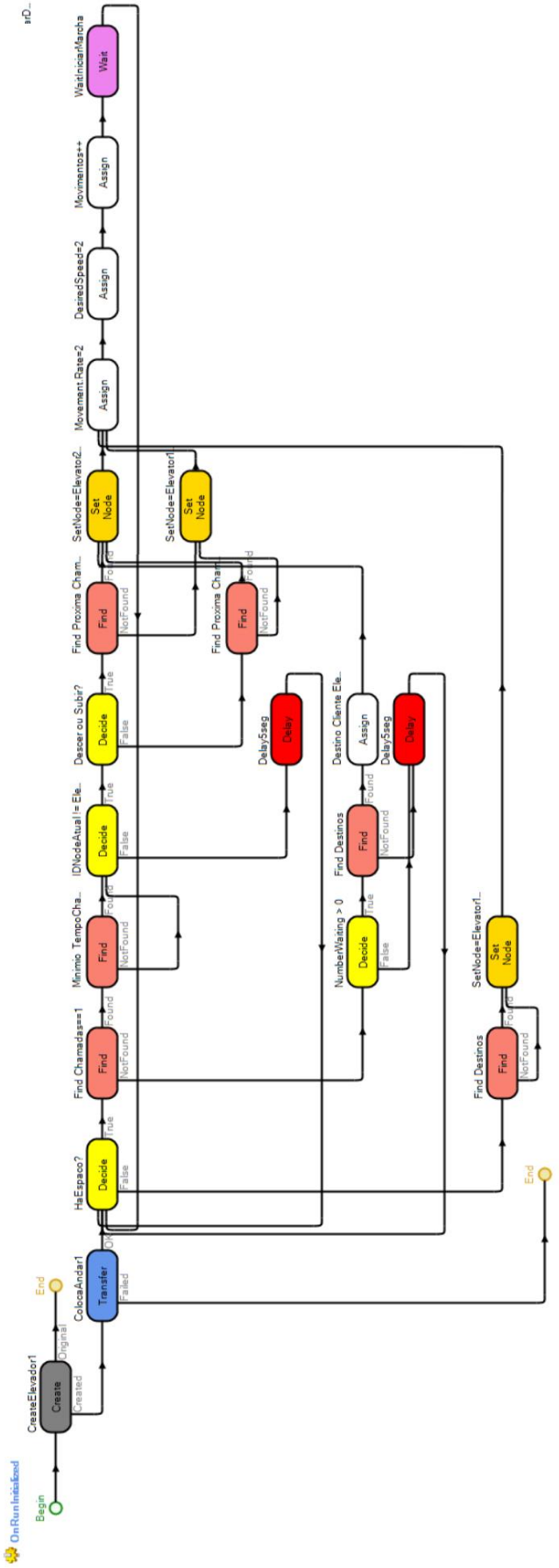
Who Invented the Elevator? (n.d.-a). Retrieved January 23, 2016, from http://gizmodo.com/who-invented-the-elevator-1570745196

Who Invented the Elevator? (n.d.-b). Retrieved January 23, 2016, from http://www.todayifoundout.com/index.php/2014/05/history-elevator/

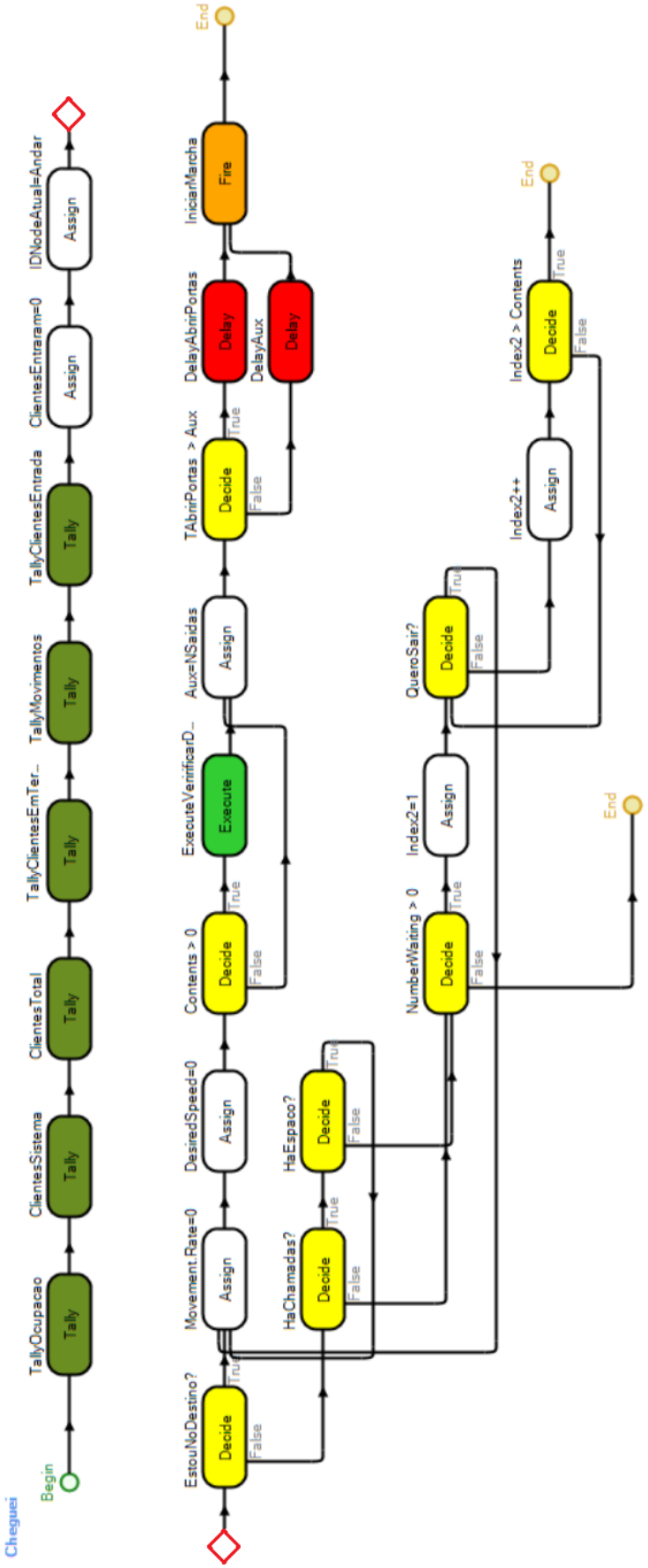
Who invented the elevator? (n.d.-b). Retrieved January 23, 2016, from http://science.howstuffworks.com/innovation/inventions/who-invented-the-elevator1.htm

Zhang, J., & Zong, Q. (2014). Energy-saving-oriented group-elevator dispatching strategy for multi-traffic patterns. *Building Services Engineering Research and Technology*, *35*(5), 543–568. http://doi.org/10.1177/0143624414526723

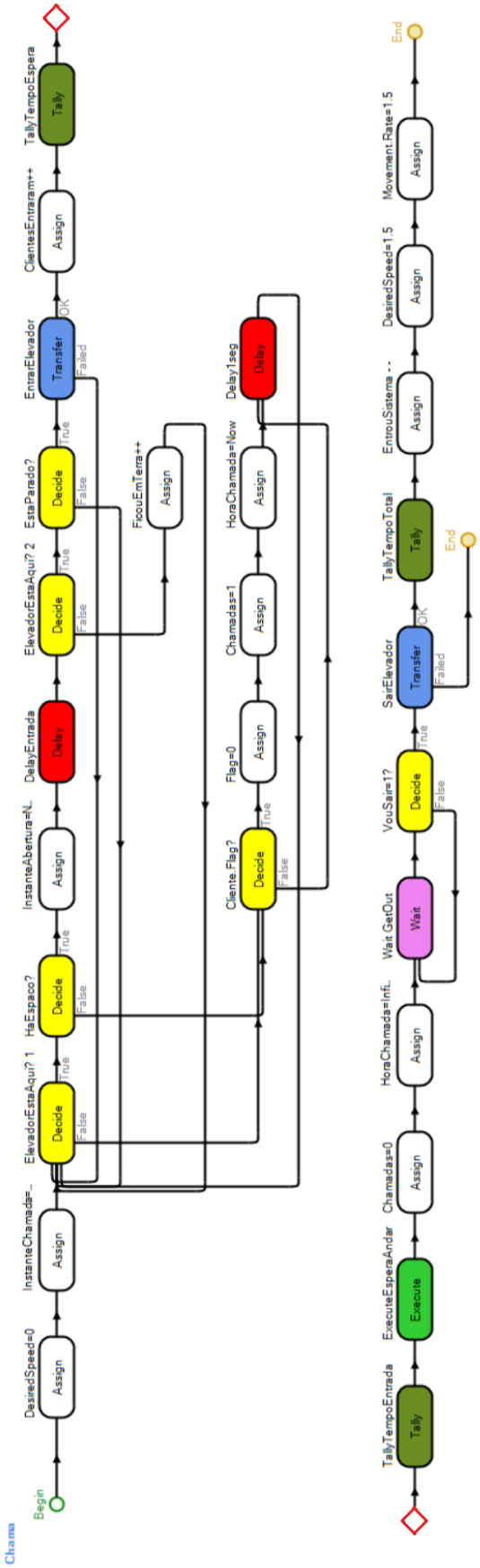
ANEXO I – PROCESSO "*ONRUNINITIALIZED*" DO OBJETO ENTIDADE



ANEXO II – PROCESSO "CHEGUEI" DO OBJETO ENTIDADE



ANEXO III – PROCESSO "CHAMA" DO OBJETO ENTIDADE



ANEXO IV – TABELA RESUMO DE DADOS

Tempo de abertura de portas [segundos]	Tempo total mínimo [minutos]	Tempo total máxima [minutos]	Tempo total médio [minutos]	Ocupação média	Ocupação máxima	Ocupação mínima	Clientes no sistema	Movimentações do elevador	Total de clientes em espera	Tempo de espera médio [segundos]
1	1,2	15,9	8,4	5,2	12,9	0,3	36,0	770,3	30,7	25,6
2	1,3	21,2	11,0	3,5	9,5	0,2	32,9	801,5	29,3	26,6
3	0,9	22,8	14,8	1,3	6,0	0,0	32,0	717,8	30,7	28,6
4	0,7	23,0	14,0	2,5	8,4	0,0	31,4	602,8	28,9	21,9
5	0,7	12,8	7,3	3,0	8,1	0,1	18,4	495,9	15,4	22,0
6	0,7	10,3	5,4	3,8	8,9	0,1	15,1	433,1	11,3	22,3
7	0,6	7,1	3,6	4,1	9,9	0,3	11,2	385,0	7,1	25,9
8	0,8	8,0	3,8	4,8	10,6	0,7	13,1	354,9	8,0	27,4
9	0,8	6,7	3,6	5,2	11,3	0,4	12,4	323,9	7,2	29,5
10	0,8	6,4	3,4	6,8	13,0	1,5	14,7	299,5	7,9	25,7
11	0,8	5,5	3,0	6,2	11,8	1,0	11,6	275,2	5,3	35,7
12	1,0	6,0	3,1	6,5	12,4	1,1	12,2	259,4	5,6	34,9
13	1,1	5,8	3,2	7,9	14,7	1,6	14,3	242,6	6,5	32,1
14	0,9	6,2	3,3	7,9	15,4	1,8	14,3	228,7	6,3	36,9
15	0,9	6,3	3,3	8,7	16,4	1,4	14,2	216,6	5,6	41,0
17	0,9	8,6	4,6	11,0	18,4	2,1	18,5	197,9	7,3	34,9
20	1,0	12,0	5,8	12,4	20,2	3,3	23,3	176,3	10,9	34,5