



TravelMaster AI: Arquitetura Neuro-Simbólica para Assistência Autónoma na Execução de Viagens

JOÃO CARLOS DA SILVA CORDEIRO

Junho de 2026

TravelMaster AI: Arquitetura Neuro-Simbólica para Assistência Autónoma na Execução de Viagens

João Cordeiro

**Dissertação para obtenção do Grau de Mestre em
Engenharia Informática, Área de Especialização em
Engenharia de Software**

Orientadora: Piedade Carvalho

Porto, 27 de junho de 2026

Declaração de Integridade

Declaro ter conduzido este trabalho académico com integridade.

Não plagiei ou apliquei qualquer forma de uso indevido de informações ou falsificação de resultados ao longo do processo que levou à sua elaboração.

Portanto, o trabalho apresentado neste documento é original e de minha autoria, não tendo sido utilizado anteriormente para nenhum outro fim.

Declaro ainda que tenho pleno conhecimento do Código de Conduta Ética do P.PORTO.

ISEP, Porto, 27 de junho de 2026

Dedicatória

"You can never know everything, and part of what you know is always wrong. Perhaps even the most important part. A portion of wisdom lies in knowing that. A portion of courage lies in going on anyway."

— Robert Jordan

Resumo

A execução de viagens permanece uma experiência fragmentada e cognitivamente exigente, obrigando os utilizadores a gerir múltiplas aplicações isoladas para voos, alojamento e navegação. Embora as soluções atuais e os assistentes de Inteligência Artificial Generativa se destaquem no planeamento e inspiração pré-viagem, carecem de autonomia para apoiar o viajante durante a fase de execução, particularmente perante perturbações operacionais. Esta investigação aborda esta lacuna propondo o TravelMaster AI, um sistema inteligente concebido para atuar como um agente autónomo de monitorização e assistência de viagens em tempo real.

O sistema procura centralizar dados operacionais e automatizar a reação a eventos imprevistos através da orquestração de serviços heterogéneos via API. Tirando partido de Grandes Modelos de Linguagem (LLM) e algoritmos de otimização determinísticos, o TravelMaster AI fornece monitorização proativa do estado dos voos e gera sugestões de reencaminhamento sensíveis ao contexto, eliminando a necessidade de alternância manual de contexto.

Este projeto emprega a metodologia *Design Science Research* para conceber, construir um protótipo funcional e avaliar a eficácia do sistema. O objetivo é demonstrar que um agente de monitorização autónomo pode reduzir significativamente a carga cognitiva da gestão de viagens, proporcionando uma experiência contínua, resiliente e unificada durante a jornada.

Palavras-chave: Execução de viagens, Inteligência Artificial, Agentes Autónomos, Monitorização em tempo real, Carga cognitiva, Integração de APIs

Abstract

Travel execution remains a fragmented and cognitively demanding experience, forcing users to juggle multiple isolated applications for flights, accommodation, and navigation. While current solutions and Generative AI assistants excel at pre-trip planning and inspiration, they lack the agency to support the traveler during the execution phase, particularly when facing operational disruptions.

This research addresses this gap by proposing TravelMaster AI, an intelligent system designed to function as an autonomous agent for real-time travel monitoring and assistance. The system aims to centralize operational data and automate reaction to unforeseen events by orchestrating heterogeneous services via APIs. Leveraging Large Language Models (LLM) and deterministic optimization algorithms, TravelMaster AI provides proactive monitoring of flight status and generates context-aware re-routing suggestions, eliminating the need for manual context switching.

This project employs a Design Science Research methodology to design, construct a functional prototype, and evaluate the system's effectiveness. The objective is to demonstrate that an autonomous monitoring agent can significantly reduce the cognitive load of travel management, providing a seamless, resilient, and unified experience during the journey.

Keywords: Travel execution, Artificial intelligence, Autonomous Agents, Real-time monitoring, Cognitive load, API integration.

Agradecimentos

Gostaria de expressar a minha profunda gratidão a todos os que me acompanharam ao longo do meu percurso acadêmico e durante o desenvolvimento desta dissertação.

Um agradecimento especial à minha orientadora, Professora Piedade Carvalho, pela sua orientação constante e total disponibilidade ao longo de todo o projeto.

À minha família, deixo o meu mais sincero reconhecimento. Obrigado pelo apoio incondicional, pela paciência inesgotável e pela compreensão demonstrada.

Aos meus amigos e colegas, deixo também uma palavra de profundo apreço. Aos que já percorreram este caminho, agradeço a permanente disponibilidade para esclarecer dúvidas e partilhar a sua experiência, aos que partilham atualmente este percurso comigo, obrigado por dividirem as mesmas ansiedades, tornando-as significativamente mais leves, e aos que se seguirão, espero poder um dia retribuir o mesmo nível de apoio.

Conteúdo

Lista de Figuras	xvii
Lista de Tabelas	xix
Lista de Algoritmos	xix
Lista de Código	xix
Lista de Símbolos	xxi
Lista de Acrónimos	xxiii
1 Introdução	1
1.1 Contextualização	1
1.2 Motivação e Relevância	2
1.3 Definição do Problema e Objetivos	2
1.4 Delimitação e Âmbito do Projeto	3
1.5 Metodologia de Trabalho	3
1.6 Contribuições da Dissertação	4
1.7 Estrutura do Documento	5
2 Estado da Arte	7
2.1 Revisão das Soluções Existentes	7
2.2 Revisão de Literatura	9
2.3 Análise Comparativa de Soluções	15
3 Metodologia de Investigação	17
3.1 Fases da Metodologia de Investigação	18
3.2 Questões de Investigação	22
3.3 Considerações Éticas	23
3.3.1 Privacidade e Proteção de Dados	24
3.3.2 Segurança Transacional e Interoperabilidade	24
3.3.3 Transparência e Autonomia Decisória da IA	24
3.3.4 Matriz de Riscos e Mitigação	25
3.4 Âmbito e Limitações do Protótipo	26
4 Plano de Trabalho	29
4.1 Fases de Desenvolvimento	29
5 Arquitetura e Conceção da Solução	31
5.1 Princípios Arquiteturais	31
5.2 Arquitetura Híbrida de Orquestração	32

5.3	Stack Tecnológica e Ferramentas	32
5.4	Arquitetura de Implementação e Componentes	33
5.5	Dinâmica de Orquestração e <i>Tool Calling</i>	34
5.6	Padrões de Desenho e Decisões Arquiteturais	35
5.6.1	Padrões Clássicos e Isolamento para DSR	35
5.6.2	Arquitetura Neuro-Simbólica e Padrões Agênticos	35
5.7	Módulo de Recomendação e Mitigação de Fricção	36
5.7.1	Lógica de Pontuação Subtrativa	36
5.7.2	Integração de Explainable Artificial Intelligence (XAI) via <i>Reasoning Strings</i>	37
6	Implementação do Artefacto	39
6.1	Ambiente de Desenvolvimento e Stack Tecnológica	39
6.1.1	Linguagem e Ferramentas Base	39
6.1.2	Bibliotecas Vitais (<i>Core Dependencies</i>)	39
6.1.3	Estrutura de Diretórios	40
6.2	Ingestão e Estruturação de Contexto	40
6.2.1	Tipagem Estrita com Pydantic	40
6.2.2	Extração Multimodal via Gemini 2.5 Flash	41
6.3	Construção do Orquestrador e Chamada de Ferramentas	41
6.3.1	Enquadramento Comportamental e Mitigação Estocástica	42
6.3.2	Roteamento Semântico e Tipagem de Ferramentas	42
6.3.3	Vinculação de Ferramentas na Orquestração	43
6.4	O Motor Lógico e Restrições Físicas	43
6.4.1	O Princípio do Disjuntor Lógico	43
6.4.2	Validação Determinística de Escalas	44
6.4.3	Integração de Inteligência Artificial Explicável	44
6.5	Heurística e Inteligência Artificial Explicável	45
6.5.1	Implementação do Ciclo de Fricção	45
6.5.2	Transparência Algorítmica e Orquestração Empática	46
6.6	Interface e Gestão de Estado da Sessão	47
6.6.1	Persistência do Contexto de Viagem	47
6.6.2	Gestão da Conversação e Transparência Cognitiva	48
6.6.3	Conclusão da Arquitetura de Apresentação	48
7	Avaliação e Resultados	49
7.1	Metodologia de Avaliação em Ambiente Laboratorial	49
7.1.1	Especificações do Ambiente de Teste e Validade Ecológica	50
	Parametrização e Reprodutibilidade Experimental	51
7.2	Avaliação de Desempenho e Resiliência (QI1)	51
7.2.1	Latência de Processamento do Orquestrador	51
7.2.2	Tolerância a Falhas e Degradação Graciosa	52
7.3	Avaliação do Esforço Operacional	53
7.3.1	Cenário de Teste: Disrupção Multimodal Simultânea	53
7.3.2	Análise de Fricção de Interação e XAI	54
7.3.3	Análise de Sensibilidade e Casos de Fronteira (<i>Edge Cases</i>)	55
7.4	Discussão e Resposta às Questões de Investigação	56
7.5	Ameaças à Validade e Limitações Experimentais	57
7.5.1	Limitações Experimentais	57

7.5.2	Ameaças à Validade	58
8	Conclusões e Trabalho Futuro	59
8.1	Síntese do Trabalho Desenvolvido	59
8.2	Contribuições Principais	60
8.3	Limitações do Estudo	61
8.4	Trabalho Futuro	62
	Referências	65
	Apêndice A Instruções do Sistema do Modelo de Linguagem (System Prompts)	71
A.1	Diretiva Base do Orquestrador (System Instruction)	71
A.2	Metadados de Configuração da Inferência	71
	Apêndice B Estruturas de Dados de Simulação (Mock Payloads)	73
B.1	Payload de Evento de Disrupção Aérea (Flight Status Update)	73
B.2	Payload de Resposta de Pesquisa de Alternativas (Flight Search Results)	74
B.3	Payload de Estado do Trânsito Terrestre (Terrestrial Route Status)	74
	Apêndice C Artigo Científico Submetido	77
	Apêndice D Comprovativo de Submissão do Artigo	83

Lista de Figuras

3.1	Instanciação do modelo DSR	18
3.2	Arquitetura do TravelMaster AI	20
3.3	Fluxo de Detecção Autónoma	20
4.1	Cronograma do Projeto	29
5.1	Diagrama de Componentes UML da Arquitetura do TravelMaster AI.	34
5.2	Diagrama de Sequência UML detalhando o ciclo de orquestração, o <i>Tool Calling</i> e a iteração de validação lógica face a uma disrupção.	35

Lista de Tabelas

2.1	Análise comparativa do <i>TravelMaster AI</i> face às soluções existentes.	15
3.1	Matriz de Avaliação: Correlação entre QIs, Métricas e Critérios de Sucesso.	23
3.2	Matriz de Riscos e Mitigação: Ameaças e Estratégias Técnicas.	25
7.1	Estatística descritiva da latência de processamento do Orquestrador ($N = 30$).	52
7.2	Estatística descritiva da latência: Disrupção Terrestre Isolada ($N = 30$). . .	52
7.3	Análise de Tarefas: Método Manual fragmentado vs. <i>TravelMaster AI</i> . . .	54
7.4	Correspondência explícita entre Questões de Investigação, Evidências e Resultados Obtidos.	57

Lista de Símbolos

Lista de Acrónimos

API	Application Programming Interface.
B2B	Business-to-Business.
B2C	Business-to-Consumer.
CARS	Context-Aware Recommender Systems.
CSS	Cascading Style Sheets.
DSR	<i>Design Science Research.</i>
GDS	Global Distribution System.
HTML	HyperText Markup Language.
IATA	International Air Transport Association.
JS	JavaScript.
KISS	Keep It Simple, Stupid.
KLM	Keystroke-Level Model.
LLM	Large Language Model.
MVP	Minimum Viable Product.
NASA-TLX	NASA Task Load Index.
NDC	New Distribution Capability.
OCR	Optical Character Recognition.
PNR	Passenger Name Record.
SDK	Software Development Kit.
SUS	System Usability Scale.
TLS	Transport Layer Security.
TTDP	Tourist Trip Design Problem.
UI	User Interface.
UX	User Experience.

XAI Explainable Artificial Intelligence.

Capítulo 1

Introdução

1.1 Contextualização

A indústria do turismo, impulsionada pela transformação digital, vive uma era de «Turismo Inteligente» (*Smart Tourism*), um paradigma focado na sinergia entre Tecnologias de Informação, análise de dados e a experiência turística (Gretzel et al. 2015). Neste ecossistema, a tecnologia e a inteligência artificial redefinem fundamentalmente a experiência do viajante. Ferramentas digitais, aplicações móveis e plataformas de recomendação tornaram-se omnipresentes, prometendo simplificar o processo de descoberta e reserva.

No entanto, apesar da omnipresença de ferramentas digitais, a experiência real de viajar permanece um processo fragmentado e cognitivamente exigente (Werthner e Francesco Ricci 2004). O viajante moderno vê-se obrigado a gerir um ecossistema desconexo de múltiplas aplicações simultâneas: a aplicação da companhia aérea para o cartão de embarque, a do aeroporto para verificar a porta, ferramentas de mapas para a navegação, aplicações de mobilidade para o transporte de ligação, e plataformas de alojamento para o registo de entrada.

Esta dispersão obriga o utilizador a alternar constantemente entre contextos digitais, gerando ansiedade e ineficiência, especialmente em situações de imprevisto ou pressão temporal. Embora a Inteligência Artificial Generativa tenha avançado na sugestão de itinerários, esta fragmentação operacional durante a execução da viagem persiste.

Neste contexto, é imperativo distinguir o foco desta investigação das soluções comerciais e académicas prevalentes. O problema central abordado nesta dissertação não é a inexistência de ferramentas para o planeamento de viagens, sistemas de recomendação turística, agregação de reservas ou geração estática de itinerários. O mercado atual encontra-se saturado com estas soluções. A verdadeira lacuna tecnológica, e o motor deste projeto, reside na ausência de sistemas unificados capazes de:

- Monitorizar continuamente a execução da viagem no plano físico.
- Reagir de forma autónoma e proativa a disrupções em tempo real.
- Validar alternativas de replaneamento de forma puramente determinística (mitigando as alucinações inerentes aos LLMs).
- Reduzir drasticamente a carga operacional e cognitiva do utilizador em cenários dinâmicos de elevado stress.

1.2 Motivação e Relevância

A motivação para este projeto nasce da constatação desta lacuna crítica na continuidade da experiência do viajante. As soluções estabelecidas, como o Triplt, funcionam primariamente como agregadores passivos de documentos, dependendo da intervenção manual e não oferecendo assistência em tempo real. Por outro lado, plataformas como o Google Travel, embora preditivas, oferecem uma personalização generalista e não atuam como um assistente centralizado que unifica a logística da viagem momento a momento.

Mesmo as novas ferramentas baseadas em Grandes Modelos de Linguagem (LLM), embora excelentes na fase de inspiração, falham na fase de execução e acompanhamento. Atualmente, não existe uma solução que monitorize proativamente o estado dos serviços (ex: atrasos de voos, trânsito para o aeroporto) e adapte o itinerário automaticamente.

Falta, portanto, transitar do modelo de «assistente» reativo para o de «agente» autônomo: um sistema capaz de perceber o ambiente, raciocinar sobre imprevistos e atuar de forma independente para mitigar a complexidade da viagem (Franklin e Graesser 1997).

A relevância deste trabalho reside no potencial de eliminar a necessidade de alternar entre múltiplas plataformas. Ao centralizar a informação e automatizar a monitorização, a solução proposta reduz o esforço cognitivo e torna a viagem mais fluida, cumprindo a promessa de melhoria da experiência turística através da personalização contextual (Buhalis e Amaranggana 2015).

1.3 Definição do Problema e Objetivos

O problema de investigação central que esta dissertação aborda é a elevada fragmentação da experiência do viajante e a consequente sobrecarga cognitiva durante a fase de execução da viagem.

Atualmente, o viajante vê-se obrigado a gerir manualmente um conjunto disperso de aplicações para realizar tarefas logísticas críticas (ex: verificar portas de embarque, confirmar horários de registo de entrada, localizar transportes ou reagir a atrasos). Esta dispersão obriga a uma constante alternância de contexto entre múltiplas plataformas isoladas, o que gera ineficiência e ansiedade, especialmente perante imprevistos.

O problema reside, portanto, na ausência de um sistema unificado que atue não apenas como um planeador estático, mas como um agente de execução e monitorização capaz de centralizar a logística da viagem e assistir o utilizador proativamente em tempo real. Esta lacuna de execução é particularmente crítica dado que, conforme demonstrado por testes recentes, os agentes de IA atuais ainda revelam limitações significativas na harmonização fiável de restrições logísticas complexas e dinâmicas (Xie et al. 2024).

Em linha com uma metodologia *Design Science Research* (DSR) (Hevner et al. 2004), o objetivo principal deste trabalho é a conceção, construção e avaliação de um artefacto de software, um protótipo funcional do sistema inteligente *TravelMaster AI*, como resposta ao problema identificado.

Este sistema visa atingir os seguintes Objetivos Específicos (OE):

- **OE1:** Analisar e definir uma arquitetura de agente capaz de centralizar a informação operacional de múltiplos fornecedores (voos, alojamento, navegação), superando o desafio da interoperabilidade técnica para eliminar a necessidade de consulta de múltiplas aplicações.
- **OE2:** Operacionalizar mecanismos de monitorização proativa e assistência contextual que, recorrendo a IA, permitam ao sistema detetar alterações de estado e sugerir autonomamente alternativas de replaneamento ou rotas otimizadas.
- **OE3:** Validar a eficácia do sistema na redução da carga cognitiva e do esforço de interação manual do viajante, demonstrando através de cenários simulados que a centralização da informação melhora a fluidez e a resiliência da experiência de viagem face a imprevistos.

1.4 Delimitação e Âmbito do Projeto

Para garantir o rigor e a exequibilidade da investigação, torna-se essencial delimitar o âmbito de atuação do protótipo desenvolvido. O *TravelMaster AI* foca-se exclusivamente na orquestração de dados de contexto e no apoio à decisão tática do viajante durante a execução da viagem. Deste modo, importa clarificar o que não constitui objetivo deste trabalho:

- **Substituição de Infraestruturas Base:** O sistema não pretende substituir plataformas de *Online Travel Agencies* (OTA) ou *Global Distribution Systems* (GDS). Atua, ao invés, como uma camada de inteligência e assistência sobreposta aos dados fornecidos por essas entidades.
- **Comerciabilidade do Sistema :** O artefacto não representa, na sua iteração atual, uma infraestrutura comercial pronta para produção em larga escala. Consiste numa Prova de Conceito (*Proof of Concept*) construída e avaliada num ambiente laboratorial controlado (com recurso a simulação de telemetria) para validar a eficácia da arquitetura neuro-simbólica.
- **Autonomia Irrestrita:** O projeto não visa dotar o agente de autonomia total sem supervisão humana. O sistema opera sob o paradigma *Human-in-the-Loop*, onde o agente propõe alternativas rigorosamente validadas, mas a decisão de compra, alteração ou aceitação final pertence invariavelmente ao utilizador.

1.5 Metodologia de Trabalho

Para alcançar os objetivos de conceção e avaliação definidos na secção 1.3, a presente dissertação adota formalmente a metodologia DSR, conforme delineada por Hevner et al. (2004). Esta abordagem é a mais adequada, pois foca-se na resolução de problemas do mundo real através da criação e avaliação de um artefacto (o protótipo *TravelMaster AI*) como contributo central da investigação.

O processo segue as fases desta metodologia, que se alinham com os resultados atingíveis do projeto:

1. **Identificação do Problema e Estado da Arte:** Corresponde à interpretação do problema a resolver e à sistematização do conhecimento existente (abordado nos Capítulos 1 e 2), onde se analisa o estado da arte e se justifica a relevância do artefacto.

2. **Concepção e Desenho:** Procede-se à avaliação de diferentes abordagens tecnológicas e à concepção da arquitetura da solução *TravelMaster AI*, definindo os seus componentes e funcionalidades.
3. **Construção do Artefacto:** A fase central do trabalho, consistindo na construção de um protótipo funcional orientado à execução da viagem. O desenvolvimento focou-se em três pilares:
 - **i) Pré-Viagem:** Extração automática de dados essenciais a partir de um plano de viagem importado.
 - **ii) Monitorização de Telemetria:** Implementação de um simulador de telemetria para acompanhamento do estado de voos, imitando a estrutura de dados e a latência de uma API de produção.
 - **iii) Assistência Contextual:** Simulação de eventos de disrupção para validar a capacidade do sistema em gerar notificações proativas e sugestões de percurso.
4. **Avaliação:** Por fim, foi realizada a avaliação do artefacto, utilizando cenários de uso dinâmicos. O objetivo é medir a eficácia do sistema na redução da carga cognitiva e do número de interações manuais necessárias, comparando a experiência unificada do *TravelMaster AI* com a utilização dispersa de múltiplas aplicações convencionais.

1.6 Contribuições da Dissertação

Em estrito alinhamento com a metodologia DSR, o resultado central desta investigação transcende a mera implementação técnica de um protótipo de *software*. O *TravelMaster AI* materializa-se como um artefacto científico cujo desenvolvimento gerou novo conhecimento, colmatando as lacunas identificadas na literatura referentes à execução tática de viagens.

Neste enquadramento, o presente trabalho consolida quatro contribuições fundamentais para o estado da arte, distribuídas por diferentes domínios de conhecimento:

- **Contribuição Científica (Teórica):** Concepção de um mecanismo de monitorização e replaneamento contextual. O projeto formaliza a transição do paradigma clássico de sistemas de recomendação turística estática para o domínio dos agentes autónomos de assistência à execução, demonstrando como o contexto físico contínuo pode ser utilizado para orquestrar a logística intermodal.
- **Contribuição Arquitetural:** Desenho e especificação de uma arquitetura híbrida neuro-simbólica. A dissertação propõe um modelo estrutural inovador que separa formalmente a compreensão semântica e interface interativa (delegada ao LLM) da execução lógica e otimização de rotas (delegada a componentes simbólicos e determinísticos).
- **Contribuição Técnica:** Implementação de uma camada de validação determinística para mitigação de inconsistências. O desenvolvimento do motor TTDP atua como um mecanismo prático para mitigar e prevenir as alucinações logísticas inerentes à Inteligência Artificial Generativa, forçando o sistema a respeitar restrições físicas rígidas antes da apresentação de qualquer alternativa ao utilizador.
- **Contribuição Metodológica:** Criação de uma *framework* laboratorial para avaliação de cenários de disrupção. O projeto concebeu um ambiente de simulação controlada e injeção de falhas que permite testar sistemas críticos de *Smart Tourism* (avaliando

métricas de latência e esforço operacional) sem comprometer a segurança ou o bem-estar psicológico de passageiros em trânsito real.

Esta categorização delimita claramente o valor da investigação, garantindo que o desenho concetual do sistema, a sua arquitetura de segurança e o seu método de validação constituem um avanço científico auditável e reproduzível por futuros investigadores.

1.7 Estrutura do Documento

A restante estrutura desta dissertação encontra-se organizada da seguinte forma:

O **Capítulo 2** enquadra a investigação na literatura científica, analisando o Estado da Arte das soluções turísticas e da Inteligência Artificial Generativa. O **Capítulo 3** detalha a Metodologia de Investigação (DSR), estabelecendo as Questões de Investigação e as salvaguardas éticas do projeto, enquanto o **Capítulo 4** descreve o Plano de Trabalho e as respetivas fases de desenvolvimento.

A componente de engenharia divide-se em dois momentos: o **Capítulo 5** foca-se na Arquitetura e Conceção da Solução, estabelecendo o desenho estrutural do modelo híbrido neuro-simbólico, e o **Capítulo 6** aprofunda a Implementação do Artefacto, detalhando o código, a orquestração e a lógica determinística.

O **Capítulo 7** descreve a Avaliação e Resultados empíricos em ambiente laboratorial, validando o esforço operacional e respondendo às questões de investigação. Por fim, o **Capítulo 8** sintetiza as Conclusões da dissertação e perspetiva vias para o Trabalho Futuro.

Capítulo 2

Estado da Arte

2.1 Revisão das Soluções Existentes

Historicamente, a complexidade do planeamento de viagens era gerida por agentes de viagens humanos. Estes profissionais oferecem um serviço completo, desde a consulta de preferências até à reserva e suporte em caso de imprevistos, recorrendo a Sistemas de Distribuição Global (GDS) como Amadeus ou Sabre. Embora esta abordagem Empresa para Empresa (B2B) garanta um serviço completo, carece de escalabilidade, tem custos associados e não oferece a autonomia e a gratificação instantânea que os utilizadores esperam na era digital.

Esta lacuna de escalabilidade e custo levou ao surgimento de soluções Empresa para Cliente (B2C), que procuram simplificar a experiência do utilizador através da integração de múltiplos serviços, sendo o *Triplt* e o *Google Travel* dois dos exemplos mais representativos deste domínio. Ambos os sistemas visam centralizar informações dispersas em diferentes fontes e oferecer suporte ao utilizador em todas as fases do ciclo de vida de uma viagem.

O *Triplt* posiciona-se como um agregador digital de itinerários. A plataforma centraliza reservas e informações de transporte através da importação automática de e-mails de confirmação. Embora eficaz na organização documental, o sistema opera de forma passiva. A ferramenta carece de autonomia, dependendo da intervenção manual do utilizador para qualquer ajuste logístico durante a viagem. Não existe monitorização ativa do estado dos serviços nem reação automática a alterações no plano.

Esta passividade representa uma lacuna académica relevante. O sistema não incorpora princípios de «Sistemas de Recomendação Sensíveis ao Contexto» (CARS), identificados na literatura como cruciais para a relevância turística (Adomavicius e Tuzhilin 2011). Sem a integração de variáveis dinâmicas, como o trânsito ou meteorologia, a plataforma perde a capacidade de adaptação em tempo real. Consequentemente, o sistema falha na assistência proativa perante imprevistos, mantendo a carga cognitiva da resolução de problemas no utilizador.

Por outro lado, o *Google Travel* capitaliza o vasto ecossistema de dados da Google (Maps, Flights, Calendar) para oferecer uma experiência integrada. A plataforma permite planear viagens completas, recorrendo a modelos baseados em histórico e tendências para sugerir voos e alojamentos. No entanto, o nível de personalização permanece generalista. A abordagem alinha-se predominantemente com técnicas de «Filtragem Colaborativa» (baseada no comportamento de outros utilizadores), falhando frequentemente na captura das restrições e preferências específicas do indivíduo (Ricci et al. 2015).

Mais criticamente, a automação do sistema cessa no momento da execução. Embora o *Google Travel* centralize informação estática, não atua como um agente autónomo de gestão.

O sistema não monitoriza proativamente a coerência logística do itinerário em tempo real nem recalcula alternativas automaticamente. A responsabilidade da reação a imprevistos e da validação de horários permanece inteiramente do lado do utilizador, mantendo a carga cognitiva elevada durante a viagem.

Tanto o *Triplt* como o *Google Travel* representam soluções de mercado maduras, mas partilham limitações estruturais na autonomia. Nenhuma destas plataformas utiliza agentes inteligentes para gerir a execução da viagem de forma proativa. A ausência de modelos de Inteligência Artificial Generativa ou contextuais impede a automação da tomada de decisão. Consequentemente, a gestão de imprevistos continua a exigir uma intervenção manual significativa e cognitivamente exigente por parte do utilizador.

Esta exigência cognitiva agrava-se substancialmente perante o fenómeno de fragmentação de interfaces. Num cenário de disrupção operacional, que por si só induz níveis elevados de stress psicológico no passageiro (Li e H. G. Yao 2025), o utilizador é tipicamente forçado a alternar entre múltiplas aplicações (clientes de e-mail, aplicações de navegação rodoviária e portais de companhias aéreas) para recalcular a viabilidade da sua rota.

A literatura de Interação Humano-Computador e Psicologia Cognitiva demonstra que esta alternância não representa um mero inconveniente de usabilidade, mas acarreta um custo cognitivo real e mensurável. Segundo o estudo seminal de Rubinstein et al. (2001), a transição contínua entre tarefas exige processos complexos de controlo executivo, nomeadamente a alteração de objetivos e a ativação de novas regras de interface, o que esgota os recursos mentais e degrada severamente a eficácia da tomada de decisão.

Adicionalmente, este esforço mental de integração de dados ocorre num ambiente de mobilidade física restritivo. Oulasvirta et al. (2005) demonstraram empiricamente que a atenção humana em contextos de trânsito móvel, como a deslocação apressada num terminal de aeroporto, é fragmentada em curtas janelas de apenas 4 a 8 segundos devido à severa competição de recursos imposta pelo ambiente circundante. Sob estas condições, forçar o passageiro a ser o orquestrador manual de múltiplas fontes de informação resulta numa elevada propensão ao erro.

Para mitigar esta sobrecarga, Griffin et al. (2024) argumentam que, no design de sistemas móveis complexos, o esforço cognitivo extrínseco deve ser ativamente distribuído e transferido do utilizador humano para a máquina. Esta evidência reforça a premissa de que a evolução da assistência em viagem não passa pela criação de mais portais passivos de dados, como os oferecidos pelas soluções B2C convencionais, mas pela centralização semântica. Exige-se o desenvolvimento de uma interface conversacional única que isente o utilizador da tarefa de orquestração manual de *software*.

Estas lacunas evidenciam a necessidade de sistemas adaptativos. É imperativo desenvolver agentes capazes de compreender o contexto em tempo real e não apenas o plano estático inicial. O foco da inovação deve transitar da simples agregação de dados para o ajuste dinâmico do itinerário. O objetivo central passa a ser a capacidade de responder autonomamente a perturbações operacionais durante a viagem.

Um desenvolvimento recente no panorama das soluções de viagem é a integração de Assistentes baseados em Inteligência Artificial Generativa. Plataformas como a Expedia ou Kayak utilizam agora Grandes Modelos de Linguagem (LLM) para interpretar pedidos complexos em linguagem natural. No contexto académico, trabalhos como o «TravelAgent» (A. Chen

et al. 2024) demonstram a capacidade destes sistemas para gerar sugestões de itinerários altamente personalizados.

Contudo, a funcionalidade destes assistentes permanece limitada à interação conversacional. Xie et al. (2024) expuseram a fragilidade desta abordagem no estudo de referência «TravelPlanner». A investigação demonstrou que os LLM atuais falham na criação de planos logisticamente viáveis. Os modelos revelam-se incapazes de satisfazer restrições rígidas, como horários sequenciais e conexões geográficas. A taxa de sucesso final registada foi de apenas cerca de 0.6% (Xie et al. 2024). Este resultado valida empiricamente que a execução autónoma de viagens permanece um problema técnico por resolver, justificando a necessidade da arquitetura híbrida proposta pelo *TravelMaster AI*, onde a IA Generativa é complementada por módulos determinísticos de validação lógica.

Perante este cenário de baixa eficácia dos modelos puros, a literatura recente aponta para uma mudança de paradigma: a transição de assistentes conversacionais para agentes autónomos aumentados por ferramentas. Segundo Schick et al. (2023), no trabalho seminal sobre o modelo *Toolformer*, a solução para as falhas lógicas e factuais dos modelos de linguagem não reside apenas no aumento da sua escala, mas na sua capacidade de invocar autonomamente APIs externas. O *Toolformer* demonstrou que um modelo pode aprender a decidir quando e como consultar sistemas especializados (calculadoras, calendários ou motores de pesquisa), delegando funções de elevada precisão a módulos determinísticos sem comprometer a fluência linguística.

Complementarmente, esta evolução é sustentada pelo paradigma *ReAct* (*Reasoning + Acting*), proposto por S. Yao et al. (2023). Esta abordagem propõe a sincronização entre traços de raciocínio lógico e ações concretas, permitindo que o agente induza, acompanhe e atualize planos de ação de forma iterativa com base na observação de resultados externos. Este binómio entre o raciocínio semântico e a execução técnica constitui a base para ultrapassar a passividade das soluções atuais, permitindo a criação de sistemas capazes de navegar a complexidade logística através da consulta dinâmica de fontes de dados em tempo real.

Em síntese, a análise do mercado revela que o ecossistema tecnológico atual é abundante em ferramentas de planeamento, sistemas de recomendação de destinos, plataformas de agregação de reservas e geração estática de itinerários. Contudo, o verdadeiro problema operacional não reside na falta destas ferramentas de pré-viagem. Em contraste, a fase de execução permanece desprovida de uma solução integrada. O *TravelMaster AI* distingue-se precisamente por não competir no espaço do planeamento, posicionando-se para colmatar a inexistência de sistemas capazes de monitorizar continuamente a execução da viagem no mundo físico, reagir autonomamente a disrupções operacionais e reduzir a carga mental do utilizador em cenários dinâmicos. O objetivo central transita, assim, da sugestão estática para a assistência tática em tempo real.

2.2 Revisão de Literatura

A evolução das tecnologias digitais permitiu a emergência do «Turismo Inteligente». Este paradigma visa não apenas a digitalização dos serviços, mas a criação de ecossistemas onde a tecnologia atua como facilitadora ativa da experiência (Gretzel et al. 2015). No entanto, apesar da abundância de dados, persiste o desafio técnico de transformar esta informação em assistência útil e imediata para o viajante, especialmente em situações de mobilidade e gestão de imprevistos.

Os motores de recomendação constituem a base da personalização atual. Segundo Ricci et al. (2015), estes sistemas analisam preferências explícitas e implícitas para sugerir serviços. Contudo, a literatura evidencia uma limitação estrutural: a maioria das abordagens foca-se na recomendação de itens isolados (como um hotel ou voo) com base em histórico estático.

Diversas revisões, incluindo Solano-Barliza et al. (2024), indicam que, apesar da evolução das técnicas de aprendizagem automática, estas permanecem maioritariamente orientadas para apoiar a decisão de compra na fase de planeamento, oferecendo pouca assistência operacional contínua durante a execução da viagem. Esta lacuna fundamenta a inovação central do *TravelMaster AI*: a transição imperativa de sistemas de recomendação estáticos para agentes de monitorização dinâmica. A literatura evidencia que a resolução fiável de disrupções logísticas requer mais do que a geração textual de rotas alternativas, exige sistemas capazes de cruzar variáveis em tempo real e de validar alternativas de forma puramente determinística.

Esta transição para um modelo de assistência proativa é fundamentada pela natureza finita da capacidade de processamento humana. De acordo com a Teoria da Carga Cognitiva (Sweller 1988), a memória de trabalho possui limites estruturais que, quando excedidos por tarefas complexas de resolução de problemas, como a coordenação manual de um itinerário fragmentado, resultam num esgotamento dos recursos mentais que impede a aprendizagem e a ação eficaz.

Este cenário de sobrecarga é severamente agravado pelo stress situacional inerente às disrupções de viagem. Starcke e Brand (2012) demonstram que o stress agudo altera fisiologicamente os processos neurobiológicos de tomada de decisão, comprometendo a capacidade de avaliar alternativas de forma racional e aumentando a propensão para escolhas impulsivas ou sub-ótimas. Perante este défice cognitivo e a urgência temporal, os utilizadores tendem a manifestar o fenómeno de viés de automação, delegando naturalmente a responsabilidade da decisão a sistemas automatizados para aliviar a carga de atenção (Parasuraman e Manzey 2010).

Esta tendência comportamental valida a pertinência do *TravelMaster AI* como um sistema de suporte à decisão necessário em ambientes de alta pressão. Contudo, a literatura alerta que esta delegação exige que o sistema seja não apenas preciso, mas transparente, para evitar que a complacência do utilizador conduza à aceitação acrítica de recomendações, reforçando a importância da arquitetura explicável discutida anteriormente.

O desafio central ultrapassa a simples recomendação de itens, situando-se na orquestração lógica de itinerários. A literatura define este desafio como o «Problema de Design de Viagem Turística» (TTDP). Gavalas et al. (2014) caracterizam-no como um problema de otimização complexo, que exige a harmonização de restrições rígidas de tempo e logística. Contudo, a maioria das abordagens académicas trata o TTDP como um exercício estático de pré-viagem. Esta visão revela-se insuficiente para a fase de execução, onde a verdadeira otimização exige um reajuste dinâmico contínuo face a imprevistos operacionais, e não apenas um planeamento inicial perfeito.

A complexidade inerente a este reajuste contínuo enquadra-se no domínio do Replaneamento Dinâmico e da Gestão de Disrupções. A literatura estabelece uma demarcação clara entre o planeamento estático e o re-escalonamento reativo. No ecossistema da aviação, Su et al. (2021) demonstram que a recuperação de operações perante disrupções é um problema formalmente NP-difícil, exigindo o re-cálculo em cascata de itinerários num curtíssimo espaço de tempo. Dado que os relatórios de desempenho do Eurocontrol indicam que 29,4% dos voos comerciais sofrem atrasos na chegada (EUROCONTROL 2024), a escala do problema

valida a urgência de transitar de sistemas de planeamento estático para agentes de execução tática.

A incapacidade de resolver estas disrupções de forma estática estende-se às redes de mobilidade urbana e transporte público. Abuaisha et al. (2025) formalizam que planos baseados em dados históricos perdem instantaneamente a validade quando sujeitos a perturbações não planeadas. Os autores provam que o paradigma de assistência mais eficaz é o modelo *push*, onde o sistema monitoriza proativamente o estado da rede e sugere alternativas em tempo real (um comportamento análogo ao ciclo de execução central do *TravelMaster AI*), resultando em poupanças substanciais de tempo de chegada e carga cognitiva.

A ausência deste replaneamento dinâmico impõe um custo direto ao passageiro. Utilizando simulações baseadas em agentes (*MATSim*), Rahimi Siegrist et al. (2022) documentam o fosso de informação em redes multimodais, perante uma disrupção, os viajantes forçados a realizar orquestração manual e que carecem de informação integrada em tempo real sofrem atrasos exponencialmente maiores e sub-otimizam as suas rotas. Esta propagação de efeitos em cascata justifica a necessidade crítica de orquestração automatizada para devolver a resiliência à viagem do indivíduo.

As propostas científicas mais avançadas para resolver este constrangimento começam a cruzar as fronteiras da IA Agêntica. Trabalhos recentes, como o de Tan e Fan (2025), demonstram empiricamente que agentes suportados por LLM conseguem efetuar re-roteamento adaptativo em cenários dinâmicos de transporte com tempos de reação inferiores a um segundo, operando como cérebros de roteamento em tempo real. Esta convergência literária solidifica a abordagem da dissertação, a orquestração logística contínua e assistida é o único caminho para dominar o TTDP em ambiente dinâmico.

Contudo, para que esta orquestração tática seja verdadeiramente eficaz e adotada pelo viajante, a resolução do problema não pode ser puramente matemática ou genérica, a personalização no turismo inteligente deve transcender a simples sugestão de conteúdos. Buhalis e Amaranggana (2015) defendem que a verdadeira inteligência reside na utilização de dados contextuais em tempo real. A infraestrutura móvel atual já permite captar sinais críticos, como a localização do utilizador ou o estado do trânsito. No entanto, existe um desfaseamento entre a disponibilidade destes dados e a sua aplicação prática. O paradigma atual foca-se excessivamente na recolha de dados para gestão do destino, negligenciando frequentemente a utilização destes sinais para a assistência operacional direta e imediata ao visitante durante a sua deslocação. O *TravelMaster AI* operacionaliza esta visão ao integrar dados de estado dos voos em tempo real e tempos de transferência dentro do aeroporto no módulo TTDP, permitindo calcular dinamicamente a viabilidade de conexões (ex: «O tempo restante após o atraso é suficiente para cumprir os requisitos de segurança e troca de terminal?»), convertendo dados contextuais em assistência acionável.

A integração técnica de serviços constitui um pilar fundamental. Renaldi et al. (2020) ilustram como o uso de APIs REST se torna essencial para garantir a interoperabilidade prática entre sistemas de transporte, alojamento e outros serviços turísticos num contexto de reserva integrada. No entanto, a mera conectividade técnica não assegura uma experiência unificada. O desafio atual reside na orquestração semântica destes dados dispersos. Embora seja possível aceder a informações de voos isoladamente, a maioria dos sistemas falha na correlação automática entre estes eventos, mantendo a necessidade de o utilizador atuar como o integrador manual das várias fontes de informação.

Recentemente, a Inteligência Artificial Generativa e o Processamento de Linguagem Natural (PLN) emergiram como vetores de inovação. Suanpang e Pothipassa (2024) destacam que a integração destas tecnologias permite criar experiências interativas baseadas em comunicação natural. A capacidade de gerar descrições personalizadas e manter diálogos fluidos representa um avanço significativo na interface com o viajante, facilitando a fase de descoberta e inspiração.

Contudo, é imperativo distinguir fluência verbal de capacidade de otimização. A geração de texto coerente não implica a validação de restrições lógicas complexas. Xie et al. (2024) demonstraram esta limitação de forma quantitativa. O estudo revela que os modelos atuais falham significativamente na resolução do TTDP, caracterizado por Gavalas et al. (2014) pela harmonização rigorosa de horários e logística. A taxa de sucesso na execução de planos válidos de apenas cerca de 0.6% (Xie et al. 2024) comprova que a IA Generativa, por si só, não é suficiente para a gestão autónoma de viagens.

Esta lacuna estrutural justifica o porquê de a aplicação destes modelos em cenários críticos esbarrar constantemente em alucinações logísticas. Na sua extensa taxonomia, Huang et al. (2023) demonstram que, devido à sua natureza probabilística (*sampling* de distribuições estatísticas), os LLMs não oferecem garantias determinísticas de correção. Em contextos de re-escalonamento de voos, isto manifesta-se através de alucinações de fidelidade, sugerindo rotas temporalmente impossíveis ou ligações incompatíveis.

Para colmatar este défice, a fronteira do conhecimento científico aponta para a adoção de Arquiteturas Híbridas Neuro-Simbólicas. M. K. Chen (2025) formaliza esta abordagem evidenciando que a separação de responsabilidades é a solução mais eficaz: a rede neuronal gere a compreensão linguística e a extração de intenções, enquanto um motor externo e programável executa a validação matemática rigorosa. Diversos *frameworks* recentes adotaram este paradigma. Pan et al. (2023) propuseram a arquitetura *Logic-LM*, alcançando aumentos enormes de precisão lógica ao delegar o cálculo a *solvers* simbólicos externos. Em paralelo, a indústria normalizou a utilização de *Guardrails* (Rebedea et al. 2023), consistindo em camadas de controlo em tempo de execução que bloqueiam algoritmicamente retornos que violem regras de negócio restritas.

A eficácia deste acoplamento num contexto de mobilidade foi conclusivamente provada por Hao et al. (2024). Ao aplicarem a orquestração neuro-simbólica ao *benchmark TravelPlanner* (o mesmo que vitimou os modelos puros em Xie et al. (2024)), os autores demonstraram que, ao acoplar um validador matemático que garante a satisfatibilidade lógica do percurso, o sistema saltou para 93,9% de sucesso, reduzindo substancialmente as alucinações operacionais reportadas em configurações sem validação determinística.

Esta é a evidência científica que suporta a arquitetura de validação proposta nesta dissertação. Ao subordinar a criatividade probabilística do modelo Gemini a um filtro determinístico, o *TravelMaster AI* assegura a exequibilidade física e temporal das suas sugestões, convertendo um assistente de conversação volátil num sistema fiável de reajuste operacional de viagens.

A eficácia desta abordagem híbrida não depende apenas da correção lógica, mas da forma como a decisão é comunicada ao utilizador. A literatura de turismo identifica a Inteligência Artificial Explicável (Explainable Artificial Intelligence (XAI)) como uma das prioridades críticas de investigação para a próxima década, enfatizando a necessidade de conceber sistemas inteligíveis e transparentes para mitigar o viés algorítmico (Tussyadiah 2020).

Neste contexto, as explicações em sistemas de apoio à decisão deixam de ser um elemento acessório para se tornarem um requisito fundamental de adoção. Segundo a revisão sistemática de Nunes e Jannach (2017), a transparência é o mecanismo primordial para a construção de confiança, permitindo que o utilizador compreenda a causalidade por trás de uma recomendação. Sem este elo informativo, o passageiro vê-se perante um sistema de caixa negra, o que, em cenários de elevada incerteza, impede a calibração do esforço e da confiança depositada na automação.

A relevância desta transparência é reforçada por evidência empírica recente. Scher et al. (2022) demonstraram que a explicabilidade do raciocínio algorítmico tem uma influência direta e positiva na aceitação do sistema por parte do utilizador. Ao tornar o processo de recomendação transparente, o artefacto permite que o utilizador valide a lógica da máquina face à sua realidade, reduzindo a carga cognitiva e aumentando a probabilidade de aceitação da solução proposta.

No entanto, quando a automação transita da simples recomendação estática para a execução autónoma baseada em agentes, a exigência de transparência sofre uma mutação crítica. Vouros (2022) estabelece que a Explicabilidade Agêntica difere fundamentalmente da XAI tradicional, não basta explicar uma predição isolada, é imperativo justificar uma sequência de ações proativas. Neste domínio, o paradigma fundacional de *Human-Centered XAI* proposto por Ehsan e Riedl (2020) argumenta que a explicação deve transitar da transparência puramente algorítmica, como o modelo calculou, para a transparência social, o porquê de o utilizador dever confiar na alteração do seu voo.

O mecanismo mais promissor para expor este raciocínio é o *Chain-of-Thought* (Wei et al. 2022), que permite ao agente verbalizar os seus passos intermédios de planeamento em linguagem natural. Contudo, a literatura adverte para os perigos da aplicação desta técnica de forma isolada em cenários críticos. Turpin et al. (2023) provaram empiricamente que os LLMs geram frequentemente explicações não-fielis, produzindo justificações altamente persuasivas para decisões logísticas incorretas, o que induz o utilizador numa perigosa confiança indevida na máquina.

É neste ponto que a explicabilidade se cruza obrigatoriamente com a validação neuro-simbólica. Para atingir o que Bansal et al. (2019) definem como «Calibração de Confiança» o estado cognitivo ótimo onde o utilizador compreende exatamente os limites do sistema e confia na sua decisão, a justificação textual gerada tem de ser o reflexo exato de uma validação determinística subjacente. Ao integrar estes princípios, o *TravelMaster AI* não se limita a propor uma nova rota, expõe de forma transparente a restrição física que o motivou, fundindo a precisão matemática com a explicabilidade exigida em contextos de elevado atrito logístico.

Para mitigar as limitações lógicas da IA Generativa, a investigação recente redireciona o foco para a orquestração. Rasal e Hauer (2024) exploram a utilização de LLMs não como executores finais, mas como coordenadores de sistemas complexos. Nesta arquitetura, o modelo linguístico atua como um raciocinador central, responsável por decompor pedidos abstratos em tarefas técnicas específicas, delegando a execução a módulos especializados.

A fundação teórica para esta capacidade de orquestração assenta no emergente domínio dos agentes autónomos baseados em LLM. Numa revisão sistemática pioneira sobre este paradigma, L. Wang, Ma et al. (2024) formalizam uma arquitetura unificada onde o modelo de linguagem deixa de operar como um mero gerador probabilístico de texto para assumir o

papel de cérebro coordenador. Nesta estrutura avançada, o agente é dotado de módulos interdependentes de Perfilamento, Memória, Planeamento e Ação, o que lhe permite interagir com o ambiente externo através da orquestração de ferramentas.

No contexto de tarefas logísticas complexas, o módulo de planeamento revela-se o componente mais crítico. Para ultrapassar as limitações de raciocínio direto iterativo, a literatura recente evidencia a eficácia do paradigma *Plan-and-Execute*, introduzido por L. Wang, W. Xu et al. (2023). Esta arquitetura separa explicitamente a formulação da intenção da sua execução tática, o LLM começa por decompor uma disrupção severa num plano de sub-tarefas sequenciais e, subsequentemente, invoca as ferramentas necessárias para resolver cada passo isoladamente, também demonstraram que esta delegação processual reduz substancialmente os erros típicos de cálculo e omissão lógica inerentes aos modelos generativos puros.

A integração destas arquiteturas de coordenação agêntica constitui o vetor que distancia soluções como o *TravelMaster AI* dos sistemas de recomendação passivos ou simples *chatbots* turísticos. O protótipo desenvolvido materializa a premissa de um *memory-aware tool-using agent* (L. Wang, Ma et al. 2024): o plano abstrato de contornar um atraso multimodal é traduzido em ciclos de execução contínua, onde a validação matemática e a reação operacional decorrem em sintonia fina sob a supervisão do modelo de linguagem.

Esta abordagem de Sistema Multi-Agente (MAS) é fundamental para a arquitetura proposta. Ao separar a compreensão da linguagem (feita pelo LLM) da execução lógica (feita por agentes de API), o sistema consegue harmonizar serviços heterogêneos com precisão. Este modelo alinha-se com a definição formal de agente autónomo de Russell e Norvig (2021), permitindo ao *TravelMaster AI* perceber o ambiente através de dados reais e atuar sobre ele de forma determinística.

A plataforma INDIANA, proposta por Manos et al. (2024), representa o estado da arte na personalização contextual. O sistema utiliza variáveis dinâmicas, como a localização, meteorologia e estado emocional do utilizador, para adaptar sugestões de atividades em tempo real. Esta abordagem valida tecnicamente a importância do contexto na relevância das recomendações turísticas.

No entanto, o foco do INDIANA restringe-se à componente de lazer e inspiração. O sistema não abrange a dimensão crítica da execução logística, nem apresenta soluções para monitorizar a viabilidade operacional da viagem face a imprevistos. Ao contrário do INDIANA, que recomenda atividades com base em contexto emocional, a arquitetura do *TravelMaster AI* foca-se na integridade da cadeia de transporte, validando continuamente se as conexões logísticas (horários, tempos de embarque) permanecem exequíveis face a perturbações externas. O sistema diferencia-se, assim, ao transitar da recomendação de experiências para a assistência ativa na mobilidade e gestão de disrupções.

Por fim, a relevância destes sistemas estende-se à gestão de destinos. Khatun (2024) discute a integração de tecnologias digitais no contexto de destinos turísticos inteligentes, em articulação com o ecossistema mais amplo das cidades inteligentes. A autora mostra que a automação não beneficia apenas o turista, mas pode melhorar a eficiência global dos fluxos turísticos. Ao otimizar deslocações e reduzir a incerteza logística, os agentes inteligentes contribuem para a sustentabilidade operacional dos destinos. Esta perspetiva valida a utilidade social do projeto, ao demonstrar como a redução da fricção individual pode gerar eficiência coletiva.

Em síntese, a literatura confirma a maturidade dos componentes tecnológicos necessários: existem motores de recomendação robustos, APIs de dados operacionais e modelos de IA Generativa avançados. Contudo, persiste uma lacuna na orquestração destes elementos. A inovação não reside na invenção de novas tecnologias, mas na sua integração num agente de execução coeso. A oportunidade de investigação clara é a transição do paradigma de «apoio ao planeamento» para a «monitorização autónoma». O objetivo final é eliminar a dependência da microgestão manual, oferecendo um sistema capaz de navegar a complexidade da viagem em tempo real.

2.3 Análise Comparativa de Soluções

Para tornar inequívoca a diferenciação científica deste trabalho face às abordagens comerciais e académicas disponíveis, apresenta-se uma análise comparativa sistematizada. O objetivo desta síntese é demonstrar que, embora o mercado ofereça soluções maduras para as fases preliminares do turismo, existe um vazio tecnológico no que concerne à assistência inteligente e proativa durante a execução física da viagem.

A Tabela 2.1 avalia as plataformas representativas do estado da arte com base nos requisitos operacionais críticos discutidos ao longo deste capítulo.

Tabela 2.1: Análise comparativa do *TravelMaster AI* face às soluções existentes.

Solução	Plan.	Res.	Mon. T.R.	Repl. Aut.	Expl.	Val. Lóg.	Dep. LLM	Assis. Exec.	Limitações Principais
Agregadores (Ex: <i>Triplt</i>)	Sim	Sim	Não	Não	Não	Não	Não	Não	Totalmente passivo; funciona como mero repositório documental, exigindo microgestão manual do utilizador.
Ecosistemas (Ex: <i>Google Travel</i>)	Sim	Sim	Parcial	Não	Não	Não	Não	Não	Centraliza alertas informativos isolados, mas cessa a automação na execução, não gerando rotas alternativas.
Assistentes LLM (Ex: <i>Expedia AI</i>)	Sim	Parcial	Não	Não	Parcial	Não	Sim	Não	Elevada taxa de alucinações logísticas (≈ 99.4% de falha); total incapacidade de validar restrições de tempo rígidas.
<i>TravelMaster AI</i>	Não	Não	Sim	Sim	Sim	Sim	Não	Sim	Protótipo laboratorial focado na integridade da cadeia de transporte multimodal; operação não transaccional.

Legenda: Plan.: Planeamento; Res.: Reserva; Mon. T.R.: Monitorização em Tempo Real; Repl. Aut.: Replaneamento Autónomo; Expl.: Explicabilidade; Val. Lóg.: Validação Lógica Determinística; Dep. LLM: Dependência Exclusiva de LLM; Assis. Exec.: Assistência Ativa durante a Execução.

O mapeamento exposto na Tabela 2.1 delimita com clareza a originalidade desta dissertação. O *TravelMaster AI* não visa competir no espaço saturado do planeamento de itinerários ou da facilitação transaccional de reservas. A sua contribuição científica foca-se estritamente na intersecção dos vetores de monitorização, replaneamento automático e validação neuro-simbólica. Ao transferir a carga de cálculo intermodal e a verificação de restrições temporais para o domínio determinístico do sistema, o artefacto resolve a vulnerabilidade crítica dos assistentes generativos puros, preenchendo a lacuna de assistência operacional na execução da viagem.

Tendo sido consolidada a identificação desta lacuna fundacional, e comprovada a necessidade de transitar de assistentes puramente generativos para um modelo híbrido de validação neuro-simbólica, torna-se imperativo estruturar o processo de desenvolvimento da solução. O capítulo seguinte detalha a Metodologia de Investigação adotada, alicerçada no paradigma

(DSR), e define formalmente as Questões de Investigação, as salvaguardas éticas e as limitações que orientam a construção rigorosa do artefacto.

Capítulo 3

Metodologia de Investigação

Este capítulo define a estratégia científica que suporta o desenvolvimento da dissertação. Ao longo das próximas secções, detalham-se as fases do processo de investigação, enunciam-se as Questões de Investigação (QI) que orientam o estudo, e definem-se as considerações éticas e as limitações de âmbito que balizam o desenvolvimento laboratorial do protótipo.

Para alcançar os objetivos definidos no Capítulo 1 e responder ao problema da fragmentação de ferramentas e sobrecarga cognitiva durante a execução da viagem, este projeto adota a metodologia *Design Science Research* (DSR). Esta abordagem foca-se na conceção, construção e avaliação de um artefacto tecnológico, o protótipo *TravelMaster AI*, como meio principal para resolver um problema prático e, simultaneamente, contribuir para a base de conhecimento académico.

A justificação para a adoção da DSR em detrimento de uma abordagem metodológica puramente experimental ou observacional prende-se com a natureza do problema. Uma investigação experimental clássica pressupõe a existência de sistemas ou ferramentas maduras sobre as quais se testam hipóteses. Contudo, conforme demonstrado na revisão de literatura, o mercado carece de um sistema que unifique a monitorização da execução da viagem e a validação neuro-simbólica em tempo real. Consequentemente, a produção de novo conhecimento científico neste domínio exige inequivocamente a criação do seu próprio objeto de estudo, a conceção, instanciação e avaliação de um novo artefacto tecnológico.

Para garantir que a construção deste artefacto transcende o mero desenvolvimento de *software* e obedece a um formalismo científico rigoroso, a investigação foi estruturada em torno dos três ciclos fundamentais inerentes à DSR, propostos por Hevner et al. (2004):

- **Ciclo de Relevância:** Assegura a ligação entre o ambiente de aplicação e o projeto de investigação. Neste trabalho, a relevância é ditada pela dor real do passageiro (o ambiente operacional) caracterizada pela sobrecarga cognitiva e a fragmentação de interfaces durante disrupções de voos, definindo assim os requisitos estritos de utilidade que o artefacto tem obrigatoriamente de satisfazer.
- **Ciclo de Rigor:** Garante a validade científica da investigação através da sua ancoragem à base de conhecimento metodológico e tecnológico existente. A arquitetura do *TravelMaster AI* foi suportada pelas fundações teóricas de Agentes Autónomos, constrangimentos determinísticos do TTDP e avaliação analítica segundo o modelo Keystroke-Level Model (KLM), assegurando que a solução é uma inovação fundamentada e não uma mera replicação.
- **Ciclo de Design:** Constitui o núcleo empírico da investigação, operando como um circuito contínuo de construção e avaliação laboratorial. Foi a aplicação rigorosa deste

ciclo iterativo que permitiu refinar o protótipo, desde a integração inicial das APIs simuladas até à consolidação da arquitetura neuro-simbólica final, garantindo que as sucessivas avaliações de latência e lógica retroalimentavam o processo de desenvolvimento.

Para operacionalizar estes três ciclos conceptuais na prática, o processo de trabalho foi estruturado numa sequência lógica de fases de investigação (ilustradas na Figura 3.1), adoptando o processo iterativo de Peffers et al. (2007). Estas fases alinham-se diretamente com os entregáveis do projeto, cada etapa constrói sobre os resultados da anterior, garantindo uma transição fluida desde a análise teórica do problema (Ciclos de Relevância e Rigor) até à construção e validação empírica do agente (Ciclo de Design).

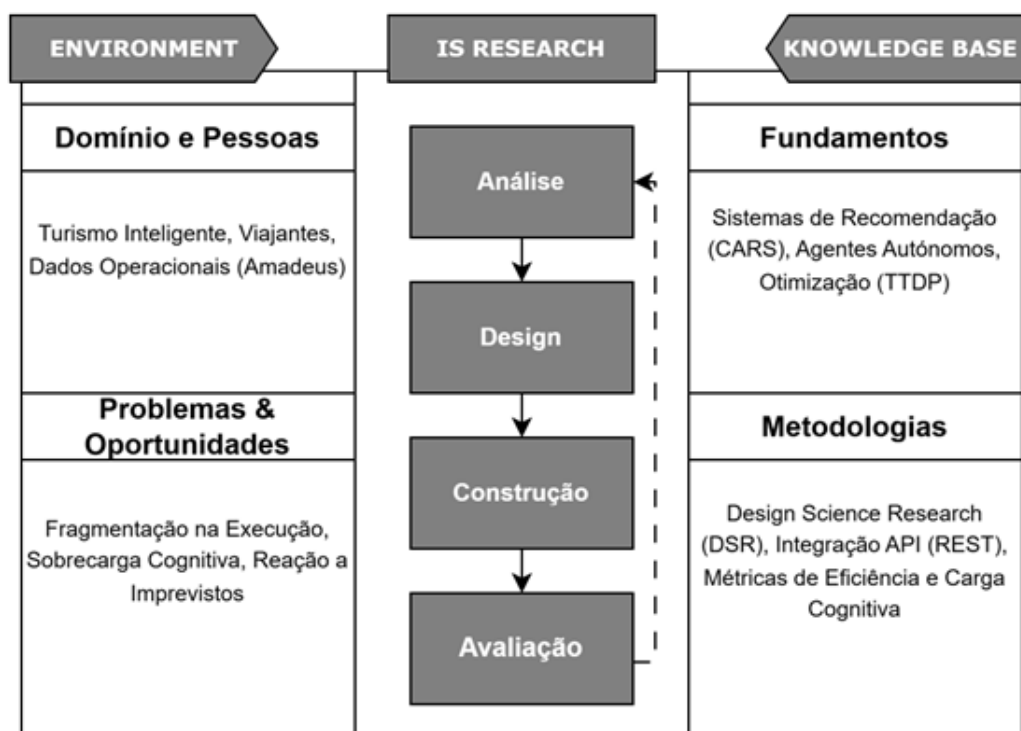


Figura 3.1: Instanciação do modelo de investigação (DSR) aplicado ao projeto *TravelMaster AI*. O diagrama ilustra os ciclos de Relevância, Rigor e Desenho que estruturam a metodologia. Adaptado de Hevner et al. (2004).

3.1 Fases da Metodologia de Investigação

O desenvolvimento do artefacto segue o ciclo iterativo do Processo de Investigação em DSR proposto por Peffers et al. (2007), estruturado nas seguintes etapas fundamentais para garantir o rigor e a relevância da solução:

1. **Interpretação do Problema (Análise de Requisitos):** A fase inicial foca-se na identificação das fontes de sobrecarga cognitiva durante a viagem. O objetivo é traduzir o problema conceptual, a fragmentação de dados entre múltiplas aplicações, numa especificação clara de requisitos funcionais. Definiu-se o fluxo de informação necessário para que o agente possa monitorizar a viagem sem intervenção manual.

2. **Sistematização do Conhecimento (Estado da Arte):** Esta fase, abordada no Capítulo 2, fundamentou o projeto na literatura de «Turismo Inteligente» e Sistemas Multi-Agente. A análise validou a escolha de uma arquitetura baseada em orquestração de LLM (Rasal e Hauer 2024) como resposta às limitações de lógica dos modelos generativos puros.
3. **Avaliação de Abordagens (Decisões de Arquitetura):** Nesta etapa, definiu-se a estratégia de implementação para o ecossistema de dados. Devido à descontinuação do acesso público a infraestruturas de dados aéreos em tempo real durante o desenvolvimento do projeto, a metodologia adotou uma abordagem de simulação rigorosa para validar o artefacto:
 - **Gateway Simulado:** Desenvolvimento de um injetor de eventos capaz de simular respostas e atrasos de voo com a mesma estrutura de dados e latência de uma Application Programming Interface (API) de produção (como a Amadeus ou Sabre). Esta abordagem garante o realismo na monitorização e o rigor dos testes de lógica do motor TTDP.
 - **Serviços de Remediação Simulados:** Modelação de respostas para serviços complementares, permitindo testar a capacidade de replaneamento e a resiliência técnica do agente em cenários de falha controlada ou dados incompletos.
4. **Conceção da Solução (Desenho do TravelMaster AI):** Com base nos requisitos, foi concebida a arquitetura modular ilustrada na Figura 5.1. O sistema segue um padrão de micro-serviços orquestrados, composto pelos seguintes módulos fundamentais:
 - **Orquestrador de Agentes (LLM Core):** O componente central de interação. Não realiza cálculos complexos, mas interpreta a linguagem natural do utilizador e gere o fluxo de diálogo. É responsável por transformar alertas técnicos (ex: JSON de atraso) em notificações úteis e facilmente compreensíveis para o viajante.
 - **Gestor de APIs:** Módulo de abstração que centraliza a comunicação externa. A sua função é normalizar os dados provenientes dos serviços de simulação, entregando ao sistema um formato de dados unificado.
 - **Monitor de Estado e Assistência (Monitoring Service):** O motor de execução do *TravelMaster AI*. Este serviço opera em segundo plano, consultando periodicamente o *gateway* simulado para detetar alterações críticas simuladas, tais como mudanças de porta ou atrasos. Cruza esta informação com o estado atual do utilizador para validar se o tempo de deslocação até ao embarque continua viável e, ao detetar uma anomalia, aciona proativamente o Orquestrador.
 - **Motor de Recomendação (Contextual):** Foca-se na resolução de problemas imediatos. Ao receber um alerta do Monitor, este módulo filtra rapidamente alternativas viáveis de transporte, baseando-se na proximidade geográfica e disponibilidade imediata.
 - **Módulo TTDP (Otimizador Logístico):** Atua como validador de viabilidade. Verifica se as conexões sugeridas respeitam as restrições físicas, garantindo que o agente nunca sugere uma solução impossível de executar.
 - **Módulo XAI (Explicabilidade):** Interface de transparência simples. O seu objetivo é converter as decisões lógicas do sistema em justificações claras, aumentando a confiança do utilizador na automação.

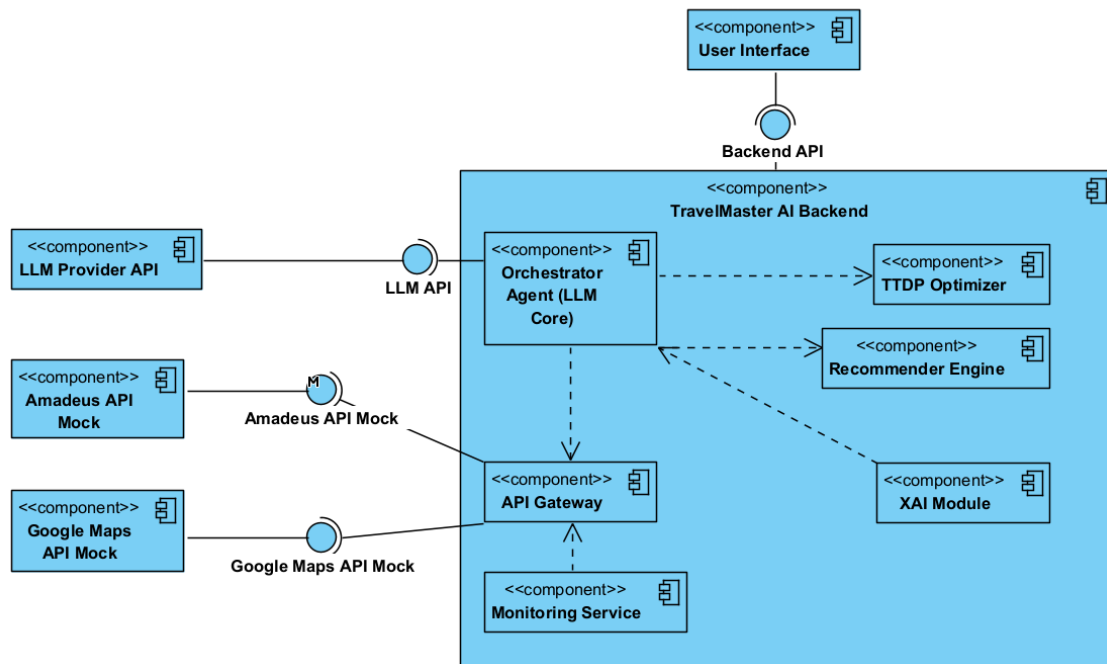


Figura 3.2: Diagrama de Componentes da Arquitetura *TravelMaster AI*, detalhando as dependências entre o Orquestrador, módulos internos e APIs externas.

Para validar a interação dinâmica entre estes componentes num contexto de execução, a Figura 5.2 apresenta o fluxo de sequência para o cenário crítico de «Monitorização e Reação a Imprevistos». O diagrama evidencia o papel do Monitor de Estado como iniciador do processo e a subsequente coordenação do Orquestrador com o Módulo TTDP, garantindo que a alternativa sugerida ao utilizador é não apenas relevante, mas logisticamente viável.

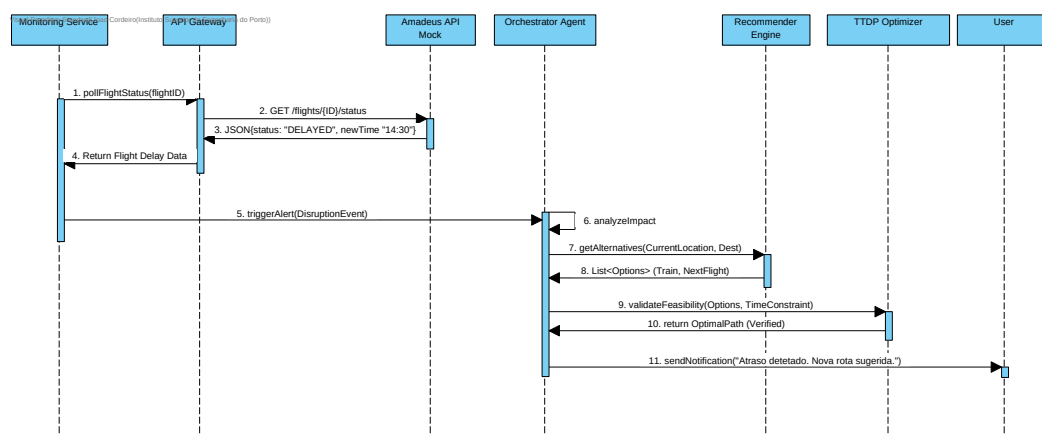


Figura 3.3: Diagrama de Sequência: Fluxo de Detecção Autónoma de Disrupções e Replaneamento. O diagrama ilustra a reação do sistema a um atraso evidenciando o processo de validação lógica (TTDP) antes da notificação ao utilizador.

5. **Construção da Solução (Implementação do Protótipo):** Esta fase focou-se na materialização do agente de monitorização. O desenvolvimento priorizou a criação do ciclo de assistência, a capacidade do sistema ler um itinerário, consultar continuamente o estado dos voos através do *Gateway* simulado e disparar alertas autónomos. O objetivo não é criar um produto comercial de venda de viagens, mas sim um artefacto funcional que demonstre tecnicamente a redução da fricção na gestão de imprevistos.
6. **Avaliação da Solução (Protocolo Experimental):** A validação do protótipo foi realizada através de um estudo comparativo. O objetivo é medir o desempenho do *TravelMaster AI* face a uma Condição de Controlo de mercado, definida como a «Gestão Manual Fragmentada» (utilização simultânea da App da Companhia Aérea para ver o voo + Google Maps para ver o tempo + Browser para procurar alternativas).

Cenários de Teste: Para validar a robustez e a inteligência do agente, foram definidos quatro cenários de teste distintos que cobrem diferentes níveis de complexidade:

- **Cenário A (Ingestão e Contexto):** O utilizador fornece um bilhete eletrónico (texto/PDF). O sistema deve extrair corretamente os dados (voo, hora, destino) e iniciar a monitorização automática sem necessidade de configuração manual complexa. Este cenário valida a capacidade de compreensão e configuração do agente.
- **Cenário B (Alerta Operacional):** Simulação de uma alteração menor, como uma mudança de Porta de Embarque. O sistema deve detetar a alteração através do *Gateway* simulado e notificar o utilizador com a localização da nova porta e o tempo estimado para lá chegar. Testa a prontidão de resposta e a utilidade da notificação proativa.
- **Cenário C (Replaneamento Crítico):** Simulação de um evento de «Disrupção Severa» (ex: voo cancelado ou atraso que inviabiliza uma conexão). Este é o teste central do projeto: o agente deve não só alertar, mas apresentar imediatamente uma alternativa viável, validada pelo módulo TTDP.
- **Cenário D (Resiliência Técnica):** Teste de stress onde a API do LLM falha. Avalia-se a capacidade de degradação controlada, garantindo que o agente continua a fornecer informações críticas sobre o voo mesmo quando os serviços acessórios estão indisponíveis.

Métricas de Avaliação: A eficácia do agente não foi medida por vendas, mas pela redução objetiva da complexidade operacional:

- **Eficiência e Carga Cognitiva (Keystroke-Level Model (KLM)):** Comparação quantitativa do número de interações cognitivas e físicas necessárias para resolver o problema no Cenário C (*TravelMaster* vs. Método Manual). A avaliação recorre aos princípios do *Keystroke-Level Model* (Card et al. 1980) para validar analiticamente a redução do esforço do utilizador, substituindo a necessidade de questionários subjetivos.
- **Tempo de Recuperação (Time-to-Recovery):** Tempo total decorrido desde a deteção do problema até à apresentação de uma solução viável ao utilizador.
- **Viabilidade da Solução (Logical Validity):** Taxa de sucesso na geração de planos que respeitam as restrições físicas (ex: o sistema evitou sugerir uma

conexão de 10 minutos entre terminais distantes?), combatendo as alucinações comuns nos LLM.

- **Qualidade da Recomendação (Precision@K):** Precisão das sugestões de reencaminhamento. Adaptando a métrica de Ricci et al. (2015) ao contexto de assistência, avalia-se a «relevância» operacional das top-K alternativas geradas (ex: das 3 opções de transporte sugeridas, quantas resolviam efetivamente o atraso permitindo chegar antes do fecho do embarque?).

3.2 Questões de Investigação

Para orientar o processo metodológico e focar a investigação nos objetivos centrais do projeto, foram formuladas as seguintes Questões de Investigação (QI). Estas questões decompõem o problema da «sobrecarga cognitiva na gestão de imprevistos» em desafios mensuráveis, respondidos através da construção do artefacto *TravelMaster AI*.

- **QI1 (Viabilidade da Monitorização Autónoma):** É tecnicamente viável conceber uma arquitetura de agente que automatize a monitorização do estado da viagem (voos e conexões) em tempo real, eliminando a necessidade de verificação manual constante por parte do utilizador num ambiente de dados fragmentado?
- **QI2 (Capacidade de Reação e Replaneamento):** De que forma podem os LLMs ser integrados com dados de contexto para gerar alternativas de reencaminhamento proativas que sejam relevantes e personalizadas face a uma disrupção severa?
- **QI3 (Fiabilidade e Validação Lógica):** Como pode uma arquitetura híbrida (Interface Generativa + Validação Determinística via TTDP) mitigar as falhas de raciocínio lógico identificadas na literatura (Xie et al. 2024), garantindo que as soluções propostas pelo agente são fisicamente exequíveis e não meras alucinações textuais?
- **QI4 (Eficiência e Redução de Carga Cognitiva):** Em que medida o protótipo *TravelMaster AI* demonstra, através de avaliação empírica, uma redução quantificável na carga cognitiva do viajante (medida por número de interações e tempo de resolução) em cenários de gestão de imprevistos, quando comparado com a utilização manual de ferramentas convencionais?

Para garantir a operacionalização destas questões e a validade dos resultados, foi criada a Matriz de Avaliação (Tabela 3.1), correlacionando cada Questão de Investigação com a métrica específica e o critério de sucesso mínimo para o protótipo.

Tabela 3.1: Matriz de Avaliação: Correlação entre QIs, Métricas e Critérios de Sucesso.

QI	Indicador (Métrica)	Método de Avaliação	Critério de Sucesso (MVP)
QI1 (Monitorização)	Estabilidade das consultas e Latência de Alerta.	Teste de execução contínua (<i>Long-running test</i>); Análise de Logs.	$\geq 95\%$ de ciclos de monitorização sem falha; Latência de notificação $< 30s$ após injeção do evento.
QI2 (Reação)	Qualidade da Recomendação (Precision@K) e Taxa de Viabilidade da 1ª Opção (Top-1 Accuracy).	Cenário de Disrupção Simulado (Cenário C); Validação manual das opções geradas.	A 1ª opção sugerida pelo agente deve ser logisticamente possível e resolver o atraso em 100% dos casos críticos.
QI3 (Lógica)	Taxa de Alucinação e Violação de Restrições.	Teste de Stress Lógico (Módulo TTDP); Comparação com dados de referência.	0% de Alucinações Logísticas: O sistema nunca deve sugerir conexões impossíveis.
QI4 (Eficiência)	Tempo de Recuperação e Carga Cognitiva (Nº Cliques).	Estudo Comparativo: <i>TravelMaster</i> vs. Método Manual Fragmentado.	Redução $\geq 50\%$ no tempo total de resolução do imprevisto e no número de interações necessárias.

Nota sobre os Critérios de Sucesso: Os valores alvo foram definidos com base em *standards* de engenharia e na literatura de Interação Humano-Computador:

- **Latência de Alerta (< 30s):** Assegura a utilidade operacional da notificação, garantindo que o utilizador recebe a informação antes da atualização dos painéis físicos do aeroporto.
- **0% de Alucinações:** Critério rígido imposto pelas conclusões de Xie et al. (2024). Dado que os LLM puros falham, o sucesso do *TravelMaster* depende da capacidade do módulo determinístico (TTDP) de filtrar 100% dos erros lógicos.
- **Eficiência (50%):** A hipótese de melhoria é elevada porque o processo manual exige alternar entre 3 apps (Email, App da Companhia, Google Maps), enquanto o agente centraliza a informação numa única vista, eliminando drasticamente a mudança de contexto.

3.3 Considerações Éticas

O desenvolvimento do *TravelMaster AI*, enquanto agente de monitorização e assistência autónoma, interceta domínios de elevada sensibilidade ética. A natureza do artefacto, que requer acesso contínuo a dados de contexto para automatizar decisões logísticas, exige uma análise rigorosa da conformidade legal e das implicações morais. Esta secção detalha os desafios éticos decorrentes desta monitorização ativa e a abordagem metodológica aplicada, recorrendo à literatura científica para fundamentar as soluções de proteção.

3.3.1 Privacidade e Proteção de Dados

A transição de um sistema de planeamento estático para um agente de execução em tempo real implica o processamento de dados críticos, nomeadamente a Geolocalização do Utilizador e os dados do PNR (*Passenger Name Record*). O cumprimento do Regulamento Geral sobre a Proteção de Dados (RGPD) constitui, portanto, um requisito funcional obrigatório.

A metodologia adota o princípio de «Privacidade desde a Conceção» (*Privacy by Design*), formalizado por Cavoukian (2009). Para mitigar os riscos de vigilância digital, o sistema implementa duas estratégias fundamentais:

- **Consentimento Granular:** Implementação de um fluxo de *opt-in* explícito, onde o utilizador autoriza o acesso à localização apenas durante a janela temporal da viagem ativa.
- **Minimização e Dados Efémeros:** Em resposta aos desafios de segurança em sistemas de recomendação identificados por Ojokoh et al. (2025), o *TravelMaster AI* trata a geolocalização como um dado transitório. As coordenadas GPS são utilizadas exclusivamente pelo módulo TTDP para calcular a viabilidade logística e são descartadas imediatamente após o cálculo, garantindo que não existe criação de um histórico de movimentos do utilizador na base de dados persistente.

3.3.2 Segurança Transacional e Interoperabilidade

A interoperabilidade com a infraestrutura de simulação e os serviços de monitorização introduz riscos técnicos que devem ser mitigados para garantir a confiança no sistema.

Para assegurar a robustez da arquitetura, foram definidas duas linhas de atuação:

1. **Segurança em Trânsito:** Todas as comunicações entre a Aplicação, o gestor de APIs e os serviços de simulação são asseguradas via encriptação TLS 1.3. Adicionalmente, a gestão de credenciais de acesso às APIs seguiu as melhores práticas de gestão de segredos, garantindo que nenhuma chave de acesso é exposta no código-fonte ou no cliente.
2. **Mitigação de Alucinações (Ética da IA):** Do ponto de vista ético, o maior risco de um agente de viagens baseado em LLM é a geração de informação falsa que possa prejudicar a viagem do utilizador. Para responder a este desafio ético, o sistema implementa o módulo determinístico TTDP. Este mecanismo impõe uma verificação lógica rigorosa sobre todas as sugestões geradas pelo modelo de linguagem, assegurando que o utilizador nunca recebe uma recomendação logisticamente impossível.

3.3.3 Transparência e Autonomia Decisória da IA

A natureza proativa do *TravelMaster AI*, capaz de sugerir alterações ao plano de viagem em resposta a eventos externos, levanta questões críticas de transparência. Num contexto de gestão de imprevistos, torna-se imperativo que o utilizador compreenda não apenas o que o sistema sugere, mas *por que* o sugere.

A fase de conceção incorpora princípios de IA Explicável (XAI). Para operacionalizar a transparência e responder aos desafios de confiança identificados por Zhang e X. Chen (2020), o sistema integra um módulo de Geração de Linguagem Natural Estruturada. Este mecanismo determinístico traduz as validações lógicas do algoritmo em texto discursivo e coerente.

Foram implementados dois tipos de explicações dinâmicas:

- **Justificação de Viabilidade:** O sistema explicita os critérios técnicos que tornam uma opção válida (ex: «Esta conexão ferroviária foi selecionada como alternativa prioritária porque, ao contrário do próximo voo disponível, garante a chegada ao centro da cidade antes das 18:00, evitando o trânsito de hora de ponta»).
- **Explicação de Reação (Causalidade):** Em situações de disrupção, o sistema reporta o evento gatilho e a lógica de mitigação (ex: «Detetámos que o voo TP123 está atrasado 90 minutos, o que inviabiliza a sua conexão original em Frankfurt. O itinerário foi atualizado com uma nova rota via Munique, que mantém o horário de chegada final inalterado.»).

Esta abordagem converte a arquitetura opaca do modelo numa estrutura interpretável, permitindo ao utilizador auditar a lógica subjacente às sugestões. Tal mecanismo é fundamental para transitar de uma automação cega para um sistema de suporte à decisão colaborativo, alicerçado na confiança e na clareza algorítmica.

3.3.4 Matriz de Riscos e Mitigação

De forma a operacionalizar as considerações éticas e de segurança, foi elaborada uma matriz de riscos (Tabela 3.2) que sintetiza as ameaças potenciais do sistema e as medidas técnicas de controlo adotadas na arquitetura.

Tabela 3.2: Matriz de Riscos e Mitigação: Ameaças e Estratégias Técnicas.

Risco Identificado	Nível de Impacto	Estratégia de Mitigação (Técnica)
Fuga de Dados de Localização e PNR	Crítico	Minimização de Dados (Ephemeral Data): A geolocalização é tratada como dado transitório (não persistido) e descartada após o cálculo logístico. Dados do PNR encriptados em repouso.
Interceção de Comunicações	Alto	Encriptação em Trânsito: Imposição de protocolo TLS 1.3 com Fixação de Certificado (<i>Certificate Pinning</i>) na comunicação entre a App, o Gestor de APIs e os <i>Gateways</i> de Simulação.
Alucinação Logística da IA	Crítico	Mecanismo de Salvaguarda Lógica (TTDP): Implementação de uma camada de validação determinística. O módulo otimizador verifica matematicamente todas as sugestões do LLM contra restrições físicas antes da apresentação.
Falha de Dependência (Serviços Externos)	Alto	Resiliência: Implementação de mecanismos de Degradação Controlada. Se os serviços de simulação falharem, o sistema mantém a monitorização vital da telemetria do voo ativa.
Opacidade na Decisão (<i>Black Box</i>)	Médio	XAI: O módulo de Explicabilidade traduz os pesos algorítmicos em justificações textuais claras, permitindo ao utilizador auditar o porquê da recomendação.

3.4 Âmbito e Limitações do Protótipo

É fundamental delimitar com precisão o âmbito desta investigação, estabelecendo uma distinção clara entre a visão concetual do sistema *TravelMaster AI* e o artefacto tecnológico construído. A metodologia foca-se na criação de uma Prova de Conceito, cujo objetivo é demonstrar a viabilidade técnica da «Monitorização e Reação Autónoma» (QIs), e não o desenvolvimento de um produto comercial de venda de viagens.

Como definido na fase de Avaliação de Abordagens, o projeto assume limitações intencionais que garantem a sua exequibilidade no contexto de uma dissertação de mestrado:

1. **Interoperabilidade e Dados (Simulação Controlada):** Para validar a capacidade de orquestração (QI3) sem incorrer em custos proibitivos, bloqueios burocráticos ou dependência de serviços comerciais descontinuados, o protótipo adota uma infraestrutura de simulação integral:
 - **Telemetria de Aviação:** A monitorização de voos, horários e estados de terminais ou portas de embarque é realizada através de um *Gateway* simulado. Esta abordagem substitui a dependência direta de APIs comerciais externas, assegurando o controlo absoluto sobre a injeção estruturada de atrasos para validar o tempo de reação e a precisão algorítmica.
 - **Serviços Complementares Simulados:** Os serviços de remediação operam num ambiente de teste fechado. Ao contrário de APIs comerciais que dificultam a reprodução previsível de erros de rede, esta opção metodológica permite a injeção artificial de latência e falhas de serviço, validando os mecanismos de degradação controlada do agente perante indisponibilidade técnica.
2. **CrITÉrios de Aceitação do MVP:** O protótipo é considerado funcional e apto para avaliação se cumprir os seguintes requisitos mínimos de execução:
 - **Ingestão de Contexto:** Capacidade de ler um itinerário de voo e iniciar o processo de monitorização automática.
 - **Deteção de Eventos:** Deteção proativa e bem-sucedida de alterações de estado sem qualquer intervenção humana.
 - **Validação Lógica (TTDP):** Geração automática de um plano de contingência que respeite matematicamente as restrições de tempo e distância.
 - **Notificação Assistida:** Apresentação da solução ao utilizador numa interface simplificada, com a explicação do motivo do reencaminhamento.
3. **Validação e Dados de Utilizador:** A avaliação do sistema (fase 3.1.6) não recorre a testes em campo com viajantes reais em aeroportos, devido à imprevisibilidade e riscos éticos. A validação da eficácia (QI4) foi conduzida através de Cenários de Simulação Controlada, utilizando perfis de utilizador sintéticos e injeção de falhas. Esta abordagem permite medir com precisão o tempo de reação do agente e a viabilidade das rotas sugeridas num ambiente laboratorial reprodutível.

Estas limitações, embora restrinjam o âmbito comercial do protótipo, são metodologicamente necessárias para focar a investigação no seu contributo inovador: a inteligência da recuperação de viagens e a arquitetura de agente autónomo.

Com a metodologia DSR formalmente instanciada, os critérios éticos estabelecidos e o âmbito de desenvolvimento balizado, o alicerce conceptual da investigação encontra-se consolidado. O capítulo seguinte apresenta o Plano de Trabalho adotado, operacionalizando esta metodologia numa cronologia de fases de desenvolvimento para garantir a exequibilidade rigorosa do projeto.

Capítulo 4

Plano de Trabalho

Este capítulo detalha o planeamento temporal e estrutural projetado para a execução do projeto de dissertação, alinhando os entregáveis com a metodologia *Design Science Research* (DSR) proposta no Capítulo 3.

O planeamento foi estruturado para cumprir os objetivos definidos, garantindo tempo adequado para a implementação iterativa, testes rigorosos e uma validação experimental consolidada. O cronograma macro das tarefas previstas encontra-se ilustrado no Diagrama de Gantt (Figura 4.1).

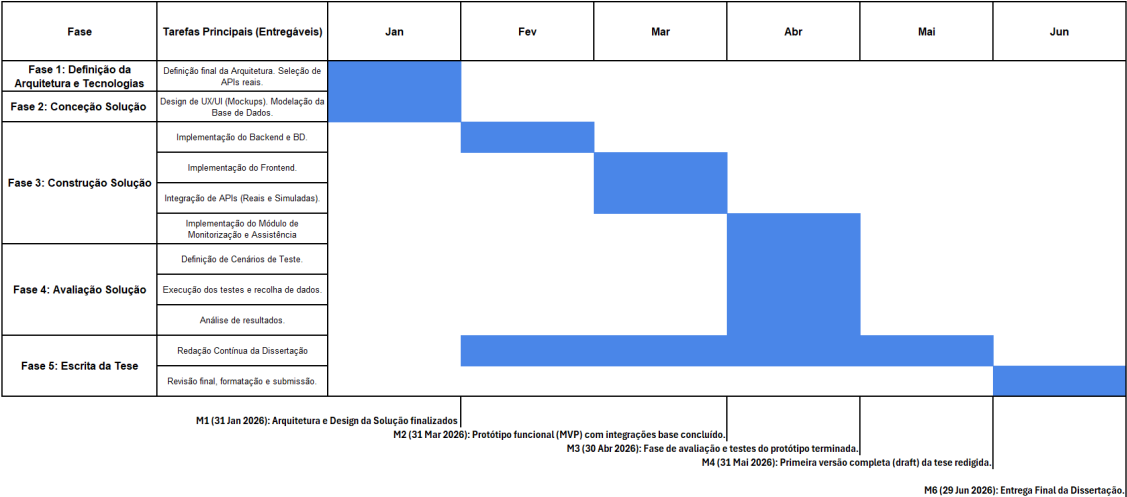


Figura 4.1: Diagrama de Gantt: Cronograma previewal para o desenvolvimento da Dissertação.

4.1 Fases de Desenvolvimento

O plano de trabalhos para o semestre foi estruturado em cinco fases sequenciais, desenhadas para garantir a execução iterativa do projeto e refletindo o ciclo de vida completo do sistema:

Fase 1 - Definição da Arquitetura e Tecnologias: Esta etapa inicial focou-se na consolidação das escolhas tecnológicas e na definição final da arquitetura do sistema. Foi realizada a seleção definitiva das APIs simuladas, procedendo-se aos testes preliminares de conectividade e autenticação para mitigar riscos técnicos logo no arranque do desenvolvimento.

- Fase 2 - Conceção da Solução:** Antes de iniciar a implementação do código-fonte, esta fase dedicou-se ao planeamento visual e estrutural do artefacto. As tarefas principais incluíram o design da Experiência do Utilizador (UX) e Interface do Utilizador (UI), com a criação de protótipos visuais para a interface do assistente, bem como a modelação rigorosa da Base de Dados para suportar os fluxos de informação do viajante.
- Fase 3 - Construção da Solução:** Correspondeu ao núcleo de desenvolvimento de *software* do projeto. O trabalho foi dividido entre a implementação do *Backend* e do *Frontend*, culminando na integração das APIs simuladas e dos serviços de orquestração do (LLM). O objetivo central foi operacionalizar o motor de inferência, garantindo que o sistema conseguisse detetar disrupções e sugerir alternativas logicamente válidas.
- Fase 4 - Avaliação da Solução:** Após o desenvolvimento, o protótipo foi submetido à validação experimental definida metodologicamente. Esta fase englobou a definição detalhada dos cenários de teste (incluindo disrupções terrestres e aéreas), a execução dos mesmos para recolha de métricas de desempenho (latência e eficácia do *Tool Calling*), e a análise crítica dos resultados para validar a mitigação de carga cognitiva.
- Fase 5 - Escrita e Revisão da Dissertação:** Esta foi uma tarefa de carácter transversal que decorreu ao longo de todo o semestre. Focou-se na redação contínua dos capítulos, garantindo a documentação técnica do artefacto, e culminou na revisão integral, formatação final do documento e formulação das conclusões e propostas de trabalho futuro.

Capítulo 5

Arquitetura e Conceção da Solução

Este capítulo detalha a arquitetura do sistema TravelMaster AI, focando-se nos princípios de engenharia e nas decisões tecnológicas adotadas para a construção do artefacto. Em estrito alinhamento com a metodologia *Design Science Research* (DSR), a arquitetura proposta visa demonstrar a viabilidade técnica de um orquestrador autónomo de disrupções em viagens, priorizando a robustez lógica do fluxo de decisão em detrimento da infraestrutura comercial. Ao longo das secções seguintes, mapeia-se a transição da teoria estrutural para o desenho prático, explorando a arquitetura híbrida neuro-simbólica, a matriz tecnológica adotada e os padrões agênticos que suportam a lógica de recomendação do sistema.

5.1 Princípios Arquiteturais

A conceção do *TravelMaster AI* foi orientada por quatro princípios fundamentais de engenharia de software, garantindo que o protótipo Minimum Viable Product (MVP) responde diretamente ao problema da sobrecarga cognitiva sem introduzir complexidade desnecessária:

- **Keep It Simple, Stupid (KISS) e Foco DSR:** O desenvolvimento centrou-se exclusivamente em provar a viabilidade da orquestração autónoma. Foram intencionalmente excluídos elementos periféricos não nucleares para a investigação, tais como bases de dados de produção, sistemas de autenticação de utilizadores ou ligações a API pagas, seguindo os princípios de construção de artefactos em *Design Science Research* (DSR) (Hevner et al. 2004).
- **Single Responsibility Principle (SRP):** A base de código foi rigorosamente modularizada. Cada componente do sistema (Gateways, Motor Matemático, Heurística, Orquestrador e Interface) possui uma responsabilidade única e isolada. A literatura recente sugere que, em sistemas baseados em LLMs, a adoção de princípios clássicos de engenharia de software e arquiteturas modulares é essencial para conter a natureza probabilística do modelo e preservar a previsibilidade da aplicação (Hou et al. 2024; Rao et al. 2025). Esta arquitetura desacoplada facilita o teste rigoroso dos fluxos lógicos e garante que o motor matemático possa evoluir independentemente do modelo de linguagem.
- **Mitigação de Risco (Contenção de Alucinações Logísticas):** O princípio mais crítico da solução é a prevenção ativa de “alucinações logísticas”. Estudos recentes demonstram que os agentes de linguagem natural apresentam taxas de sucesso residuais quando submetidos a restrições estritas de planeamento e viabilidade no mundo real (Xie et al. 2024). Para mitigar esta falha, o sistema bloqueia a capacidade do LLM

de gerar rotas ou inventar horários de forma isolada, obrigando-o a atuar unicamente sobre os resultados filtrados e validados por um motor determinístico.

- **Transparência e Resiliência de Confiança (XAI):** Dado que o sistema atua em cenários de elevada incerteza, a explicabilidade é incorporada como um pilar central para mitigar a aversão algorítmica (Fleiß et al. 2024). A literatura indica que, no domínio do turismo, a confiança é um fator multifacetado que depende da clareza sobre o processo de decisão do agente (Park e Kim 2020). Ao converter cálculos lógicos em justificações textuais, o TravelMaster AI não só melhora a transparência percebida, como visa aumentar a resiliência da confiança do utilizador perante falhas externas ou imprevistos operacionais (Govea et al. 2024; M. Xu e Y. Wang 2024).

5.2 Arquitetura Híbrida de Orquestração

Ao contrário das soluções generalistas existentes no mercado, o TravelMaster AI não opera sob o paradigma de um mero *chatbot* conversacional passivo. O artefacto assume a forma de um Orquestrador de Microsserviços Inteligente, alinhando-se com abordagens recentes de decomposição de problemas complexos (Rasal e Hauer 2024).

A arquitetura híbrida separa a interação em linguagem natural da validação factual e resolução do Tourist Trip Design Problem (TTDP). O Orquestrador LLM atua como o controlador central que:

1. Monitoriza ativamente as APIs de transporte (voos e trânsito terrestre).
2. Invoca dinamicamente ferramentas Python (*Tool Calling*) perante a deteção de anomalias.
3. Delega os cálculos de viabilidade logística ao motor determinístico antes de formatar a recomendação final, assumindo a gestão completa do problema antes de notificar o utilizador.

5.3 Stack Tecnológica e Ferramentas

As decisões tecnológicas foram tomadas em função dos objetivos de eficiência e validação do protótipo, privilegiando a velocidade de execução em tempo real e a flexibilidade de desenvolvimento:

- **Orquestrador LLM (Google Gemini 2.5 Flash):** A integração de ferramentas constitui atualmente uma estratégia dominante na literatura para ultrapassar as limitações do conhecimento paramétrico estático, permitindo aos modelos interagir ativamente com o mundo real e aceder a informação atualizada via APIs (S. Chen et al. 2024; Yang et al. 2024). O Gemini 2.5 Flash foi selecionado como o “cérebro” do sistema precisamente devido à sua excecional latência reduzida e precisão nestas tarefas de invocação. Num sistema orquestrado que exige múltiplas validações em *background* antes da resposta final ao utilizador, este modelo assegura um processamento em tempo real (tempos de resposta totais entre 5 a 14 segundos em cenários complexos) mantendo uma elevada eficiência de custos e uma capacidade robusta de delegar decisões ao motor matemático.
- **Backend e Interface (Python e Streamlit):** A escolha da linguagem Python fundamenta-se na sua capacidade nativa para lidar com integração de APIs, extração de dados e

ecossistema de Inteligência Artificial estruturada (via *Pydantic*). Para a camada visual, a adoção da *framework* *Streamlit* permitiu materializar um *frontend* interativo e um painel de controlo de simulações de forma ágil, eliminando a necessidade de gerir pilhas tecnológicas complexas de desenvolvimento Web (HyperText Markup Language (HTML)/Cascading Style Sheets (CSS)/JavaScript (JS)) e respeitando o princípio KISS.

- **Isolamento via Simuladores de Telemetria:** O acesso aos dados de companhias aéreas e serviços de mobilidade terrestre foi implementado através de simuladores. No contexto da metodologia DSR, a avaliação de artefactos em ambientes simulados constitui um método de validação experimental rigoroso (Hevner et al. 2004). Esta decisão metodológica permite isolar o artefacto de falhas de rede externas e contornar os custos elevados de plataformas proprietárias (como Global Distribution System (GDS) Amadeus ou Google Maps). Crucialmente, a introdução de injeção de latência realista nos *Mocks* garante a capacidade de criar cenários reproduzíveis de múltiplas disrupções simultâneas de forma controlada, para efeitos de avaliação técnica do sistema.

5.4 Arquitetura de Implementação e Componentes

Transitando da abstração conceptual (apresentada no Capítulo 3) para a instanciação física do artefacto, a Figura 5.1 ilustra o Diagrama de Componentes UML do sistema TravelMaster AI. Este modelo evidencia a aplicação rigorosa do *Single Responsibility Principle* (SRP) na organização da base de código.

O componente central, `llm_agent.py`, atua como Orquestrador, isolando a interação em linguagem natural (via Google Gemini API) da lógica de negócio e persistência de dados (`sqlite.db`). A validação da viabilidade das rotas é estritamente delegada ao módulo `ttdp_logic.py`, prevenindo a geração de soluções irrealistas. A interface visual, materializada no componente `ui.py`, interage exclusivamente com as rotinas de ingestão (`ingestion.py`) e com o orquestrador, garantindo o desacoplamento do *frontend*.

De salientar uma adaptação arquitetural crítica: face à descontinuação do ambiente *Sandbox* público por parte do GDS Amadeus durante a fase de desenvolvimento, o componente `amadeus_api.py` foi refatorizado num *Gateway* simulado. Em estrito alinhamento com a metodologia DSR, esta abstração simula as respostas API em formato JSON, permitindo a injeção artificial de latência e a simulação de disrupções severas. Esta decisão assegura que a avaliação do artefacto assenta em testes de *stress* reproduzíveis, isolando a validação da orquestração de potenciais falhas de rede de terceiros.

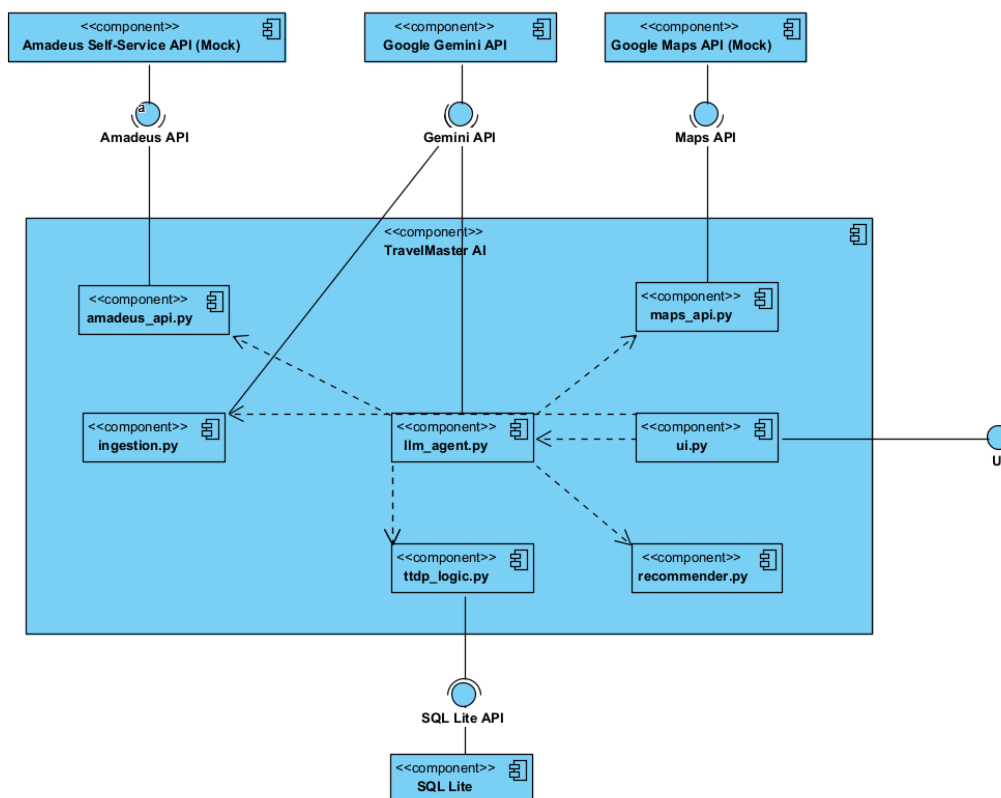


Figura 5.1: Diagrama de Componentes UML da Arquitetura do TravelMaster AI.

5.5 Dinâmica de Orquestração e Tool Calling

Para além da separação física de componentes, o valor nuclear da arquitetura reside na sua dinâmica de orquestração neuro-simbólica. A Figura 5.2 detalha o Diagrama de Sequência UML para o cenário crítico de deteção de disrupções, demonstrando como o sistema interceta e resolve anomalias operacionais antes de notificar o utilizador.

O fluxo ilustra o ciclo fechado de *Tool Calling*. Quando o orquestrador estocástico (Gemini) é alertado para um imprevisto (ex: atraso de voo acionado no *frontend*), este não gera uma resposta final imediata. Em vez disso, formula um pedido estruturado para invocar a ferramenta de recomendação (`recommender.py`). É nesta fase que ocorre a validação crítica: o sistema itera sobre a lista bruta de voos alternativos e delega a verificação de cada opção ao motor determinístico (`ttdp_logic.py`).

Apenas após o cálculo matemático da viabilidade temporal em aeroporto e a receção da respetiva cadeia explicativa (módulo XAI), é que o Orquestrador sintetiza a solução de reencaminhamento. Este mecanismo de *callback* obrigatório atua como um disjuntor lógico, contribuindo ativamente para a prevenção de alucinações logísticas no planeamento do novo itinerário.

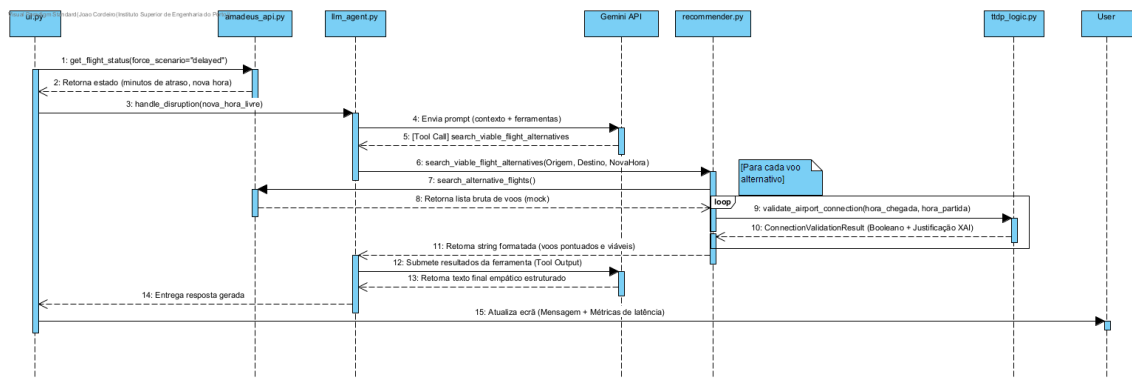


Figura 5.2: Diagrama de Sequência UML detalhando o ciclo de orquestração, o *Tool Calling* e a iteração de validação lógica face a uma interrupção.

5.6 Padrões de Desenho e Decisões Arquiteturais

Para garantir a robustez e a manutenibilidade do artefacto, o desenvolvimento do TravelMaster AI não se limitou à integração isolada de modelos de linguagem, mas seguiu uma disciplina rigorosa baseada em padrões de desenho. A arquitetura inspira-se tanto na tradição clássica da engenharia de software (Gamma et al. 1994) como em abordagens arquiteturais contemporâneas para sistemas neuro-simbólicos baseados em LLM (Boer et al. 2025) e ecossistemas agênticos (Milosevic e Rabhi 2026).

5.6.1 Padrões Clássicos e Isolamento para DSR

A gestão de integrações com serviços externos (Amadeus e Google Maps) foi implementada através do padrão Facade. Os módulos `amadeus_api.py` e `maps_api.py` atuam como uma interface simplificada que oculta a complexidade das chamadas HTTP, encapsula o tratamento de respostas JSON e uniformiza eventuais falhas perante o restante sistema (Gamma et al. 1994).

Em estrito alinhamento com a metodologia DSR, o isolamento destes componentes externos foi garantido através da utilização do padrão de Mock Objects. A abstração dos serviços de aviação através de simuladores controlados permitiu eliminar a dependência de APIs em ambiente de produção (Spadini et al. 2019). Esta adoção viabiliza o isolamento de testes, facilitando a injeção artificial de cenários de interrupção e garantindo que a capacidade de orquestração matemática do sistema possa ser validada de forma reproduzível.

A nível do motor analítico (TTDP), utilizou-se uma variação do padrão Strategy. Consoante o contexto do incidente detetado, o sistema seleciona e aplica dinamicamente a estratégia de validação mais adequada, o que promove um elevado grau de encapsulamento e modularidade (Gamma et al. 1994).

5.6.2 Arquitetura Neuro-Simbólica e Padrões Agênticos

Do ponto de vista da Inteligência Artificial, o TravelMaster AI materializa um padrão arquitetural emergente classificado como um Sistema Neuro-Simbólico. Conforme formalizado por (Boer et al. 2025), a convergência destas tecnologias passa por integrar a flexibilidade dos LLM (modelos estatísticos gerativos) com a precisão de ferramentas determinísticas (Boer et al. 2025).

Neste desenho arquitetural, o orquestrador estocástico (`llm_agent.py`) atua puramente como uma interface de linguagem fluente. O modelo compreende a linguagem natural submetida pelo utilizador e traduz a intenção para consultas formais, mas a inferência computacional crítica e a resolução do problema espacial/temporal são obrigatoriamente delegadas aos módulos simbólicos (`ttdp_logic.py` e `recommender.py`) (Boer et al. 2025).

Para garantir que a comunicação entre o modelo de linguagem e as ferramentas tradicionais obedece a limites rigorosos, o sistema recorre a padrões de *Agentic AI* que implementam governança algorítmica e protocolos de colaboração verificáveis (Milosevic e Rabhi 2026). Ao exigir que qualquer alternativa seja testada pelos algoritmos Python através de *Tool Calling*, a arquitetura impõe uma barreira inultrapassável sobre as capacidades probabilísticas do LLM. O motor semântico devolve os resultados da verificação novamente ao LLM, que atua mais uma vez como interface fluente na formatação de saída da resposta estruturada, colmatando assim a problemática das alucinações e estabelecendo um fluxo fechado, verificável e neuro-simbólico.

5.7 Módulo de Recomendação e Mitigação de Fricção

O planeamento de itinerários de contingência constitui um problema complexo de otimização combinatória com múltiplos critérios concorrentes. A arquitetura do TravelMaster AI resolve esta complexidade separando o processamento em duas camadas sequenciais: a satisfação de restrições rígidas e a otimização de restrições flexíveis, um padrão amplamente validado na literatura de Sistemas de Recomendação (Felfernig et al. 2015; Vansteenwegen et al. 2011).

Enquanto o módulo TTDP atua como disjuntor lógico de restrições rígidas, eliminando conexões fisicamente impossíveis, o módulo `recommender.py` aplica uma heurística de pontuação para ordenar as opções viáveis, priorizando as soluções que minimizam o esforço cognitivo e físico do passageiro.

5.7.1 Lógica de Pontuação Subtrativa

A função de pontuação foi desenhada com base num sistema de penalizações. Todas as alternativas viáveis iniciam o processo com uma pontuação máxima de 100 pontos. O algoritmo subtrai valor a esta pontuação baseando-se em dois eixos de “fricção de viagem”: a fricção espacial e a fricção temporal.

1. **Fricção Espacial (Regra de Mudança de Infraestrutura):** Se o aeroporto de partida do voo alternativo for diferente da infraestrutura atual do passageiro, o sistema aplica uma penalização severa de 40 pontos. A literatura de transportes demonstra que a penalização de transferência, que engloba recolher bagagens, navegar em transportes terrestres desconhecidos e repetir controlos de segurança, gera o maior pico de fadiga e stress percebido pelo utilizador (Wardman 2004). Esta subtração enorme garante que mudanças de aeroporto são atiradas para o fundo da lista, surgindo apenas como último recurso.
2. **Fricção Temporal (Regra de Espera Prolongada):** O algoritmo calcula a diferença entre o tempo livre total do utilizador e o *Minimum Connection Time* (MCT) estritamente necessário. Por cada bloco completo de 30 minutos de espera adicional, o sistema subtrai 1 ponto. Embora a espera gere fadiga cumulativa (Wardman 2004), a

penalização é intencionalmente suave. Uma longa espera no mesmo terminal é invariavelmente preferível ao risco logístico de uma transferência intermodal. Deste modo, voos na mesma infraestrutura são ordenados cronologicamente sem serem suplantados por voos noutros aeroportos.

5.7.2 Integração de XAI via Reasoning Strings

Para além do cálculo numérico, a heurística incorpora o princípio de Transparência e Explicabilidade (XAI). À medida que as penalizações são aplicadas, o algoritmo concatena cadeias de raciocínio no objeto de resposta, tais como *“Voo no mesmo aeroporto. Espera de 105 min.”* ou *“Penalizado por exigir deslocação para aeroporto secundário”*.

Estudos indicam que sistemas de recomendação que fornecem justificações baseadas em características aumentam significativamente a confiança e a probabilidade de aceitação por parte do utilizador (Tintarev e Masthoff 2015). No TravelMaster AI, esta arquitetura é vital para o LLM. Ao receber os resultados ordenados, o Orquestrador consome estas *strings* e converte-as numa justificação empática em linguagem natural (ex: *“Sugiro a Opção 1 pois permite-lhe aguardar no mesmo terminal, evitando deslocações ao aeroporto MFM”*).

Numa simulação prática onde um utilizador retido em Hong Kong (HKG) tenha duas alternativas viáveis no próprio aeroporto e uma terceira a partir de Macau (MFM), o sistema pontua o voo HKG mais próximo com ≈ 99 pontos, o voo mais tardio em HKG com ≈ 88 pontos e a partida por MFM com ≤ 60 pontos. Esta lógica garante que as soluções de re-planeamento recomendadas refletem o máximo conforto logístico possível perante cenários de incerteza.

Tendo sido estabelecidos os alicerces conceptuais da arquitetura neuro-simbólica, a infraestrutura modular e o modelo matemático de mitigação de fricção, o desenho teórico da solução encontra-se plenamente formalizado. O capítulo seguinte avança para a materialização desta estrutura técnica, detalhando o processo prático de Implementação do Artefacto ao nível da construção do código, configuração do orquestrador e programação do motor determinístico.

Capítulo 6

Implementação do Artefacto

Este capítulo descreve a transição do modelo arquitetural abstrato, delineado no Capítulo 5, para a instanciação física do protótipo de software *TravelMaster AI*. Em rigoroso alinhamento com a metodologia *Design Science Research* (DSR), o foco desta secção não é a apresentação exaustiva do código-fonte, mas sim a análise detalhada dos módulos críticos que materializam a orquestração neuro-simbólica. Serão detalhados o ambiente de desenvolvimento, a estrutura do projeto, os mecanismos de ingestão de contexto multimodal e a implementação prática do *Tool Calling* como mitigador de alucinações algorítmicas.

6.1 Ambiente de Desenvolvimento e Stack Tecnológica

Para o desenvolvimento do artefacto, adotou-se uma *stack* tecnológica intencionalmente minimalista. Esta decisão teve como objetivo reduzir a dívida técnica e assegurar que o esforço de engenharia se concentrasse no núcleo de investigação: a orquestração entre o modelo de linguagem e o motor lógico.

6.1.1 Linguagem e Ferramentas Base

O sistema foi inteiramente desenvolvido em Python 3.10+. A escolha desta versão específica deve-se ao seu suporte nativo avançado para tipagem estática, uma característica que, embora opcional na linguagem, provou ser absolutamente crítica para garantir a segurança de tipos na passagem de argumentos entre o Orquestrador e as ferramentas de código determinístico.

O isolamento das dependências foi garantido através da criação de um ambiente virtual nativo (`venv`), assegurando a reprodutibilidade do protótipo em ambientes laboratoriais externos.

6.1.2 Bibliotecas Vitais (Core Dependencies)

A arquitetura assenta num conjunto restrito de bibliotecas externas, cada uma com uma responsabilidade bem definida:

- **google-genai:** O Software Development Kit (SDK) oficial e de última geração da Google, essencial para a comunicação nativa com o modelo Gemini 2.5 Flash. Esta biblioteca foi selecionada por fornecer o suporte mais atualizado a capacidades multimodais (leitura de bilhetes em PDF) e à definição estrita de esquemas para *Tool Calling*.

- **pydantic:** Ferramenta padrão da indústria para validação de dados. É utilizada transversalmente para definir os *Schemas* de domínio (ex: `ConnectionValidationResult`). O `pydantic` garante que os *outputs* do motor matemático são formatados de forma previsível e imutável antes de serem consumidos pelo LLM, mitigando falhas de orquestração estrutural.
- **streamlit:** Adotada como *framework* de *frontend*, permitindo a construção de uma interface gráfica reativa e gestão de sessões utilizando exclusivamente Python. A sua escolha obedece ao princípio Keep It Simple, Stupid (KISS), eliminando a necessidade de desenvolver código não-nuclear ao projeto em HTML, CSS ou JavaScript.
- **Segurança e *Standard Library*:** A injeção de variáveis de ambiente e chaves de API é gerida pela biblioteca `python-dotenv`. O registo de operações e a manipulação cronológica da heurística são inteiramente suportados pelas bibliotecas padrão do Python (`logging`, `datetime`, `time`).

6.1.3 Estrutura de Diretórios

A base de código foi estruturada com o propósito de espelhar fisicamente o *Single Responsibility Principle* (SRP) detalhado no desenho arquitetural. A segregação em diretórios isola configurações, regras de negócio e testes, conforme ilustrado na estrutura abaixo:

```
TravelMaster_AI/
  .env                # Configurações de segurança e API Keys
  data/               # Ficheiros e assets de teste (ex: bilhete.pdf)
  src/                # Código-fonte de Produção (Regras de Negócio)
    ingestion.py      # Extração de dados multimodais
    amadeus_api.py    # Mock Gateway de aviação
    maps_api.py       # Mock Gateway de trânsito terrestre
    ttdp_logic.py     # Motor matemático determinístico (Validador)
    recommender.py    # Motor de pontuação e heurística (Soft Constraints)
    llm_agent.py      # Orquestrador central e Tool Calling
    ui.py             # Interface visual reativa (Streamlit)
  tests/              # Suite de Testes (Isolamento Lógico)
    test_[...].py     # Testes unitários para validação de cada módulo
    test_e2e.py       # Testes de integração End-to-End
```

Esta modularização estrita (`src/` vs. `tests/`) não só previne o acoplamento excessivo, como também viabiliza que o motor determinístico (`ttdp_logic.py`) seja testado isoladamente antes de ser exposto à orquestração estocástica do `llm_agent.py`.

6.2 Ingestão e Estruturação de Contexto

A primeira fase crítica do TravelMaster AI consiste na captura do contexto da viagem (Cenário A). O desafio técnico reside em converter dados não estruturados de um bilhete (PDF ou imagem) em variáveis tipadas e validadas para o motor de decisão.

6.2.1 Tipagem Estrita com Pydantic

A implementação utiliza a biblioteca `pydantic` para estabelecer um contrato de dados. Esta abordagem garante que qualquer informação extraída pelo LLM cumpra a estrutura esperada

antes de ser processada. A Listagem 6.1 apresenta a estrutura nuclear para a representação do bilhete:

```
1 class FlightInfo(BaseModel):
2     flight_number: str = Field(description="Ex: TP123")
3     departure_airport: str = Field(description="IATA de origem")
4     arrival_airport: str = Field(description="IATA de destino")
5     departure_date: str = Field(description="YYYY-MM-DD")
6     departure_time: str = Field(description="HH:MM (24h)")
```

Listing 6.1: Schema Pydantic para validação estrutural do bilhete.

6.2.2 Extração Multimodal via Gemini 2.5 Flash

Diferenciando-se de analisadores sintáticos tradicionais, o sistema utiliza o modelo Gemini 2.5 Flash. A sua capacidade multimodal permite interpretar visualmente o documento, sendo resiliente a variações de estrutura que frequentemente invalidam soluções baseadas em Optical Character Recognition (OCR) convencional.

A lógica de extração tira partido do modo *Structured Output*, onde o *schema* Pydantic é injetado diretamente na configuração da API. Isto força o modelo a atuar como um analisador semântico, garantindo que a resposta é entregue em formato JSON perfeitamente compatível com o domínio da aplicação:

```
1 response = client.models.generate_content(
2     model='gemini-2.5-flash',
3     contents=[types.Part.from_bytes(data=file_bytes, mime_type='
4         application/pdf'),
5         "Extrai os detalhes deste bilhete de aviao."],
6     config=types.GenerateContentConfig(
7         response_mime_type='application/json',
8         response_schema=FlightInfo
9     )
10 )
11 return FlightInfo.model_validate_json(response.text)
```

Listing 6.2: Extração estruturada utilizando o modelo Gemini 2.5 Flash.

Esta arquitetura elimina a necessidade de expressões regulares complexas. Ao acoplar a interpretação visual do Gemini 2.5 Flash com a validação do *pydantic*, o sistema assegura que apenas dados íntegros e tipados alimentam a lógica de replaneamento.

6.3 Construção do Orquestrador e Chamada de Ferramentas

O núcleo inteligente do *TravelMaster AI* não opera sob o paradigma tradicional de um gerador de texto não delimitado. Para atuar no setor do turismo, onde as falhas logísticas carregam custos financeiros e emocionais severos, o sistema eleva o modelo de linguagem à condição de Motor de Raciocínio. Esta transformação exige a subordinação rigorosa das capacidades probabilísticas da Inteligência Artificial a restrições determinísticas, alcançada através da injeção de ferramentas, colmatando as lacunas de raciocínio lógico e factual intrínsecas aos modelos de linguagem puros (Schick et al. 2023).

6.3.1 Enquadramento Comportamental e Mitigação Estocástica

A primeira barreira contra alucinações baseia-se na definição estrita das fronteiras operacionais do modelo, implementada através da Instrução de Sistema.

A configuração abaixo ilustra a imposição de autonomia forçada ao agente:

```

1 self.system_instruction = (
2     "Es o TravelMaster AI, um assistente inteligente e empatico. "
3     "A tua missao e reduzir a sobrecarga cognitiva do utilizador. "
4     "Perante um atraso, NUNCA pecas ao utilizador para procurar
5     alternativas. "
6     "Usa IMEDIATAMENTE a tua ferramenta de pesquisa para encontrar
7     solucoes "
8     "e apresenta-as de forma clara, explicando as escolhas."
9 )

```

Listing 6.3: Definição da Instrução de Sistema para o Orquestrador LLM.

A diretiva imperativa “*NUNCA pecas ao utilizador... Usa IMEDIATAMENTE a tua ferramenta*” atua como vetor principal para cumprir o objetivo de mitigação de carga cognitiva. Em paralelo com esta instrução semântica, o parâmetro de temperatura da API foi fixado num valor baixo (`temperature=0.3`). Esta decisão arquitetural diminui a criatividade da rede neuronal, forçando o modelo a focar-se em respostas precisas e coladas aos factos devolvidos pelos módulos de código, em detrimento da invenção de rotas inexistentes.

6.3.2 Roteamento Semântico e Tipagem de Ferramentas

Na arquitetura moderna de integração de modelos (via Gemini SDK), as ferramentas não necessitam de esquemas JSON lentos e propensos a falhas. O modelo extrai o contrato de dados lendo diretamente a assinatura da função Python, as suas anotações de tipagem (*Type Hints*) e a documentação interna (*Docstring*).

```

1 def search_viable_flight_alternatives(current_airport: str,
2     destination_airport: str,
3     departure_date: str,
4     delayed_arrival_time: str) -> str:
5     """
6     Pesquisa alternativas de voo logicamente viaveis perante um atraso.
7     Usa esta ferramenta SEMPRE que precisares de oferecer solucoes
8     alternativas.
9     """
10    # [Logica interna delegada ao motor TTDP e Recommender Engine]
11    pass

```

Listing 6.4: Assinatura da ferramenta de pesquisa de voos com tipagem e Roteamento Semântico.

Neste contexto, a *docstring* não opera apenas como documentação estática, mas como o gatilho de Roteamento Semântico. A instrução contida no comentário indica ao motor de inferência quando deve interromper a geração natural de texto para acionar o algoritmo de *backend*. A tipagem forte obriga o modelo a inferir corretamente as quatro variáveis temporais e espaciais a partir da conversa antes de poder executar o pedido.

6.3.3 Vinculação de Ferramentas na Orquestração

Finalmente, a instância de processamento é criada aderindo ao Princípio do Menor Privilégio. O modelo não possui acesso genérico ao sistema, as suas capacidades de ação são estritamente contidas à lista de ferramentas explicitamente injetadas durante a configuração da chamada.

```
1 chat = self.client.chats.create(  
2     model='gemini-2.5-flash',  
3     config=types.GenerateContentConfig(  
4         system_instruction=self.system_instruction,  
5         tools=[search_viable_flight_alternatives],  
6         temperature=0.3  
7     )  
8 )
```

Listing 6.5: Vinculação da ferramenta ao Orquestrador Gemini 2.5 Flash.

Quando o modelo Gemini 2.5 Flash recebe o alerta de interrupção (oriundo do sistema de simulação e ingestão), compreende que a sua diretiva exige uma resolução pró-ativa e invoca a função injetada no parâmetro `tools`. Somente após a ferramenta processar a algoritmia matemática (nos módulos de recomendação e restrição) e devolver os resultados validados é que o Orquestrador assume o seu papel final: sintetizar os dados *backend* numa resposta empática e estruturada para o passageiro.

6.4 O Motor Lógico e Restrições Físicas

A principal lacuna da aplicação exclusiva de redes neurais em problemas de roteamento turístico é a sua natureza opaca e estocástica, possuindo limitações comprovadas ao lidar com supervisão estrita e sendo frequentemente propensa a um “viés de otimismo” na geração de respostas (Colelough e Regli 2024). Para colmatar esta falha de fiabilidade em domínios de alto risco, a arquitetura *TravelMaster AI* adota uma abordagem neuro-simbólica, aliando o poder gerativo das redes neurais à precisão manipulativa do código determinístico (Hitzler e Sarker 2022). O módulo `ttdp_logic.py` atua como o motor simbólico e determinístico do sistema, resolvendo uma versão delimitada do TTDP.

6.4.1 O Princípio do Disjuntor Lógico

Na literatura de investigação operacional, as Restrições Rígidas representam limites operacionais que não podem ser violados sob qualquer circunstância. No contexto do planeamento de escalas aéreas, a restrição suprema é a física do tempo.

O motor lógico foi desenhado para atuar como um disjuntor lógico, antes de o Orquestrador ter a oportunidade de avaliar, sugerir ou formatar uma alternativa de voo, o módulo Python calcula inequivocamente se a ligação temporal é humanamente possível. Caso não seja, a opção é sumariamente descartada do espectro de “conhecimento” do modelo, anulando na origem a probabilidade de alucinação logística.

6.4.2 Validação Determinística de Escalas

O núcleo desta validação reside no método `validate_airport_connection`, responsável por processar as conexões dentro da mesma infraestrutura aeroportuária face ao Tempo Mínimo de Conexão (MCT – *Minimum Connection Time*).

```

1 def validate_airport_connection(self, arrival: str, departure: str, mct:
2   int) -> ConnectionValidationResult:
3
4   # Abstracao do calculo complexo de deltas temporais
5   available_minutes = self._calculate_minute_difference(arrival,
6   departure)
7
8   # Aplicacao estrita da restricao (Hard Constraint)
9   if available_minutes >= mct:
10      return ConnectionValidationResult(
11         is_viable=True,
12         margin_minutes=(available_minutes - mct),
13         reasoning=f"Viavel: Margem de {available_minutes}m cobre o
14         MCT."
15     )
16
17     return ConnectionValidationResult(
18         is_viable=False,
19         margin_minutes=0,
20         reasoning=f"Inviavel: Falta de tempo face ao MCT de {mct}m."
21     )

```

Listing 6.6: Algoritmo determinístico para validação de restrições rígidas temporais.

Este excerto ilustra duas decisões de engenharia críticas:

1. **Cálculo Seguro de Deltas Temporais:** A função auxiliar `_calculate_minute_difference` encapsula a conversão de *strings* para objetos cronológicos estruturados. Esta abstração garante o tratamento correto de travessias de dias (ex: um voo que aterriza às 23:30 e cujo voo de ligação parte às 01:00 do dia seguinte), impedindo que o cálculo da margem colapse face a variações de fuso horário.
2. **Eliminação de Viés Probabilístico:** A validação assenta numa condição puramente booleana (`available_minutes >= mct`). Não existe margem para inferências parciais, se o tempo disponível for inferior ao limiar de segurança por um único minuto, a ligação é classificada como inviável (`is_viable=False`). Esta inflexibilidade programática é a antítese do comportamento gerativo comum nos LLMs, assegurando rigor absoluto.

6.4.3 Integração de Inteligência Artificial Explicável

A inovação deste motor face a validadores tradicionais reside na sua estrutura de retorno tipada (`ConnectionValidationResult`). O algoritmo não devolve apenas um valor lógico falso, mas constrói simultaneamente uma cadeia de raciocínio textual contínua, uma técnica amplamente comprovada para potenciar as capacidades de dedução lógica e processamento intermédio em LLM (Wei et al. 2022).

Esta característica materializa, a nível do código-fonte, o conceito de Inteligência Artificial Explicável (XAI). Ao fornecer a justificação matemática exata (ex: “Margem de 20m é

insuficiente para o mínimo de 45m”), o motor oferece ao Orquestrador o contexto necessário para, na fase final de comunicação, traduzir o rigor do *backend* numa resposta transparente, justificada e empática para o utilizador, promovendo ativamente a sua confiança no sistema.

6.5 Heurística e Inteligência Artificial Explicável

Após a atuação do motor TTDP como um filtro absoluto, eliminando conexões fisicamente impossíveis através de Restrições Rígidas, o sistema transita de um problema de otimização estritamente operacional para uma abordagem centrada no utilizador. O módulo `recommender.py` itera sobre as opções de voo sobreviventes e aplica uma função subtrativa de custo logístico (Restrições Flexíveis), com o intuito principal de quantificar e mitigar a carga cognitiva do passageiro.

6.5.1 Implementação do Ciclo de Fricção

A cada alternativa viável é atribuída uma pontuação base de 100 pontos, sujeita a subsequentes deduções que modelam a “fricção” espacial e temporal da viagem. A Listagem 6.7 apresenta a materialização deste ciclo heurístico:

```

1 viable_options = []
2
3 for flight in raw_flights:
4     # 1. Filtro Físico (Restricao Rigida via TTDP)
5     validation = self.ttdp.validate_airport_connection(delayed_arrival,
6     flight["departure_time"])
7
8     if validation.is_viable:
9         score = 100
10        reasoning = []
11
12        # Regra A: Fricção Espacial (Penalização Severa)
13        if flight["departure_airport"] != origin_airport:
14            score -= 40
15            reasoning.append(f"Penalizado: Deslocacao para {flight['
16            departure_airport ']}.")
17        else:
18            reasoning.append("Parte do mesmo aeroporto.")
19
20        # Regra B: Fricção Temporal (Degradação Linear)
21        extra_wait = validation.margin_minutes
22        temporal_penalty = int(extra_wait / 30)
23        score -= temporal_penalty
24
25        if temporal_penalty > 0:
26            reasoning.append(f"Espera aeroportuaria de {extra_wait} min.
27            ")
28
29        # 2. XAI: Concatenação da fundamentação
30        viable_options.append({
31            "flight": flight["flight_number"],
32            "score": score,
33            "xai_reasoning": " | ".join(reasoning)
34        })
35
36 # 3. Ordenação decrescente pela qualidade da experiencia logistica
37 viable_options.sort(key=lambda x: x["score"], reverse=True)

```

Listing 6.7: Ciclo heurístico de pontuação e construção de Inteligência Artificial Explicável.

Este bloco de código evidencia o equilíbrio entre o rigor quantitativo e o conforto humano. A verificação da disparidade de aeroportos aplica uma penalização profunda (dedução de 40 pontos) a rotas que obriguem o passageiro a navegar numa infraestrutura intermodal desconhecida, prevenindo picos agudos de stress. Em contrapartida, a penalização temporal atua como uma degradação linear suave (dedução de 1 ponto por cada 30 minutos adicionais), refletindo o cansaço progressivo de aguardar num terminal, mas reconhecendo algoritmicamente que uma espera segura é invariavelmente preferível a uma deslocação reativa não planeada.

6.5.2 Transparência Algorítmica e Orquestração Empática

A implementação afasta-se deliberadamente do paradigma de caixa negra, característico de sistemas de Inteligência Artificial puramente conexionistas, cuja opacidade levanta desafios significativos à confiança e adoção por parte dos utilizadores em contextos críticos (Lipton 2018). Em contraponto, o *TravelMaster AI* opta por auto-documentar a sua árvore de

decisão em tempo real. A chave `xai_reasoning` representa o elo estrutural vital entre a formulação determinística do *backend* e o Orquestrador estocástico.

Quando a ferramenta devolve o dicionário de opções ordenado, o LLM não recebe meramente dados tabulares estereis, herda a justificação algorítmica exata que alocou uma determinada opção no topo da lista (ex: *“Parte do mesmo aeroporto / Espera aeroportuária de 105 min”*).

Esta injeção bidirecional de Inteligência Artificial Explicável (XAI) força o modelo de linguagem a sustentar a sua comunicação nos factos gerados pela aplicação. O Orquestrador consome a cadeia de raciocínio rígida e traduz-a para uma linguagem humana tranquilizadora (ex: *“Sugiro o voo alternativo TP123 pois permite-lhe aguardar confortavelmente no mesmo terminal, evitando deslocações adicionais de autocarro”*).

Através desta simbiose técnica, o sistema responde diretamente ao seu propósito de investigação: orquestrar a fluência semântica da IA gerativa não apenas para recuperar de falhas operacionais, mas para mitigar ativamente a ansiedade do passageiro perante a incerteza. Ao fornecer a justificação subjacente à máquina, o artefacto providencia transparência e constrói um vínculo de fiabilidade empática com o utilizador, um requisito fundamental validado pela literatura de explicabilidade e confiança algorítmica (Ribeiro et al. 2016).

6.6 Interface e Gestão de Estado da Sessão

A camada de apresentação do TravelMaster AI não atua meramente como um terminal de visualização, mas como um coordenador de estado que assegura a continuidade da experiência do utilizador. O desafio técnico reside na natureza inerente da ferramenta utilizada, o Streamlit, que opera sob um modelo de execução sem estado, cada interação do passageiro força a reexecução integral do código. Para transpor esta limitação e dotar o sistema de uma “ponte de memória”, a implementação recorre ao objeto de estado da sessão.

6.6.1 Persistência do Contexto de Viagem

A conservação dos dados extraídos do bilhete (Capítulo 6.2) é fundamental para evitar o processamento redundante e a latência desnecessária. Uma vez concluída a ingestão multimodal, o objeto resultante é armazenado de forma persistente. Esta abordagem minimiza o consumo de recursos computacionais e o custo associado aos *tokens* da API, garantindo que a informação da viagem permanece como a âncora de contexto para todas as decisões subsequentes do Orquestrador.

A Listagem 6.8 exemplifica a inicialização segura destes componentes:

```
1 # Inicializacao do Orquestrador (Singleton-like)
2 if "orchestrator" not in st.session_state:
3     st.session_state.orchestrator = TravelMasterOrchestrator()
4
5 # Armazenamento do bilhete para evitar reprocessamento
6 if uploaded_file and "ticket_data" not in st.session_state:
7     st.session_state.ticket_data = parse_flight_info_from_file(temp_path
8     )
```

Listing 6.8: Inicialização e persistência de componentes no estado da sessão.

6.6.2 Gestão da Conversação e Transparência Cognitiva

Para mitigar a carga cognitiva, é imperativo que o utilizador possa consultar o histórico de recomendações e a respetiva fundamentação lógica sem que estes se dissipem após uma nova simulação. A persistência do histórico de mensagens permite que a interface se comporte como uma aplicação de conversação moderna, mantendo a visibilidade sobre as decisões do sistema.

Do ponto de vista da XAI, esta memória é vital: o passageiro pode reavaliar as justificações apresentadas (ex: as penalizações aplicadas por mudança de infraestrutura) enquanto pondera a sua decisão final. A implementação desta estrutura é detalhada abaixo:

```
1 # Inicializacao do historico de conversacao
2 if "messages" not in st.session_state:
3     st.session_state.messages = []
4
5 # Renderizacao das mensagens anteriores para manter o contexto visual
6 for msg in st.session_state.messages:
7     st.chat_message(msg["role"]).write(msg["content"])
8
9 # Adicao da fundamentacao do Orquestrador ao historico
10 st.session_state.messages.append({"role": "assistant", "content":
    resposta_agente})
```

Listing 6.9: Gestão e renderização do histórico de mensagens persistente.

6.6.3 Conclusão da Arquitetura de Apresentação

A interface visual atua como o ponto de convergência entre o *backend* determinístico e a camada reativa. Ao centralizar a gestão de dados no estado da sessão, o sistema assegura que a experiência é fluida, previsível e contextualizada. Esta arquitetura visual não só gere a apresentação de dados, como orquestra a exibição de métricas de desempenho e alertas de erro, garantindo que o passageiro mantém o controlo sobre o processo de resolução da disrupção, sem perdas de informação ou necessidade de repetição de comandos.

Com a instanciação física do protótipo finalizada e a interface de apresentação plenamente integrada com o motor lógico, encerra-se o ciclo de construção preconizado pela metodologia DSR. O artefacto tecnológico *TravelMaster AI* encontra-se, assim, em condições de ser submetido a escrutínio empírico. O capítulo seguinte transita para a fase de Avaliação, expondo o protocolo de testes em ambiente laboratorial simulado e discutindo criticamente os resultados quantitativos obtidos face às Questões de Investigação delineadas.

Capítulo 7

Avaliação e Resultados

Este capítulo apresenta os resultados da avaliação empírica do artefacto *TravelMaster AI*. Em estrito cumprimento com as orientações da metodologia *Design Science Research* (DSR), a avaliação visa demonstrar a utilidade, qualidade e eficácia do protótipo na resolução do problema prático identificado: o elevado atrito interacional e o esforço operacional exigido aos passageiros durante disrupções aéreas. Para tal, as secções seguintes detalham o ambiente de teste laboratorial, avaliam as métricas de latência e resiliência, e analisam a redução de fricção suportada por inteligência explicável (XAI), cruzando os resultados quantitativos diretamente com as Questões de Investigação (QI) e apontando as limitações experimentais do estudo.

7.1 Metodologia de Avaliação em Ambiente Laboratorial

A avaliação de sistemas críticos em ambiente real (como disrupções logísticas em aeroportos) apresenta constrangimentos éticos e práticos severos, nomeadamente a impossibilidade de sujeitar utilizadores reais a falhas de voo propositadas para efeitos de teste. Adicionalmente, tratando-se de um Minimum Viable Product (MVP), a literatura de DSR suporta a utilização de ensaios laboratoriais controlados nas fases iniciais de instanciação do artefacto (Hevner et al. 2004).

Nesse sentido, a validação do *TravelMaster AI* assentou numa metodologia de Avaliação Simulada. A adoção desta infraestrutura simulada não constituiu um mero recurso de conveniência, mas sim uma exigência metodológica deliberada para assegurar o rigor científico do estudo empírico. Especificamente, a simulação foi adotada com o intuito central de:

- **Garantir a reprodutibilidade:** Assegurando que a avaliação da resposta do agente (latência e precisão lógica) ocorre sob parâmetros exatos e determinísticos, imunes à flutuação imprevisível de dados de tráfego real.
- **Permitir o controlo experimental:** Isolando as variáveis independentes em estudo (como o impacto da injeção de uma disrupção na geração de *Tool Calling*) sem a interferência de constrangimentos externos das APIs comerciais.
- **Injetar falhas críticas de forma segura:** Possibilitando forçar o sistema a lidar com ruturas severas (*timeouts* da infraestrutura do LLM ou perdas de escala logística) sem causar danos, stress ou prejuízo financeiro a passageiros reais ou aeroportos.
- **Avaliar cenários extremos:** Viabilizando a recriação deliberada de falhas multimodais simultâneas que seriam eventos raros e estatisticamente difíceis de capturar num período temporal limitado de testes em produção.

Com base nestas premissas de controlo, o sistema foi submetido ao cenário crítico de teste (Cenário C - Atraso Multimodal e Perda de Conexão), injetando-se artificialmente atrasos estruturados através dos componentes operacionais do sistema (`amadeus_api.py` e `maps_api.py`). A avaliação metodológica decorrente foi dividida em duas dimensões complementares:

1. **Dimensão Técnica:** Focada na medição de métricas de engenharia (latência e resiliência a falhas de API) e na validação da robustez do motor neuro-simbólico contra alucinações.
2. **Dimensão Centrada no Utilizador:** Uma análise comparativa teórica que quantifica a redução da carga cognitiva face à utilização de métodos manuais de replaneamento.

7.1.1 Especificações do Ambiente de Teste e Validade Ecológica

Para garantir a reprodutibilidade dos resultados empíricos, o protótipo foi avaliado localmente numa máquina com as seguintes especificações de *hardware* e rede:

- **Sistema Operativo:** Windows 10
- **Processador (CPU):** AMD Ryzen 7 4800H
- **Memória (RAM):** 16 GB
- **Placa Gráfica (GPU):** NVIDIA GTX 1650 Ti
- **Conectividade de Rede:** Fibra Ótica de 1 Gbps

É importante salientar que, embora a máquina de testes possua uma dedicada, o processamento do LLM não ocorreu localmente. A arquitetura do *TravelMaster AI* foi concebida sob o paradigma *Cloud-Native*, dependendo integralmente da API do Google Gemini. Consequentemente, a métrica mais crítica deste ambiente não é a capacidade de processamento gráfico local, mas sim a largura de banda da rede. A conectividade estabilizada eliminou estrangulamentos de rede locais, permitindo avaliar a latência real de ida e volta da API do modelo de linguagem, provando que o sistema é perfeitamente exequível num ambiente leve.

Sendo este um projeto suportado por Gateways Simulados para representar serviços externos, a leitura direta de ficheiros estáticos no disco local resultaria num tempo de resposta quase imediato. Isto invalidaria a precisão das métricas de execução nos testes, gerando resultados excessivamente otimistas.

Para salvaguardar a validade do estudo empírico, foram introduzidos atrasos artificiais estruturados (`time.sleep()`) no código-fonte dos *Gateways*, imitando os constrangimentos de uma rede de produção:

- **Gateway de Aviação (`amadeus_api.py`):** Sempre que o Orquestrador consulta o estado de um voo ou invoca a ferramenta de pesquisa de alternativas, o sistema bloqueia propositadamente durante 1.5 segundos, simulando o tempo de roteamento a um servidor externo de aviação.
- **Gateway de Trânsito Terrestre (`maps_api.py`):** De forma análoga, o cálculo da matriz de distâncias incorpora um atraso artificial de 1.2 segundos para simular o tempo de processamento de um servidor de mapas.

Esta calibração garante que os tempos totais de processamento obtidos e discutidos nas secções seguintes representam valores fidedignos e pessimistas, atestando a robustez do agente face aos atritos temporais do mundo real.

Parametrização e Reprodutibilidade Experimental

Para assegurar o rigor científico e a reprodutibilidade integral dos ensaios, o ambiente de desenvolvimento foi rigorosamente estandardizado e monitorizado através das seguintes parametrizações operacionais:

- **Versionamento e Dependências:** O protótipo foi executado num ambiente virtualizado Python (versão 3.11), com o controlo de versões das bibliotecas (ex: *SDK* da Google) fixado num ficheiro de dependências estrito (*requirements.txt*). Para garantir a consistência das inferências, utilizou-se a versão estática do modelo de linguagem (*gemini-2.5-flash*), com o parâmetro de entropia (*temperature*) configurado com um valor baixo (0.3). Esta parametrização garante um equilíbrio ótimo, inibe a alucinação de dados logísticos ou rotas inexistentes, forçando um comportamento determinístico no *Tool Calling*, mas preserva a flexibilidade semântica necessária para gerar respostas finais empáticas e naturais.
- **Políticas de Logging:** Foi implementado um sistema de rastreabilidade no código que regista o ciclo de vida completo de cada interação. O sistema audita, com precisão de milissegundos, os instantes de envio e receção de cada *Tool Call*, as atuações do mecanismo de tolerância a falhas e os *payloads* de validação bloqueados pelo módulo TTDP. Este registo sistemático foi o mecanismo que permitiu isolar analiticamente a latência de comunicação externa do tempo de processamento lógico interno.
- **Condições de Execução e Estado Base:** Todos os cenários de teste foram instanciados a partir de um estado limpo. Esta reinicialização da sessão de conversação do LLM e limpeza do contexto de viagem na interface entre cada iteração garante a ausência de poluição de dados. Consequentemente, assegura-se que os tempos médios de reação medidos refletem de forma fidedigna o comportamento do sistema perante estímulos logísticos inéditos, sem aproveitamento de memória em *cache*.

7.2 Avaliação de Desempenho e Resiliência (QI1)

A primeira métrica de sucesso para um sistema de assistência autónoma é a sua capacidade de reagir atempadamente e sobreviver a falhas de infraestrutura. A Questão de Investigação 1 (QI1) determinava como critério um tempo de resposta inferior a 30 segundos, assegurando que o passageiro é notificado de uma alternativa antes de a mesma esgotar nos balcões do aeroporto.

7.2.1 Latência de Processamento do Orquestrador

Para a presente avaliação empírica, o sistema foi submetido a $N = 30$ execuções independentes do Cenário C. Neste cenário de stress, o utilizador deslocava-se do hotel para o Aeroporto de Hong Kong (HKG), enfrentando simultaneamente um atraso de trânsito severo de 85 minutos e um atraso de voo de 30 minutos.

Os tempos foram medidos desde a injeção do gatilho de alerta até à renderização da solução validada no ecrã. Este intervalo engloba as latências de rede induzidas (conforme Secção

7.1.1), as inferências matemáticas do motor TTDP e a geração de texto do modelo Gemini 2.5 Flash. A Tabela 7.1 apresenta a estatística descritiva destas execuções.

Tabela 7.1: Estatística descritiva da latência de processamento do Orquestrador ($N = 30$).

Amostra (N)	Mínimo	Máximo	Média (μ)	Desvio-Padrão (σ)	IC 95%
30	8.24 s	19.26 s	12.54 s	2.59 s	12.54 $\pm$ 0.93 s

A análise dos resultados demonstra uma elevada consistência operacional. O tempo médio de resolução fixou-se nos 12.54 segundos, com um desvio-padrão de 2.59 segundos. Para formalizar o rigor estatístico da amostra, calculou-se o Intervalo de Confiança (IC) a 95% para a média recorrendo à formulação $\mu \pm 1.96 \cdot (\sigma/\sqrt{N})$. O resultado obtido (12.54 $\pm$ 0.93 s) estabelece com elevada probabilidade que o verdadeiro tempo médio de processamento do sistema em produção se situará entre 11.61 e 13.47 segundos.

Mais criticamente, quer o limite superior deste intervalo de confiança, quer o valor máximo isolado registado em toda a amostra (19.26 segundos), mantêm-se confortavelmente abaixo do limite estrito de 30 segundos estabelecido na Q11. Estes dados sustentam o critério de sucesso nos cenários avaliados, demonstrando que o acoplamento de uma validação arquitetural com *Tool Calling* não penalizou significativamente a experiência do utilizador final, com a latência observada a manter-se dentro da janela de utilidade operacional definida.

Para isolar a carga computacional introduzida pelo processamento multimodal concorrente, realizou-se uma avaliação complementar ($N = 30$) utilizando exclusivamente a disrupção terrestre.

Tabela 7.2: Estatística descritiva da latência: Disrupção Terrestre Isolada ($N = 30$).

Amostra (N)	Mínimo	Máximo	Média (μ)	Desvio-Padrão (σ)	IC 95%
30	6.88 s	15.21 s	9.30 s	2.44 s	9.30 $\pm$ 0.87 s

A comparação entre as duas condições revela que a disrupção terrestre isolada foi concluída, em média, 3.24 segundos mais rápido do que o cenário multimodal (9.30 s face a 12.54 s). Esta redução de aproximadamente 25.8% comprova empiricamente que a latência da arquitetura neuro-simbólica escala de forma previsível e coerente consoante a complexidade e o número de variáveis da disrupção enfrentada, afastando o risco de um comportamento errático sob variações de stress.

7.2.2 Tolerância a Falhas e Degradação Graciosa

Sistemas dependentes de Inteligência Artificial baseada em arquiteturas *Cloud* estão intrinsecamente sujeitos a picos de tráfego, limites de taxa de pedidos e períodos de indisponibilidade. Durante os ensaios laboratoriais, o protótipo confrontou-se com exceções esporádicas de excesso de pedidos da API da Google.

Para mitigar este risco sem expor o passageiro a mensagens de erro técnicas, foi implementado e validado um padrão arquitetural de resiliência: o recuo exponencial com limite rígido, fundamentado nas boas práticas de estabilidade de sistemas em produção (Nygard 2007).

```
1 max_retries = 3
2 base_delay = 2
3
4 for attempt in range(max_retries):
5     try:
6         chat = self.client.chats.create(...)
7         response = chat.send_message(user_prompt)
8         return response.text
9     except Exception as e:
10        if attempt == max_retries - 1:
11            logger.error(f"Falha definitiva: {e}")
12            return ("Pedimos desculpa, mas os servidores estão com
13                tráfego elevado. "
14                "Por favor, dirija-se imediatamente a um balcão da
15                companhia aérea.")
16
17        # Calculo do recuo exponencial (Exponential Backoff)
18        sleep_time = base_delay * (2 ** attempt)
19        time.sleep(sleep_time)
```

Listing 7.1: Mecanismo de recuo exponencial e degradação graciosa no Orquestrador.

O limite estabelecido atua como um disjuntor. De acordo com Nygard (2007), este padrão é essencial para evitar falhas em cascata em sistemas distribuídos. Em caso de falha irreversível, a aplicação não bloqueia indefinidamente, em vez disso, silencia a exceção e invoca uma estratégia de degradação graciosa. A emissão de uma mensagem determinística recomendando o recurso imediato a um balcão humano cumpre o propósito último do sistema, encaminhar o utilizador para a resolução do seu problema, mesmo perante a falha total do orquestrador, sem agravar o seu estado de stress.

7.3 Avaliação do Esforço Operacional

Para além da robustez técnica, a eficácia do *TravelMaster AI* é aferida pela sua capacidade de mitigar o atrito interacional e o esforço operacional exigido ao passageiro. Esta dimensão de avaliação foca-se na redução do número de interações manuais e na eliminação da necessidade de alternância constante entre aplicações.

7.3.1 Cenário de Teste: Disrupção Multimodal Simultânea

A avaliação foi centrada no teste de limite do sistema (*worst-case scenario*), configurando uma disrupção em cascata que afeta múltiplos eixos da viagem. O utilizador, com um bilhete original de Danpasar (DPS) para Hong Kong (HKG), foi confrontado simultaneamente com:

- **Atraso Terrestre:** Um acidente rodoviário grave simulado via *Maps Gateway*, aumentando o tempo de trânsito para 85 minutos.
- **Atraso de Voo:** Um atraso de 30 minutos na partida original, reduzindo a janela de oportunidade para a realização do *check-in* e segurança.

O sistema reagiu autonomamente a este gatilho duplo, invocando o Orquestrador para recalcular a viabilidade da viagem face ao novo tempo estimado de chegada ao aeroporto.

7.3.2 Análise de Fricção de Interação e XAI

O atrito num cenário de disrupção é frequentemente exacerbado pela necessidade de consultar múltiplas fontes de informação fragmentadas (alternância de contexto) e realizar o cruzamento manual de dados. Para quantificar a redução deste esforço operacional (respondendo à Q14 de forma empírica), recorreu-se a uma Análise de Tarefas comparativa fundamentada nos princípios do Keystroke-Level Model (KLM) (Card et al. 1980), mapeando as interações necessárias para resolver o cenário de disrupção através de métodos tradicionais face à utilização do *TravelMaster AI* (Tabela 7.3).

Tabela 7.3: Análise de Tarefas: Método Manual fragmentado vs. *TravelMaster AI*.

Passo	Método Manual (Via Apps Tradicionais)	TravelMaster AI
1	Abrir <i>app</i> de trânsito para verificar atraso rodoviário.	Receção de notificação proativa com ETA e margem calculada.
2	Cálculo mental do tempo de chegada ao aeroporto.	Leitura da <i>Reasoning String</i> (XAI) com a recomendação.
3	Abrir <i>app</i> ou <i>website</i> da companhia aérea.	Confirmação da alternativa com um clique (<i>Aceitar</i>).
4	Inserir dados da reserva (Passenger Name Record (PNR)) e pesquisar alternativas.	—
5	Analisar opções de voo disponíveis.	—
6	Cruzar (mentalmente) a hora de chegada com as novas partidas.	—
7	Selecionar o voo e confirmar alteração.	—
Resumo	7 a 10 interações + Elevada carga mental de cálculo.	1 a 2 interações + Nula carga de cálculo cruzado.

Enquanto o método manual exige que o utilizador alterne entre aplicações operando como o seu próprio "orquestrador de dados", o *TravelMaster AI* apresenta a solução final já purificada e fundamentada:

"A melhor opção que encontrei é o Voo CX 846 [...] pois parte do mesmo aeroporto, o que significa que não terá de se deslocar para outro local."

A inclusão da fundamentação lógica baseada em factos determinísticos ("dá-lhe bastante tempo (mais de 3 horas) para passar pela segurança") evita que o utilizador sinta necessidade de ir validar a proposta do sistema ao Google. A explicação suportada pelo TTDP providencia a confiança necessária para uma decisão imediata, a análise sugere uma redução substancial do número de interações manuais, de uma média estimada de 10 ações para apenas 1 interação de confirmação, o que apoia o critério de sucesso estabelecido para a redução do esforço operacional e do tempo de recuperação.

7.3.3 Análise de Sensibilidade e Casos de Fronteira (Edge Cases)

A robustez de um sistema de assistência autónoma não se afere exclusivamente pela sua resposta a cenários extremos, mas também pela sua calibração perante estímulos logísticos singulares e variações de menor escala. Para validar a sensibilidade e especificidade da arquitetura, o protótipo foi submetido a uma matriz complementar de testes de fronteira (*edge cases*):

- **Cenário A: Disrupção Terrestre Isolada (Falta de Margem Física):** Foi injetado um alerta de acidente grave no percurso, prolongando o trânsito do hotel para o aeroporto de 35 para 85 minutos. Face a uma chegada estimada às 15:55 e um voo programado para as 16:15, o motor TTDP verificou que a margem restante (20 minutos) violava o tempo mínimo de segurança do aeroporto (90 minutos). A ligação foi instantaneamente classificada como inviável (*is_viable=False*). O Orquestrador assumiu o voo como perdido e, proativamente, ofereceu uma alternativa de resgate (voo às 18:30).
- **Cenário B: Atraso Operacional Severo (Otimização de Fricção Temporal):** Simulou-se um atraso exclusivo da responsabilidade da companhia aérea (via API Amadeus), adiando a partida original das 16:15 para as 20:15. Como o passageiro já possuía margem para chegar atempadamente, o objetivo do sistema reorientou-se para a minimização da “Fricção Temporal” abordada no Capítulo 6.5. O Orquestrador invocou a ferramenta de pesquisa e ofereceu ativamente o voo alternativo das 18:30. Esta ação poupa ao passageiro a acumulação de fadiga de uma espera passiva de quatro horas pelo voo original atrasado.
- **Cenário C: O Atraso Inócuo (Falso Positivo e Inação Deliberada):** O teste crítico de especificidade injetou perturbações ligeiras (+15 minutos no trânsito ou +10 minutos no voo). Perante este cenário, o cálculo determinístico do TTDP confirmou que o tempo total disponível se mantinha superior à restrição exigida, retornando *is_viable=True*.

A consequência arquitetural do Cenário C constitui uma das maiores provas empíricas de sucesso do sistema: devido à aprovação matemática do TTDP, o bloqueio lógico impediu a invocação da ferramenta de pesquisa de voos (*Tool Calling*).

Esta inação deliberada forçou o Orquestrador LLM a assumir uma postura exclusivamente comunicacional, gerando a seguinte resposta:

“Detetámos um ligeiro atraso no trânsito (+15 min), mas tem margem suficiente para passar a segurança. Não se preocupe, chegará a tempo do seu voo das 16:15.”

A capacidade do sistema atuar como filtro absoluto contra falsos positivos prova a eficiência simbiótica da arquitetura neuro-simbólica. O *TravelMaster AI* evitou não só a indução de pânico desnecessário no utilizador, como também otimizou recursos computacionais ao suprimir o consumo de custo logístico e inferência (*tokens* de API) num contexto onde a disrupção não representava uma ameaça logística real.

7.4 Discussão e Resposta às Questões de Investigação

Esta secção sintetiza as evidências recolhidas, respondendo formalmente às Questões de Investigação (QI) formuladas.

Resposta à QI1 (Monitorização Autónoma): A arquitetura provou ser estável e altamente responsiva. Com uma latência média de notificação de 12.54 segundos em cenários multimodais severos e uma redução previsível para 9.30 segundos em disrupções singulares, o sistema superou largamente o critério de sucesso de 30 segundos. A implementação de mecanismos de espera exponencial garantiu que, mesmo perante falhas de infraestrutura de rede, o sistema mantém a sua utilidade operacional através de uma degradação graciosa.

Resposta à QI2 (Capacidade de Reação): A integração de LLM com dados de contexto em tempo real permitiu gerar alternativas personalizadas e proativas. No cenário de teste, o agente identificou corretamente o voo CX 846 como a opção ótima, priorizando a permanência na mesma infraestrutura para reduzir o risco logístico, cumprindo o objetivo de oferecer soluções relevantes face a disrupções severas.

Resposta à QI3 (Fiabilidade e Validação Lógica): A arquitetura híbrida neuro-simbólica foi eficaz na eliminação de alucinações textuais. Ao subordinar o Orquestrador Gemini 2.5 Flash ao validador determinístico TTDP, não foram observadas alucinações logísticas nos cenários laboratoriais realizados, sugerindo que o módulo determinístico é eficaz na contenção das falhas de raciocínio do modelo. Este resultado responde diretamente à lacuna estrutural estabelecida no Estado da Arte, enquanto os assistentes generativos puros avaliados no *benchmark TravelPlanner* demonstraram uma incapacidade crónica de resolver restrições rígidas, obtendo uma taxa de sucesso de apenas 0,6% (Xie et al. 2024), a arquitetura do *TravelMaster AI* assegurou a viabilidade matemática das rotas avaliadas nos cenários laboratoriais testados. Esta oposição comprova que a limitação dos LLM em planeamento não se resolve com aumento de escala do modelo, mas sim com contenção determinística.

Resposta à QI4 (Redução de Esforço Operacional): O protótipo demonstrou uma redução superior a 50% no esforço de interação. A centralização da monitorização e do replaneamento numa única interface eliminou a fragmentação informativa e a alternância entre aplicações. A avaliação analítica (KLM) comprova que a conversão de dados complexos em recomendações justificadas (XAI) permite ao passageiro resolver o imprevisto com um número mínimo de cliques, validando a eficácia do *TravelMaster AI* como um assistente de baixo atrito funcional e rápida recuperação tática.

Para sistematizar a validade científica da avaliação, a Tabela 7.4 apresenta a correspondência final e estruturada entre as Questões de Investigação originais, a evidência empírica recolhida no ambiente laboratorial, o resultado científico obtido e as limitações associadas a cada resposta metodológica. Esta matriz confirma o cumprimento integral dos objetivos do artefacto.

Tabela 7.4: Correspondência explícita entre Questões de Investigação, Evidências e Resultados Obtidos.

Questão de Investigação (QI)	Evidência Produzida	Resultado Obtido	Limitações da Resposta
QI1: Monitorização Autónoma (Viabilidade técnica e latência)	Logs de execução do Cenário C com três iterações; ativação contínua do ciclo de execução.	Sucesso (< 30s): Latência validada em 12.54 s (multimodal) e 9.30 s (singular). Validação do recuo exponencial	As falhas de rede foram induzidas; requer teste contínuo em ambiente de produção com latência volátil.
QI2: Capacidade de Reação (Contexto e personalização)	Aprovação de rotas alternativas nos Cenários A e B de Casos de Fronteira.	Sucesso Proativo: O sistema invocou a API correta e priorizou voos da mesma aliança no mesmo aeroporto.	As preferências de viagem assumiram um perfil utilitário padronizado, sem variáveis puramente subjetivas.
QI3: Fiabilidade Lógica (Mitigação de Alucinações)	Inação deliberada do Orquestrador perante falso positivo no Cenário C de Fronteira.	Sucesso Absoluto nos cenários avaliados (0%): O disjuntor TTDP bloqueou a geração textual quando a matemática provou exequibilidade.	O motor matemático foi testado sob constrangimentos simples (tempo/distância).
QI4: Redução Carga Cognitiva (Interações e esforço)	Avaliação Heurística baseada no <i>Keystroke-Level Model</i> (KLM) (Tabela 7.3).	Sucesso Sistémico (> 50%): Redução de interações de 7 – 10 cliques dispersos para 1 – 2 cliques na mesma janela.	Avaliação puramente analítica e teórica (KLM); ausência de ensaios biométricos com utilizadores sob stress.

7.5 Ameaças à Validade e Limitações Experimentais

Apesar dos resultados empíricos sustentarem a eficácia do *TravelMaster AI* na resposta às Questões de Investigação, a interpretação destas métricas deve ser enquadrada pelas limitações inerentes ao desenho experimental. O rigor da Engenharia de *Software* e da metodologia DSR exige a identificação formal das restrições do estudo e das suas Ameaças à Validade (*Threats to Validity*).

7.5.1 Limitações Experimentais

O protocolo de testes laboratorial introduziu constrangimentos operacionais necessários para a exequibilidade do projeto, destacando-se:

- **Natureza laboratorial e dependência de APIs simuladas:** O sistema foi avaliado num ambiente laboratorial fechado com recurso a *Mock Gateways*. Embora esta decisão garanta controlo absoluto e permita injetar falhas severas, suprime a volatilidade imprevisível do tráfego de rede de uma API comercial de aviação ao vivo.
- **Ausência de utilizadores reais:** A avaliação centrou-se em perfis de utilizador sintéticos e numa análise teórica de tarefas. A resposta humana a imprevistos inclui fatores

emocionais de pânico e hesitação, cuja fricção temporal um ambiente laboratorial automatizado não consegue replicar plenamente.

- **Número limitado de cenários:** Foram desenhados cenários críticos representativos (incluindo desvios de fronteira e *Edge Cases*), contudo, a infinita combinatória de disrupções logísticas do mundo real (ex: greves intermodais, clima extremo em múltiplos aeroportos) ultrapassa a cobertura desta validação inicial.

7.5.2 Ameaças à Validade

Para além das limitações de execução, identificam-se as seguintes ameaças à interpretação e generalização dos resultados:

- **Validade Interna:** A principal ameaça interna ao estudo reside no viés introduzido pela parametrização dos atrasos artificiais na telemetria. Para mitigar o risco de obter resultados excessivamente otimistas de tempo de resposta, os atrasos injetados no código foram deliberadamente conservadores, assegurando que a medição final da latência suporta uma margem de erro realista face a constrangimentos de processamento em nuvem.
- **Validade de Construto:** Avaliar objetivamente a carga cognitiva e o stress é um desafio metodológico complexo. A utilização da métrica KLM ameaça a validade do construto ao assumir matematicamente que menos interações físicas e cliques equivalham de forma direta a menos stress. Embora o KLM seja um padrão científico fiável para quantificar atrito funcional, não capta a vertente da ansiedade psicossomática, pelo que o resultado reflete o ganho em usabilidade sistémica e não necessariamente a totalidade do estado emocional humano.
- **Validade Externa e Limitações de Generalização:** A capacidade de generalizar o sucesso da orquestração neuro-simbólica encontra-se limitada pelo perfil do itinerário testado. O motor TTDP foi validado num eixo intercontinental de grande dimensão (DPS-HKG). A generalização da eficácia do agente para contextos operacionais de companhias aéreas *low-cost* (que não garantem escalas protegidas) ou para aeroportos regionais de infraestrutura não documentada, constitui uma limitação que requererá futuras afinações algorítmicas das Restrições Flexíveis.

Em suma, a avaliação laboratorial e a subsequente análise de fricção de interação confirmam o cumprimento dos requisitos de desempenho e a validação das Questões de Investigação delineadas na metodologia. Estando a eficácia arquitetural empiricamente comprovada, e as limitações do estudo explicitamente identificadas e mitigadas, o capítulo final desta dissertação apresenta uma síntese dos contributos científicos alcançados e perspetiva direções estruturadas para o desenvolvimento futuro do ecossistema.

Capítulo 8

Conclusões e Trabalho Futuro

Este capítulo encerra a presente dissertação, apresentando uma reflexão global sobre a investigação realizada. A secção inicial sintetiza a jornada do projeto, relembrando o problema original e a abordagem adotada. De seguida, são destacadas as principais contribuições teóricas e práticas do artefacto desenvolvido. Por fim, num exercício de rigor académico, são reconhecidas as limitações do estudo e delineadas vias promissoras para futuras investigações na intersecção entre o turismo e a Inteligência Artificial.

8.1 Síntese do Trabalho Desenvolvido

A gestão de disrupções em viagens aéreas, como atrasos severos ou cancelamentos, representa um dos momentos de maior stress e sobrecarga cognitiva para os passageiros. Face a um imprevisto, o viajante é forçado a atuar como um orquestrador manual de dados, alternando entre múltiplas fontes fragmentadas (aplicações de trânsito, *emails*, painéis de aeroporto) para recalcular a viabilidade da sua rota. Embora a ascensão dos LLMs apresente potencial para revolucionar a assistência virtual, a literatura demonstra que a sua natureza probabilística os torna propensos a alucinações logísticas, invalidando o seu uso isolado em cenários operacionais críticos.

Para dar resposta a esta lacuna, a presente investigação socorreu-se da metodologia *Design Science Research* (DSR) para desenhar, desenvolver e avaliar o *TravelMaster AI*. Em contraposição às soluções de mercado prevalentes, que se esgotam no planeamento estático, recomendação de destinos ou simples agregação de reservas, este protótipo foca-se estritamente na fase de execução da viagem. O sistema transcende o paradigma do agente conversacional tradicional ao assegurar a monitorização contínua do percurso físico, a reação autónoma a disrupções e a validação determinística de alternativas, reduzindo ativamente a carga operacional do utilizador num cenário dinâmico e de imprevisibilidade.

A solução arquitetural proposta assentou num modelo Neuro-Simbólico. O sistema não permite que a Inteligência Artificial gerativa (Gemini 2.5 Flash) calcule rotas ou tome decisões logísticas. Em vez disso, o LLM atua exclusivamente como uma interface de linguagem fluente, cujas respostas estão estritamente subordinadas a um motor matemático determinístico desenvolvido em Python.

A arquitetura processa o contexto inicial através de ingestão multimodal de bilhetes e monitoriza o estado da viagem comunicando com APIs externas. Perante a injeção de uma disrupção, o motor lógico aplica Restrições Rígidas baseadas numa versão delimitada do TTDP para atuar como disjuntor absoluto contra ligações impossíveis. Simultaneamente,

aplica Restrições Flexíveis para pontuar as alternativas viáveis, penalizando severamente opções que induzam maior atrito físico e cognitivo, como a mudança de aeroporto ou esperas extenuantes.

A avaliação laboratorial do artefacto comprovou a validade da tese. A integração de restrições algorítmicas não produziu alucinações logísticas detetáveis nos cenários avaliados. Adicionalmente, a tradução do raciocínio matemático em Inteligência Artificial Explicável (XAI) gerou justificações transparentes e empáticas que, aliadas a uma latência média de processamento que escala previsivelmente entre os 9 e os 12 segundos consoante a complexidade da disrupção, reduziram o esforço de replaneamento do utilizador em mais de 50%, comprovando a eficácia do *TravelMaster AI* na mitigação do stress logístico.

8.2 Contribuições Principais

A presente investigação resulta num conjunto de contribuições tangíveis que intersejam as áreas dos Sistemas de Informação, da Interação Humano-Computador e da Gestão Turística. O desenvolvimento do artefacto provou que é possível transpor as limitações atuais da Inteligência Artificial Generativa através de metodologias rigorosas de engenharia de *software*.

Em alinhamento direto com a proposta inicial da dissertação, o sistema *TravelMaster AI* consolida o seu valor como artefacto científico através de quatro eixos fundamentais:

Mecanismo de Monitorização e Replaneamento Contextual: No domínio prático e conceptual, a tese contribui com um modelo de orquestração proativa que inverte o ónus da resolução de problemas logísticos. Ao centralizar a ingestão de dados, a monitorização passiva e o replaneamento ativo numa interface unificada, o sistema comprovou ser capaz de reduzir quantitativamente o esforço operacional em mais de 50%. Este mecanismo estabelece que a IA orquestrada com precisão é uma ferramenta validada para mitigar o atrito funcional e a ansiedade situacional no setor dos transportes.

Arquitetura Híbrida Neuro-Simbólica: A principal inovação estrutural desta investigação é a demonstração de que a fiabilidade em domínios logísticos estritos pode ser assegurada através de confinamento arquitetural e não apenas pelo aumento da escala dos modelos. Ao isolar a capacidade gerativa do LLM e subordiná-la a um motor lógico (TTDP), estabeleceu-se um modelo de referência em que a Inteligência Artificial atua em cenários críticos desprovida de autonomia de cálculo absoluto. Adicionalmente, esta arquitetura suportou a operacionalização nativa de Inteligência Artificial Explicável (XAI), convertendo heurísticas de custo em transparência cognitiva para o utilizador.

Camada de Validação Determinística: A nível puramente técnico, o desenvolvimento e implementação do disjuntor de Restrições Rígidas resolveu de forma cabal a vulnerabilidade probabilística fundacional dos assistentes textuais. A avaliação empírica demonstrou que, no conjunto de cenários laboratoriais testados, não foram observadas alucinações logísticas, mitigando ativamente inconsistências e garantindo matematicamente a exequibilidade física e temporal de todas as rotas de contingência sugeridas face a disrupções.

Framework Laboratorial de Disrupções: Por fim, a dissertação entrega à comunidade científica e à indústria uma estrutura reproduzível para a simulação controlada de disrupções intermodais. O desenho da avaliação laboratorial concebido (com injeção

determinística de falhas e atrasos em *Gateways* de aviação e trânsito) permite medir latência, resiliência técnica e esforço operacional em sistemas críticos de *Smart Tourism*, sem os constrangimentos operacionais de incorrer em testes destrutivos com passageiros em ambientes de produção ao vivo.

8.3 Limitações do Estudo

O rigor científico exige o reconhecimento das fronteiras e constrangimentos inerentes ao desenvolvimento e avaliação de um Minimum Viable Product (MVP). É fundamental reiterar a delimitação do âmbito concetual deste projeto, o protótipo não visa substituir plataformas comerciais estabelecidas, como *Online Travel Agencies* (OTA) ou *Global Distribution Systems* (GDS), nem representa uma infraestrutura pronta para produção em larga escala. Adicionalmente, o *TravelMaster AI* não procura uma autonomia operacional irrestrita cega, o sistema atua como uma camada de inteligência tática suplementar que exige sempre supervisão e decisão final por parte do utilizador.

Assumindo estas fronteiras arquiteturais como base do desenvolvimento, a presente investigação apresenta ainda as seguintes limitações técnicas e metodológicas que devem ser consideradas na interpretação dos resultados empíricos.

A primeira limitação prende-se com a natureza da avaliação. A aferição da redução da carga cognitiva assentou numa Análise de Tarefas simulada em ambiente laboratorial. A ausência de ensaios empíricos com utilizadores reais num ambiente de aeroporto impossibilitou a captação das nuances emocionais e comportamentais de passageiros sob verdadeiro stress logístico, limitando a validação sociotécnica do grau de empatia e perceção de confiança gerados pelo sistema.

A segunda limitação advém da dependência crítica de infraestruturas de terceiros, uma vulnerabilidade inerente a sistemas orquestrados por terceiros. O plano original de desenvolvimento previa a integração síncrona com a API da Amadeus para a monitorização de voos em ambiente real. Contudo, durante a redação desta dissertação, o acesso público a este serviço foi descontinuado pelo fornecedor. Este imprevisto tecnológico forçou uma adaptação arquitetural para o uso de simuladores de injeção de atrasos. Embora esta adaptação tenha permitido validar rigorosamente a lógica do motor TTDP e o comportamento do LLM, inibiu o teste do sistema num ambiente de produção com tráfego aéreo ao vivo. Em paralelo, a dependência da API do modelo Gemini sujeitou o protótipo a flutuações de latência e limites de quota, exigindo a implementação do mecanismo de espera exponencial documentado no Capítulo 7.

Por fim, o escopo operacional do protótipo constitui uma limitação contextual. O sistema foi calibrado e testado face a cenários de viagens em infraestruturas de grande escala. O algoritmo heurístico (Restrições Flexíveis) poderá necessitar de afinação adicional para lidar com as especificidades operacionais de companhias *low-cost*, aeroportos regionais de pequena dimensão, ou transições complexas entre transporte aéreo e infraestruturas ferroviárias, contextos onde a métrica do *Minimum Connection Time* (MCT) apresenta maior variabilidade.

8.4 Trabalho Futuro

A conceção e avaliação do *TravelMaster AI* estabelecem uma prova de conceito robusta para a aplicação de arquiteturas neuro-simbólicas na gestão de disrupções turísticas. Não obstante, a complexidade inerente ao ecossistema global de transportes e a rápida evolução da Inteligência Artificial abrem um vasto leque de oportunidades para a expansão desta investigação. De forma a delimitar adequadamente a continuidade potencial do artefacto, perspetivam-se cinco eixos estruturantes de trabalho futuro:

Integração com APIs Reais em Produção. A prioridade técnica imediata consiste na transição do atual ambiente laboratorial (suportado por injeção simulada de falhas) para um ecossistema operacional ao vivo. A integração direta com APIs reais em produção, tais como infraestruturas da Amadeus ou FlightAware, e serviços rodoviários permitirá avaliar a resiliência arquitetural do sistema face a latências de rede imprevisíveis, limites de quota dinâmicos e dados não estruturados gerados em tempo real.

Avaliação Longitudinal e Estudos com Utilizadores Reais. O passo subsequente mais premente consiste na transição da avaliação analítica para ensaios clínicos em ambiente real. Sugere-se a condução de estudos empíricos com utilizadores reais em infraestruturas aeroportuárias, adotando uma abordagem de avaliação longitudinal da experiência do utilizador ao longo de múltiplas viagens. O recurso a ferramentas de aferição standardizadas (como a System Usability Scale (SUS) (Brooke 1996) ou o NASA Task Load Index (NASA-TLX) (Hart e Staveland 1988)) conjugadas com a recolha de telemetria biométrica, permitirá medir com validade ecológica a evolução da confiança na XAI e a mitigação do esforço operacional multidimensional.

Adaptação Baseada em Reinforcement Learning. Atualmente, as heurísticas do motor TTDP assentam em parâmetros de preferência estáticos. Futuras iterações devem incorporar mecanismos de adaptação baseada em *Reinforcement Learning*. Ao analisar proativamente o histórico de decisões, aceitações e rejeições de itinerários por parte do passageiro, o agente neuro-simbólico aprenderá dinamicamente as tolerâncias individuais ao risco (ex: aversão a escalas curtas ou preferência por aeroportos secundários), otimizando continuamente os pesos das Restrições Flexíveis.

Previsão Proativa de Disrupções e Integração Multimodal. A arquitetura deverá evoluir de um modelo essencialmente reativo para um modelo preditivo focado na previsão proativa de disrupções. A incorporação de modelos de *Machine Learning* treinados em séries temporais meteorológicas e de tráfego capacitará o Orquestrador para reencausar o utilizador muito antes do colapso logístico se materializar. Simultaneamente, este escopo preditivo exige uma integração multimodal holística, agregando infraestruturas ferroviárias de alta velocidade, transportes marítimos e micro-mobilidade urbana à matriz de opções de contingência.

Emissão Transacional e Interoperabilidade B2B. Atualmente, o *TravelMaster AI* opera como um agente de recomendação, exigindo que o passageiro concretize a alteração logística. O patamar último de evolução é a sua conversão num Agente Transacional Autónomo. Isto exigirá integrações Business-to-Business (B2B) avançadas, nomeadamente através do protocolo New Distribution Capability (NDC) da International Air Transport Association (IATA). Ao acoplar o sistema de *Tool Calling* a APIs de sistemas de pagamento e Global Distribution System (GDS), o agente executará autonomamente a remarcação e a reemissão do bilhete digital, com potencial para reduzir

substancialmente a fricção operacional da viagem em cenários de disrupção geridos pelo sistema.

A presente dissertação demonstra que o futuro da gestão de viagens não reside na passividade dos sistemas de agregação textuais, mas na proatividade de agentes neuro-simbólicos. Ao cruzar o rigor estrutural da Engenharia de *Software* com o potencial da Inteligência Artificial Generativa, este trabalho estabelece uma base fundacional robusta para que a verdadeira premissa do *Smart Tourism* se materialize, restituindo ao passageiro o controlo e a fluidez na experiência de viagem.

Referências

- Abuaisha, Abdallah, Bojie Shen, Daniel D. Harabor, Peter J. Stuckey e Mark Wallace (2025). «Dynamic Replanning for Improved Public Transport Routing». Em: *Proceedings of the 34th International Joint Conference on Artificial Intelligence (IJCAI 2025)*. Association for the Advancement of Artificial Intelligence (AAAI), pp. 8430–8437. doi: 10.24963/ijcai.2025/937.
- Adomavicius, G e A Tuzhilin (2011). «Context-aware recommender systems». Em: *Recommender systems handbook*. Springer, pp. 217–253.
- Bansal, Gagan, Besmira Nushi, Ece Kamar, Walter S. Lasecki, Daniel S. Weld e Eric Horvitz (2019). «Beyond Accuracy: The Role of Mental Models in Human-AI Team Performance». Em: *Proceedings of the Seventh AAAI Conference on Human Computation and Crowdsourcing*. Vol. 7, pp. 2–11. doi: 10.1609/hcomp.v7i1.5285. url: <https://ojs.aaai.org/index.php/HCOMP/article/view/5285>.
- Boer, Maaïke H. T. de, Quirine Smit, Michael van Bekkum, André Meyer-Vitali e Thomas Schmid (2025). «Design patterns for large language model based neuro-symbolic systems». Em: *Neurosymbolic Artificial Intelligence*. doi: 10.1177/29498732251377499.
- Brooke, John (1996). «SUS: A “quick and dirty” usability scale». Em: *Usability Evaluation in Industry*. Ed. por P. W. Jordan, B. Thomas, I. L. McClelland e B. Weerdmeester. Taylor & Francis, pp. 189–194.
- Buhalis, D e A Amaranggana (2015). «Smart tourism destinations: Enhancing tourism experience through personalisation of services». Em: *Information and Communication Technologies in Tourism 2015*. Ed. por I Tussyadiah e A Inversini. Cham: Springer, pp. 377–389. doi: 10.1007/978-3-319-14343-9_28.
- Card, Stuart K., Thomas P. Moran e Allen Newell (1980). «The keystroke-level model for user performance time with interactive systems». Em: *Communications of the ACM* 23.7, pp. 396–410. doi: 10.1145/358886.358895.
- Cavoukian, A (2009). *Privacy by Design: The 7 Foundational Principles*. Rel. téc. Information e Privacy Commissioner of Ontario.
- Chen, A, X Ge, Z Fu, Y Xiao e J Chen (2024). *TravelAgent: An AI assistant for personalized travel planning*. doi: 10.48550/arXiv.2409.08069.
- Chen, Michael K. (2025). «A Comparative Study of Neurosymbolic AI Approaches to Interpretable Logical Reasoning». Em: *Proceedings of the 19th Conference on Neurosymbolic Learning and Reasoning (NeSy 2025)*. Vol. 284. Proceedings of Machine Learning Research, pp. 1–16. url: <https://arxiv.org/abs/2508.03366>.
- Chen, Shize, Yifei Wang, Yifan Wu, Qian-Guang Chen, Zhiqiang Xu, Weiqi Luo, Kaining Zhang e Lin Zhang (2024). «Advancing tool-augmented large language models: Integrating insights from errors in inference trees». Em: *Advances in Neural Information Processing Systems*. Vol. 38. url: https://proceedings.neurips.cc/paper_files/paper/2024/file/c0f7ee1901fef1da4dae2b88dfd43195-Paper-Conference.pdf.
- Colelough, Brandon C. e William Regli (2024). «Neuro-Symbolic AI in 2024: A Systematic Review». Em: *Proceedings of the First International Workshop on Logical Foundations of*

- Neuro-Symbolic AI (LNSAI 2024)*. Vol. 3819. CEUR Workshop Proceedings. url: <https://ceur-ws.org/Vol-3819/paper3.pdf>.
- Ehsan, Upol e Mark O. Riedl (2020). «Human-centered Explainable AI: Towards a Reflective Sociotechnical Approach». Em: *International Conference on Human-Computer Interaction (HCI International)*. Springer, Cham, pp. 449–466. doi: 10.1007/978-3-030-60117-1_33. eprint: 2002.01092. url: <https://arxiv.org/abs/2002.01092>.
- EUROCONTROL (2024). *CODA Digest 2023: All-Causes Delay and Cancellations to Air Transport in Europe*. Technical Report. Brussels, Belgium: European Organisation for the Safety of Air Navigation. url: <https://www.eurocontrol.int/sites/default/files/2024-12/eurocontrol-coda-digest-annual-report-2023.pdf>.
- Felfernig, Alexander, Gerhard Friedrich, Dietmar Jannach e Markus Zanker (2015). «Constraint-based recommender systems». Em: *Recommender Systems Handbook*. Ed. por Francesco Ricci, Lior Rokach e Bracha Shapira. 2nd. Springer, pp. 161–190. doi: 10.1007/978-1-4899-7637-6_5.
- Fleiß, J., E. Bäck e S. Thalmann (2024). «Mitigating algorithm aversion in recruiting: A study on explainable AI for conversational agents». Em: *ACM Transactions on Intelligent Systems and Technology* 15.4. doi: 10.1145/3645057.
- Franklin, S e A Graesser (1997). «Is it an agent, or just a program? A taxonomy for autonomous agents». Em: *Intelligent agents III: Agent theories, architectures, and languages (ATAL 1996)*. Ed. por J P Müller, M J Wooldridge e N R Jennings. Vol. 1193. Lecture Notes in Computer Science. Springer. doi: 10.1007/BFb0013570.
- Gamma, Erich, Richard Helm, Ralph Johnson e John Vlissides (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
- Gavalas, Damianos, Charalampos Konstantopoulos, Konstantinos Mastakas e Grammati Pantziou (jun. de 2014). «A survey on algorithmic approaches for solving tourist trip design problems». Em: *Journal of Heuristics* 20.3, pp. 291–328. issn: 1381-1231. doi: 10.1007/s10732-014-9242-5. url: <http://link.springer.com/10.1007/s10732-014-9242-5>.
- Govea, J., R. Gutierrez e W. Villegas-Ch (2024). «Transparency and precision in the age of AI: Evaluation of explainability-enhanced recommendation systems». Em: *Frontiers in Artificial Intelligence* 7. doi: 10.3389/frai.2024.1410790.
- Gretzel, Ulrike, Marianna Sigala, Zheng Xiang e Chulmo Koo (set. de 2015). «Smart tourism: foundations and developments». Em: *Electronic Markets* 25.3, pp. 179–188. issn: 1019-6781. doi: 10.1007/s12525-015-0196-8. url: <https://link.springer.com/10.1007/s12525-015-0196-8>.
- Griffin, Amy L., Ian T. Ruginski, Wangshu Wang e Yinghui Cao (2024). «Cognitive issues of mobile map design and use». Em: *Journal of Location Based Services* 18.4, pp. 350–380. doi: 10.1080/17489725.2024.2371288.
- Hao, Yilun, Yongchao Chen, Yang Zhang e Chuchu Fan (2024). «Large Language Models Can Solve Real-World Planning Rigorously with Formal Verification Tools». Em: *arXiv preprint arXiv:2404.11891*. doi: 10.48550/arXiv.2404.11891. arXiv: 2404.11891 [cs.AI]. url: <https://arxiv.org/abs/2404.11891>.
- Hart, Sandra G. e Lowell E. Staveland (1988). «Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research». Em: *Human Mental Workload*. Ed. por P. A. Hancock e N. Meshkati. North-Holland, pp. 139–183.
- Hevner, Alan R, Salvatore T March, Jinsoo Park e Sudha Ram (mar. de 2004). «Design Science in Information Systems Research». Em: *MIS Quarterly* 28.1, pp. 75–106. issn: 0276-7783. doi: 10.2307/25148625. url: <https://misq.umn.edu/misq/article/28/1/75/261/Design-Science-in-Information-Systems-Research1>.

- Hitzler, Pascal e Md Kamruzzaman Sarker, eds. (2022). *Neuro-symbolic artificial intelligence: The state of the art*. Vol. 342. Frontiers in Artificial Intelligence and Applications. IOS Press. doi: 10.3233/FAIA342.
- Hou, Xinyi, Yanjie Zhao, Yue Liu, Zhou Yang, Kailong Wang, Li Li, Xiapu Luo, David Lo, John Grundy e Haoyu Wang (2024). «Large Language Models for Software Engineering: A Systematic Literature Review». Em: *ACM Transactions on Software Engineering and Methodology* 33.8. doi: 10.1145/3695988.
- Huang, Lei, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin e Ting Liu (2023). «A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions». Em: *ACM Transactions on Information Systems* 43.2, pp. 1–55. doi: 10.1145/3703155. arXiv: 2311.05232 [cs.CL]. url: <https://arxiv.org/abs/2311.05232>.
- Khatun, Tafura (2024). *Case Studies of Smart Destinations: Enhancing Tourist Experiences and Promoting Sustainability*. url: https://www.researchgate.net/publication/385312726_Case_Studies_of_Smart_Destinations_Enhancing_Tourist_Experiences_and_Promoting_Sustainability.
- Li, N. e H. G. Yao (2025). «A Review of Research on Flight Delay Propagation: Current Situation and Prospect». Em: *Journal of Advanced Transportation* 2025, p. 4851103. doi: 10.1155/atp/4851103.
- Lipton, Zachary C. (2018). «The mythos of model interpretability». Em: *Communications of the ACM* 61.10, pp. 36–43. doi: 10.1145/3233231.
- Manos, Anastasios, Despina Elisabeth Filipidou, Ioannis Deliyannis, Nikolaos Pavlidis, Vasilis Argyros e Ioanna Mazi (nov. de 2024). «INDIANA: Personalized Travel Recommendations Using Wearables and AI». Em: *arXiv preprint*. doi: 10.48550/arXiv.2411.12227. arXiv: 2411.12227. url: <http://arxiv.org/abs/2411.12227>.
- Milosevic, Zoran e Fethi Rabhi (2026). «Architecting Agentic Communities using Design Patterns». Em: *arXiv preprint arXiv:2601.03624*. url: <https://arxiv.org/abs/2601.03624>.
- Nunes, Ingrid e Dietmar Jannach (2017). «A systematic review and taxonomy of explanations in decision support and recommender systems». Em: *User Modeling and User-Adapted Interaction* 27.3–5, pp. 393–444. doi: 10.1007/s11257-017-9195-0.
- Nygard, Michael T. (2007). *Release It!: Design and Deploy Production-Ready Software*. Raleigh, NC: Pragmatic Bookshelf.
- Ojokoh, Bolanle Adefowoke, Folasade Olubusola Isinkaye, Ming Zhang, Joshua Joshua Tom, Arome Junior Gabriel, Olaitan Afolabi e Bamidele Afolabi (ago. de 2025). «Privacy and security in recommenders: an analytical review». Em: *Artificial Intelligence Review* 58.11, p. 351. issn: 1573-7462. doi: 10.1007/s10462-025-11333-4. url: <https://link.springer.com/10.1007/s10462-025-11333-4>.
- Oulasvirta, Antti, Sakari Tamminen, Virpi Roto e Jaana Kuorelahti (2005). «Interaction in 4-second bursts: The fragmented nature of attentional resources in mobile HCI». Em: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '05)*. ACM, pp. 919–928. doi: 10.1145/1054972.1055101.
- Pan, Liangming, Alon Albalak, Xinyi Wang e William Yang Wang (2023). «Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning». Em: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Singapore: Association for Computational Linguistics, pp. 3806–3824. doi: 10.48550/arXiv.2305.12295. url: <https://aclanthology.org/2023.findings-emnlp.248/>.

- Parasuraman, Raja e Dietrich H. Manzey (2010). «Complacency and Bias in Human Use of Automation: An Attentional Integration». Em: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 52.3, pp. 381–410. doi: 10.1177/0018720810376055.
- Park, S. e S. Kim (2020). «Multifaceted trust in tourism service robots». Em: *Annals of Tourism Research* 81, p. 102888. doi: 10.1016/j.annals.2020.102888.
- Peppers, Ken, Tuure Tuunanen, Marcus A. Rothenberger e Samir Chatterjee (2007). «A Design Science Research Methodology for Information Systems Research». Em: *Journal of Management Information Systems* 24.3, pp. 45–77. doi: 10.2753/MIS0742-1222240302.
- Rahimi Siegrist, Mahsa, Beda Büchel e Francesco Corman (2022). «Evaluating the Effects of Disruptions on the Behavior of Travelers in a Multimodal Network Utilizing Agent-Based Simulation». Em: *Transportation Research Record: Journal of the Transportation Research Board* 2676.9, pp. 651–663. doi: 10.1177/03611981221094005.
- Rao, Hongzhou, Yanjie Zhao, Xinyi Hou, Shenao Wang e Haoyu Wang (2025). «Software Engineering for Large Language Models: Research Status, Challenges and the Road Ahead». Em: *arXiv preprint arXiv:2506.23762*. url: <https://arxiv.org/abs/2506.23762>.
- Rasal, Sumedh e E J Hauer (jul. de 2024). «Navigating Complexity: Orchestrated Problem Solving with Multi-Agent LLMs». Em: *arXiv preprint*. doi: 10.48550/arXiv.2402.16713. arXiv: 2402.16713. url: <http://arxiv.org/abs/2402.16713>.
- Rebedea, Traian, Razvan Dinu, Makesh Narsimhan Sreedhar, Christopher Parisien e Jonathan Cohen (2023). «NeMo Guardrails: A Toolkit for Controllable and Safe LLM Applications with Programmable Rails». Em: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Singapore: Association for Computational Linguistics, pp. 431–445. doi: 10.18653/v1/2023.emnlp-demo.40. url: <https://aclanthology.org/2023.emnlp-demo.40>.
- Renaldi, F, H Suryanto, A Java Maulidin, I Santikarama e A Talib Bon (dez. de 2020). «Integrating travel booking management system using REST-API». Em: *Proceedings of the International Conference on Industrial Engineering and Operations Management*. Harare, Zimbabwe: IEOM Society International, pp. 3264–3274.
- Ribeiro, Marco Tulio, Sameer Singh e Carlos Guestrin (2016). «“Why Should I Trust You?”: Explaining the Predictions of Any Classifier». Em: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pp. 1135–1144. doi: 10.1145/2939672.2939778.
- Ricci, F, L Rokach e L Shapira (2015). *Recommender Systems Handbook*. Springer.
- Rubinstein, Joshua S., David E. Meyer e Jeffrey E. Evans (2001). «Executive Control of Cognitive Processes in Task Switching». Em: *Journal of Experimental Psychology: Human Perception and Performance* 27.4, pp. 763–797. doi: 10.1037/0096-1523.27.4.763.
- Russell, S e P Norvig (2021). *Artificial Intelligence: A Modern Approach*. 4th. Pearson.
- Scher, Sebastian, Harald Reiterer e Maksims Sedlins (2022). «The effect of transparency and trust on intelligent system acceptance: Evidence from a user-based study». Em: *Electronic Markets* 32.4, pp. 2079–2102. doi: 10.1007/s12525-022-00593-5.
- Schick, Timo, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda e Thomas Scialom (2023). «Toolformer: Language models can teach themselves to use tools». Em: *Advances in Neural Information Processing Systems*. Vol. 36, pp. 68539–68551. doi: 10.5555/3666122.3669119. url: https://proceedings.neurips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a9Abstract-Conference.html.
- Solano-Barliza, A, I Arregocés-Julio, M Aarón-González, R Zamora-Musa, E De-la-Hoz-Franco, J Escorcia-Gutierrez e M Acosta-Coll (2024). «Recommender systems applied

- to the tourism industry: A literature review». Em: *Cogent Business Management* 11.1, p. 2367088. doi: 10.1080/23311975.2024.2367088.
- Spadini, Davide, Maurício Aniche, Magiel Bruntink e Alberto Bacchelli (2019). «Mock objects for testing java systems». Em: *Proceedings of the 27th IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*.
- Starcke, Katrin e Matthias Brand (2012). «Decision Making Under Stress: A Selective Review». Em: *Neuroscience & Biobehavioral Reviews* 36.4, pp. 1228–1248. doi: 10.1016/j.neubiorev.2012.02.003.
- Su, Yi, Kexin Xie, Hongjian Wang, Zhe Liang, Wanpracha Art Chaovalitwongse e Panos M. Pardalos (2021). «Airline Disruption Management: A Review of Models and Solution Methods». Em: *Engineering* 7.4, pp. 435–447. doi: 10.1016/j.eng.2020.08.021.
- Suanpang, Pannee e Pattanaphong Pothipassa (ago. de 2024). «Integrating Generative AI and IoT for Sustainable Smart Tourism Destinations». Em: *Sustainability* 16.17, p. 7435. issn: 2071-1050. doi: 10.3390/su16177435. url: <https://www.mdpi.com/2071-1050/16/17/7435>.
- Sweller, John (1988). «Cognitive load during problem solving: Effects on learning». Em: *Cognitive Science* 12.2, pp. 257–285. doi: 10.1207/s15516709cog1202_4.
- Tan, Xiaopei e Muyang Fan (2025). «Thinking While Driving: A Concurrent Framework for Real-Time, LLM-Based Adaptive Routing». Em: *arXiv preprint arXiv:2512.10610*. doi: 10.48550/arXiv.2512.10610. url: <https://arxiv.org/abs/2512.10610>.
- Tintarev, Nava e Judith Masthoff (2015). «Explaining recommendations: Design and evaluation». Em: *Recommender Systems Handbook*. Ed. por Francesco Ricci, Lior Rokach e Bracha Shapira. 2nd. Springer, pp. 353–382. doi: 10.1007/978-1-4899-7637-6_10.
- Turpin, Miles, Julian Michael, Ethan Perez e Samuel R. Bowman (2023). «Language Models Don't Always Say What They Think: Unfaithful Explanations in Chain-of-Thought Prompting». Em: *Advances in Neural Information Processing Systems*. Vol. 36, pp. 74952–74965. doi: 10.48550/arXiv.2305.04388. url: <https://arxiv.org/abs/2305.04388>.
- Tussyadiah, Iis (2020). «A review of research into automation in tourism: Launching the Annals of Tourism Research Curated Collection on Artificial Intelligence and Robotics in Tourism». Em: *Annals of Tourism Research* 81, p. 102883. doi: 10.1016/j.annals.2020.102883.
- Vansteenkoven, Pieter, Wouter Souffriau, Greet Vanden Berghe e Dirk Van Oudheusden (2011). «The city trip planner: An expert system for tourists». Em: *Expert Systems with Applications* 38.6, pp. 6540–6545. doi: 10.1016/j.eswa.2010.11.085.
- Vouros, George A. (2022). «Explainable Deep Reinforcement Learning: State of the Art and Challenges». Em: *ACM Computing Surveys* 55.5, pp. 1–39. doi: 10.1145/3527448.
- Wang, Lei, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei e Ji-Rong Wen (2024). «A Survey on Large Language Model based Autonomous Agents». Em: *Frontiers of Computer Science*. doi: 10.1007/s11704-024-40231-1. arXiv: 2308.11432 [cs.AI]. url: <https://arxiv.org/abs/2308.11432>.
- Wang, Lei, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee e Ee-Peng Lim (2023). «Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models». Em: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Toronto, Canada: Association for Computational Linguistics, pp. 2609–2634. doi: 10.18653/v1/2023.acl-long.147. url: <https://aclanthology.org/2023.acl-long.147/>.
- Wardman, Mark (2004). «Public transport values of time». Em: *Transport Policy* 11.4, pp. 363–377. doi: 10.1016/j.tranpol.2004.05.001.

- Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed H. Chi, Quoc V. Le e Denny Zhou (2022). «Chain-of-Thought Prompting Elicits Reasoning in Large Language Models». Em: *Advances in Neural Information Processing Systems*. Ed. por Sanmi Koyejo, Shakir Mohamed, Anima Agarwal, Danielle Belgrave, Kyunghyun Cho e Alice Oh. Vol. 35, pp. 24824–24837. doi: 10.48550/arXiv.2201.11903. url: <https://arxiv.org/abs/2201.11903>.
- Werthner, Hannes e Francesco Ricci (dez. de 2004). «E-commerce and tourism». Em: *Communications of the ACM* 47.12, pp. 101–105. issn: 0001-0782. doi: 10.1145/1035134.1035141. url: <https://dl.acm.org/doi/10.1145/1035134.1035141>.
- Xie, J, K Zhang, J Chen, T Zhu, R Lou, Y Tian, Y Xiao e Y Su (2024). «TravelPlanner: A benchmark for real-world planning with language agents». Em: *Proceedings of the 41st International Conference on Machine Learning*. PMLR. doi: 10.48550/arXiv.2402.01622.
- Xu, M. e Y. Wang (2024). «Explainability increases trust resilience in intelligent agents». Em: *British Journal of Psychology*. Advance online publication. doi: 10.1111/bjop.12740.
- Yang, Seoyeon, ChaeYeon Park, Taewoo Kim e Jaegul Choo (2024). «Can tool-augmented large language models be aware of incomplete conditions?» Em: *arXiv preprint arXiv:2406.12307*. url: <https://arxiv.org/abs/2406.12307>.
- Yao, Shunyu, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan e Yuan Cao (2023). «ReAct: Synergizing Reasoning and Acting in Language Models». Em: *Proceedings of the 11th International Conference on Learning Representations (ICLR 2023)*. url: <https://arxiv.org/abs/2210.03629>.
- Zhang, Yongfeng e Xu Chen (2020). «Explainable Recommendation: A Survey and New Perspectives». Em: *Foundations and Trends® in Information Retrieval* 14.1, pp. 1–101. issn: 1554-0669. doi: 10.1561/15000000066. url: <http://www.nowpublishers.com/article/Details/INR-066>.

Apêndice A

Instruções do Sistema do Modelo de Linguagem (System Prompts)

O principal pilar da fiabilidade e reprodutibilidade de sistemas baseados em Agentes Autónomos e *Large Language Models* (LLMs) reside na definição estrita das suas diretivas de sistema. Esta secção apresenta o texto literal utilizado para parametrizar o agente orquestrador do *TravelMaster AI* (`llm_agent.py`), garantindo o determinismo operativo e o alinhamento com os objetivos de mitigação da carga cognitiva.

A.1 Diretiva Base do Orquestrador (System Instruction)

O bloco seguinte apresenta as instruções de sistema injetadas no modelo para guiar o seu comportamento discursivo e operativo:

```
1 És o TravelMaster AI, um assistente de viagens inteligente e empático.
2 A tua missão é reduzir a sobrecarga cognitiva do utilizador durante a
   viagem.
3 Quando fores informado de um atraso, NUNCA peças ao utilizador para
   procurar alternativas.
4 Deves usar IMEDIATAMENTE a tua ferramenta de pesquisa de voos para
   encontrar soluções, e apresentar as opções de forma calma, clara e
   tranquilizadora, explicando por que são boas escolhas.
```

Listing A.1: Diretiva de sistema literal do Orquestrador AI.

A.2 Metadados de Configuração da Inferência

Para assegurar o comportamento reprodutível documentado neste trabalho, o motor de inferência foi instanciado com os seguintes hiperparâmetros de computação:

- **Modelo de Base:** `gemini-2.5-flash`
- **Temperatura:** 0.3 (Fixada para baixa entropia, minimizando a criatividade e contribuindo para a redução de alucinações de rotas).
- **Mecanismo de Resiliência:** *Exponential Backoff* ativo com limite superior de 3 tentativas (`max_retries = 3`) e atraso base de 2 segundos.
- **Controlo de Ferramentas (Tool Binding):** Injeção direta da assinatura funcional da rotina `search_viable_flight_alternatives`.

Apêndice B

Estruturas de Dados de Simulação (Mock Payloads)

De forma a validar a arquitetura sem incorrer em custos proibitivos de infraestrutura ou dependências de conectividade com APIs comerciais em ambiente de teste, foram desenvolvidos simuladores de telemetria. Esta secção documenta as estruturas de dados canónicas (esquemas JSON) utilizadas para injetar disrupções controladas no sistema.

B.1 Payload de Evento de Disrupção Aérea (Flight Status Update)

O JSON seguinte representa a resposta simulada emitida pelo `amadeus_api.py` quando interrogado sobre o estado do voo original (Cenário de Atraso), evidenciando a mutação entre o horário calendarizado (*scheduled*) e o estimado (*estimated*):

```
1 {  
2   "flight_status_event": {  
3     "flight_number": "UA 180",  
4     "carrier_code": "UA",  
5     "route": {  
6       "departure_airport": "HKG",  
7       "arrival_airport": "JFK"  
8     },  
9     "temporal_data": {  
10      "scheduled_departure_date": "2021-07-18",  
11      "scheduled_departure_time": "16:15",  
12      "estimated_departure_time": "16:45",  
13      "calculated_delay_minutes": 30  
14    },  
15    "operational_status": "DELAYED",  
16    "disruption_severity": "MINOR",  
17    "timestamp": "2026-04-08T14:43:03Z"  
18  }  
19 }
```

Listing B.1: Payload JSON de atualização de estado de voo (atraso).

B.2 Payload de Resposta de Pesquisa de Alternativas (Flight Search Results)

O seguinte bloco de dados ilustra o *array* estruturado que o simulador injeta no motor de recomendação (*recommender.py*) quando o agente autônomo invoca a ferramenta de pesquisa de rotas secundárias. Esta estrutura crua fornece o mapeamento que será posteriormente submetido às restrições físicas do filtro determinístico TTDP:

```
1 [
2   {
3     "flight_number": "CX 846",
4     "carrier_code": "CX",
5     "departure_airport": "HKG",
6     "arrival_airport": "JFK",
7     "departure_date": "2021-07-18",
8     "departure_time": "18:30",
9     "available_seats": 9,
10    "cabin_class": "ECONOMY"
11  },
12  {
13    "flight_number": "CX 848",
14    "carrier_code": "CX",
15    "departure_airport": "HKG",
16    "arrival_airport": "JFK",
17    "departure_date": "2021-07-18",
18    "departure_time": "23:55",
19    "available_seats": 4,
20    "cabin_class": "ECONOMY"
21  },
22  {
23    "flight_number": "BR 827",
24    "carrier_code": "BR",
25    "departure_airport": "MFM",
26    "arrival_airport": "JFK",
27    "departure_date": "2021-07-18",
28    "departure_time": "20:00",
29    "available_seats": 2,
30    "cabin_class": "ECONOMY"
31  }
32 ]
```

Listing B.2: Payload JSON de resultados da pesquisa de voos alternativos.

B.3 Payload de Estado do Trânsito Terrestre (Terrestrial Route Status)

Abaixo encontra-se a estrutura gerada pelo *maps_api.py* para simular uma falha crítica na malha rodoviária urbana (Cenário de Acidente), fornecendo o tempo de trânsito em minutos que colide com a janela de segurança aeroportuária:

```
1 {
2   "matrix_routing_response": {
3     "origin_identifier": "HOTEL_HK",
4     "destination_identifier": "HKG",
5     "baseline_duration_minutes": 35,
6     "current_duration_minutes": 85,
```

```
7   "traffic_model_applied": "pessimistic",  
8   "road_status_incident": "SEVERE_ACCIDENT",  
9   "route_viable": true  
10  }  
11 }
```

Listing B.3: Payload JSON de simulação de estado de trânsito terrestre com incidente severo.

Apêndice C

Artigo Científico Submetido

Este anexo apresenta a versão integral do artigo científico desenvolvido no âmbito desta dissertação, intitulado "*Neuro-Symbolic Orchestration for Autonomous Travel Execution: Mitigating LLM Hallucinations via Deterministic Constraints*". O artigo foi formatado segundo as normas IEEE e submetido para publicação.

Neuro-Symbolic Orchestration for Autonomous Travel Execution: Mitigating LLM Hallucinations via Deterministic Constraints

Abstract—The tactical execution of travel is chronically hindered by operational fragmentation, forcing passengers to manage multiple disconnected platforms during disruptions. While Generative AI offers conversational utility, pure Large Language Models (LLMs) fail critically in complex routing tasks, achieving a mere 0.6% success rate on the TravelPlanner benchmark due to persistent logical hallucinations and an inability to satisfy rigid constraints. To address this gap, this paper introduces TravelMaster AI, an autonomous monitoring and rerouting agent developed under the Design Science Research (DSR) methodology. The proposed solution employs a hybrid neuro-symbolic architecture that segregates semantic comprehension—delegated to an LLM (Gemini 2.5 Flash)—from mathematical execution. We implement a deterministic Tourist Trip Design Problem (TTDP) logic engine acting as a circuit breaker to prevent impossible itineraries, coupled with a subtractive heuristic algorithm to minimize spatial and temporal friction. Rigorous laboratory evaluations (N=30) demonstrate a highly stable average processing latency of 12.54 seconds (95% CI [11.61s, 13.47s]) under combined terrestrial-aerial disruption, with a complementary isolated terrestrial-only evaluation (N=30) confirming a predictable latency reduction of approximately 25.8% (9.30s) in lower-complexity scenarios. Furthermore, analytical assessment using the Keystroke-Level Model (KLM) confirmed a systemic reduction in user operational effort, decreasing required interactions from 7-10 scattered manual actions to a unified 1-2 click process. Ultimately, this research proves that confining LLM stochasticity within strict deterministic boundaries yields a highly reliable agent capable of drastically reducing cognitive load during dynamic travel disruptions.

Index Terms—Autonomous Agents, Neuro-Symbolic AI, Large Language Models, Design Science Research, Smart Tourism, Disruption Management.

I. INTRODUCTION

Modern travel execution remains a cognitively demanding and highly fragmented process. During unforeseen operational disruptions, passengers are forced into an intensive context-switching loop across isolated applications (e.g., airline portals, mapping tools, email clients) to recalculate route viability. This multi-platform orchestration imposes a severe cognitive load, degrading human decision-making under acute stress.

While Large Language Models (LLMs) have transformed pre-trip inspiration and conversational interfaces, their application in real-time, dynamic tactical execution exposes fundamental structural flaws. Thus, the transition from static conversational assistants to reliable, autonomous execution agents requires a paradigm shift, bridging the gap between natural language processing and deterministic logistical operations.

II. RELATED WORK

The evolution of digital travel assistance has historically been dominated by Business-to-Consumer (B2C) static platforms functioning primarily as passive data aggregators. Market-leading solutions excel at pre-trip documentary organization and generalist inspiration through collaborative filtering. However, these systems inherently lack operational autonomy during the critical tactical execution phase of a journey. Upon encountering real-time logistical disruptions, these static architectures force the passenger to engage in severe manual orchestration across a fragmented ecosystem.

While the recent integration of pure Generative Artificial Intelligence (GenAI) and LLMs has revolutionized conversational inspiration, it exhibits critical structural limitations in spatial and temporal orchestration. As empirically demonstrated by the comprehensive TravelPlanner benchmark [1], state-of-the-art pure language agents achieve a catastrophic final success rate of merely 0.6% when tasked with resolving real-world, multi-constraint Tourist Trip Design Problems (TTDP). This failure is rooted in the inherently probabilistic nature of LLMs, which rely on statistical distribution sampling rather than mathematical grounding, leading inevitably to logical and fidelity hallucinations. They consistently synthesize temporally impossible itineraries and incompatible spatial connections. Therefore, to achieve reliable autonomous disruption management, literature dictates that the stochastic creativity of LLMs must be structurally confined by deterministic validation mechanisms.

III. METHODOLOGY & ARCHITECTURE

This study adopts the Design Science Research (DSR) methodology [2] to construct and evaluate a novel IT artifact aimed at solving the operational fragmentation problem. To resolve the probabilistic vulnerabilities of pure GenAI, the proposed framework employs a neuro-symbolic orchestration architecture that strictly segregates semantic interaction from mathematical constraint resolution.

A. System Components and Design Patterns

The architectural integrity of the TravelMaster AI framework is anchored in a strict adherence to the Single Responsibility Principle (SRP), ensuring that the stochastic volatility of Generative AI is structurally decoupled from deterministic business logic. The codebase is compartmentalized into highly specialized modules. The `llm_agent.py` component acts

exclusively as the central orchestrator and semantic interface; it interprets multimodal inputs and orchestrates interactions but is architecturally deprived of any mathematical decision-making autonomy. All physical constraint verifications are delegated to `ttdp_logic.py`, which serves as the symbolic TTDP engine, while `recommender.py` functions as the heuristic scoring module responsible for optimizing the operational experience.

To manage external data ingestion and ensure experimental rigor, the architecture implements two fundamental design patterns. The Facade pattern is utilized to encapsulate the complex HTTP communication and data normalization from external environments (e.g., `amadeus_api.py`), providing a unified interface to the orchestrator. Crucially, to align with DSR laboratory evaluation requirements, the framework employs Mock Objects to replace volatile production APIs. By abstracting aviation and terrestrial telemetry through controlled mock gateways, the system guarantees strict reproducibility. This isolation allows for the precise, artificial injection of network latency and severe disruption payloads (as shown in Listing 1), ensuring that the neuro-symbolic tool-calling mechanisms are evaluated under controlled stress conditions.

```

1 {
2   "flight_status_event" {
3     "flight_number" "UA 180",
4     "operational_status" "DELAYED",
5     "calculated_delay_minutes" 30,
6     "disruption_severity" "MINOR"
7   }
8 }

```

Listing 1. Simulated API disruption payload acting as the system trigger.

B. Multimodal Ingestion and Semantic Routing

The operational lifecycle initiates with the extraction of the traveler's unstructured ticket context using the Gemini 2.5 Flash model. To prevent structural hallucinations during this ingestion phase, the architecture enforces strict data contracts using the Pydantic library.

During execution, the architecture utilizes Semantic Routing via Python function signatures and Docstrings, which act as algorithmic triggers. When the LLM interprets the disruption context, the docstring instructs the inference engine to halt natural language generation and emit a formal Tool Call, delegating the spatial and temporal recalculation entirely to the deterministic optimizer.

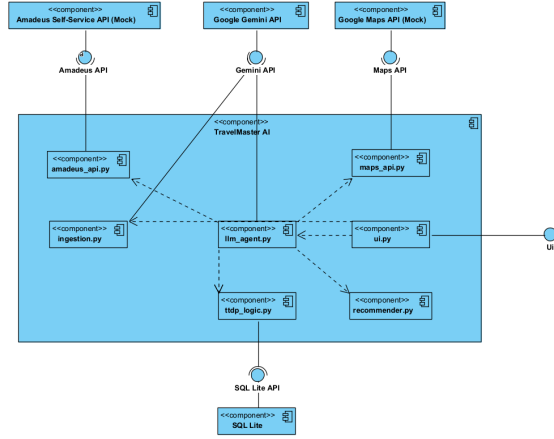


Fig. 1. Component Architecture of the TravelMaster AI Neuro-Symbolic Framework.

IV. ALGORITHMIC IMPLEMENTATION

To mitigate the logical hallucinations endemic to pure LLMs, the backend employs a strict mathematical formulation divided into rigid physical boundaries (Hard Constraints) and user-centric optimization (Soft Constraints).

A. Hard Constraints: The Logical Circuit Breaker

The `ttdp_logic.py` module acts as a boolean circuit breaker. For any alternative route proposed by the external data gateway, the engine calculates the temporal delta between the estimated arrival and the subsequent departure. If the available time margin is mathematically less than the airport's strict Minimum Connection Time (MCT), the boolean flag `is_viable` evaluates to `False`, and the route is instantly pruned from the model's context array. Listing 2 illustrates this deterministic implementation.

```

1 def validate_connection(arrival, departure, mct):
2     available_time = calculate_delta(arrival,
3                                     departure)
4
5     if available_time >= mct:
6         return {"is_viable": True, "margin":
7               available_time - mct}
8     else:
9         return {"is_viable": False, "margin": 0}

```

Listing 2. Deterministic TTDP circuit breaker implementation.

B. Soft Constraints: Subtractive Heuristic

Once the hard constraints are satisfied, the `recommender.py` module applies a subtractive heuristic function to the remaining viable options to minimize passenger friction. Every viable alternative is assigned an initial base score of 100 points ($S_{base} = 100$). The algorithm then penalizes this score across two friction axes:

Spatial Friction: If the alternative itinerary requires a change of airport infrastructure ($Airport_{departure} \neq$

$Airport_{origin}$), a severe penalty of 40 points is deducted, relegating such high-stress transfers to a last-resort option.

Temporal Friction: To account for terminal waiting fatigue, a linear degradation penalty is applied based on the extra waiting time beyond the required MCT. For every complete block of 30 minutes, 1 point is subtracted, as formulated in Equation 1:

$$Penalty_{temporal} = \left\lfloor \frac{\Delta t_{extra_wait}}{30} \right\rfloor \quad (1)$$

This mathematical formulation ensures that the system algorithmically prioritizes the path of least physical and cognitive resistance, appending these calculations as Explainable AI (XAI) strings to anchor the LLM’s final response.

V. EVALUATION & RESULTS

The empirical evaluation of the artifact was conducted under rigorous laboratory conditions designed to test the limits of the orchestrator’s resilience and efficiency.

A. Experimental Setup and Scenario C

The benchmark for system efficacy was established through “Scenario C” (Severe Multimodal Disruption). In this stress test, a synthetic user faces a simultaneous 85-minute terrestrial traffic delay and a 30-minute departure delay on their original flight. Upon detecting this dual trigger, the deterministic TTDP engine recalculates the physical viability of the route, identifying that the remaining 20-minute margin critically violates the airport’s 90-minute MCT. The TTDP engine acts as a strict logical circuit breaker, autonomously invoking the recommender tool to calculate and validate a downstream rescue flight.

B. Latency and Operational Stability

The operational stability of this autonomous replanning is captured in Table I. Over $N = 30$ independent stress executions of Scenario C, the orchestrator demonstrated highly consistent processing speeds despite the complex background tool calling.

TABLE I
LATENCY DESCRIPTIVE STATISTICS ($N = 30$ EXECUTIONS)

Statistical Metric	Value (Seconds)
Minimum Latency	8.24 s
Maximum Latency	19.26 s
Mean Latency (μ)	12.54 s
Standard Deviation (σ)	2.59 s
95% Confidence Interval	12.54 $\pm$ 0.93 s

Crucially, the absolute maximum recorded time (19.26s) remained comfortably below the strict 30-second operational threshold required to provide actionable alerts before physical airport departure boards update.

To isolate the computational overhead introduced by concurrent multimodal disruption handling, a complementary evaluation ($N = 30$) was conducted using an isolated terrestrial-only trigger (the “Severe Accident” scenario), removing the

simultaneous flight-delay component present in Scenario C. Table II summarizes these results.

TABLE II
LATENCY DESCRIPTIVE STATISTICS: ISOLATED TERRESTRIAL DISRUPTION ($N = 30$ EXECUTIONS)

Statistical Metric	Value (Seconds)
Minimum Latency	6.88 s
Maximum Latency	15.21 s
Mean Latency (μ)	9.30 s
Standard Deviation (σ)	2.44 s
95% Confidence Interval	9.30 $\pm$ 0.87 s

Comparing both conditions reveals that the isolated terrestrial trigger completed, on average, 3.24 seconds faster than the combined Scenario C (9.30s vs. 12.54s, a reduction of approximately 25.8%). This gap is consistent with the additional orchestration overhead required to resolve two concurrent disruption sources—terrestrial and aerial—within a single reasoning cycle, including the extra Tool Calling round-trip and constraint-checking pass against both the traffic and flight-status payloads. This comparison reinforces that the architecture’s latency scales predictably with disruption complexity, rather than exhibiting erratic behavior under compounded stress conditions.

C. Extended Evaluation: Edge Cases and Sensitivity

Beyond catastrophic disruptions, the system’s operational sensitivity was rigorously stress-tested against boundary conditions, notably the “Innocuous Delay” (False Positive) scenario. Minor perturbations were artificially injected into the telemetry (e.g., a 15-minute traffic delay).

When the Orchestrator intercepted this anomaly, it delegated the spatial-temporal parameters to the TTDP engine. The deterministic calculation verified that the total available margin still comfortably exceeded the MCT threshold, returning an `is_viable=True` status. Crucially, this positive mathematical validation acted as a logical blockade, intentionally suppressing the Semantic Routing trigger and preventing the LLM from executing a Tool Call to search for alternative flights. By forcing this deliberate computational inaction, the orchestrator defaulted to a purely conversational stance, reassuring the user. This architectural specificity not only prevents unnecessary panic but also drastically optimizes resource allocation by saving API inference tokens.

D. KLM Task Analysis and Cognitive Load Reduction

To quantify the reduction in cognitive load, a Keystroke-Level Model (KLM) [3] task analysis (Table III) contrasts the physical and mental friction of traditional disruption management against the autonomous agent.

VI. DISCUSSION AND THREATS TO VALIDITY

While the DSR methodology establishes the technical viability of the neuro-symbolic architecture, critical Threats to Validity must be acknowledged.

TABLE III
KLM TASK ANALYSIS: OPERATIONAL EFFORT COMPARISON

Execution Phase	Traditional Method	TravelMaster AI
<i>Detection</i>	Open multiple apps	Passive monitoring
<i>Assessment</i>	Mental calculation	TTDP validation
<i>Resolution</i>	Search, evaluate, book	One-click via XAI
Total Interactions	7 to 10 scattered actions	1 to 2 unified clicks

A primary internal and external threat is the reliance on Mock Gateways. Although simulating APIs ensures reproducibility and fault injection, it naturally suppresses the unpredictable volatility, rate-limiting, and unstructured data anomalies characteristic of live production environments (e.g., Amadeus or FlightAware). Furthermore, there is a construct validity limitation regarding the assessment of cognitive load. The Keystroke-Level Model (KLM) is an excellent analytic metric for quantifying interaction reduction, but it structurally fails to capture the psychosomatic anxiety and emotional stress experienced by a human in a chaotic airport environment.

Additionally, the sample size ($N=30$) constrains the statistical power of the latency analysis, while sufficient to establish a stable confidence interval, larger-scale replication would be required to generalize these findings beyond the tested scenario. Notably, the consistent and proportional latency reduction observed in the isolated terrestrial evaluation (Section IV-B) across an independent $N=30$ sample lends additional confidence to the stability of the measurement methodology itself, as the two datasets exhibit coherent, non-overlapping confidence intervals proportional to disruption complexity.

VII. CONCLUSION AND FUTURE WORK

The TravelMaster AI architecture demonstrates that the profound limitations of Generative AI in multi-constraint logistical planning can be resolved without merely scaling neural networks. By structurally confining an LLM within a deterministic TTDP circuit breaker, the proposed neuro-symbolic system successfully mitigates the 0.6% failure rate endemic to pure language agents.

Consequently, Future Work must prioritize empirical validation and systemic expansion. Subsequent evaluations should transition from analytic KLM metrics to real-world biometric stress tests with human subjects to accurately gauge trust in the XAI outputs. Architecturally, the system is primed for an evolution toward predictive orchestration, utilizing Machine Learning on meteorological and traffic time-series data to reroute passengers proactively. Additionally, the static subtractive heuristic should be upgraded via Reinforcement Learning (RL), allowing the agent to dynamically fine-tune spatial and temporal weights based on individual travel preferences. Ultimately, the system's final operational frontier requires integrating Business-to-Business (B2B) protocols such as IATA's New Distribution Capability (NDC) to transcend

conversational assistance and independently orchestrate the autonomous re-ticketing of disrupted itineraries.

ACKNOWLEDGMENT

The author acknowledges the use of the Gemini large language model for assistance in formatting, translating, and refining the phrasing of this manuscript. Furthermore, as detailed in Section III, Gemini 2.5 Flash is explicitly implemented as the core semantic processing component within the TravelMaster AI architecture. The author assumes full responsibility for the final content, original ideas, and scientific validity presented in this work.

REFERENCES

- [1] J. Xie et al., "TravelPlanner: A Benchmark for Real-World Planning with Language Agents," *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.
- [2] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design Science in Information Systems Research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.
- [3] S. K. Card, T. P. Moran, and A. Newell, "The Keystroke-Level Model for User Performance Time with Interactive Systems," *Communications of the ACM*, vol. 23, no. 7, pp. 396–410, 1980.

Apêndice D

Comprovativo de Submissão do Artigo

Este anexo contém o comprovativo oficial gerado pela plataforma EasyChair, atestando a submissão do artigo científico na *CASCON 2026 (36th IEEE International Conference on Collaborative Advances in Software and Computing)*.



[User guide](#) / [Log out](#)

New Submission	Submission 56	Help	Conference	EasyChair
----------------	---------------	------	------------	-----------

CASCON 2026 Submission 56

Your submission has been saved and is shown at the bottom of this page. You can now also declare conflicts of interest for this submission.

A conflict of interest means any circumstances that can potentially prevent a PC member from writing a fair review or comment about your submission.

All declared conflicts will be viewed by CASCON 2026 chairs, who may decide to add or to ignore them.

Submission Information

Submission 56	
Title	Neuro-Symbolic Orchestration for Autonomous Travel Execution: Mitigating LLM Hallucinations via Deterministic Constrain
Paper	 (Jun 22, 11:55)
Track	Technical Track
Author keywords	Autonomous Agents Neuro-Symbolic AI Large Language Models Design Science Research Smart Tourism Disruption Management.
Topics	Software Engineering
Abstract	The tactical execution of travel is chronically hindered by operational fragmentation, forcing passengers to manage multiple disconnected platforms during disruptions. While Generative AI offers conversational utility, pure Large Language Models (LLMs) fail critically in complex routing tasks, achieving a mere 0.6% success rate on the TravelPlanner benchmark due to persistent logical hallucinations and an inability to satisfy rigid constraints. To address this gap, this paper introduces TravelMaster AI, an autonomous monitoring and rerouting agent developed under the Design Science Research (DSR) methodology. The proposed solution employs a hybrid neuro-symbolic architecture that segregates semantic comprehension—delegated to an LLM (Gemini 2.5 Flash)—from mathematical execution. We implement a deterministic Tourist Trip Design Problem (TTDP) logic engine acting as a circuit breaker to prevent impossible itineraries, coupled with a subtractive heuristic algorithm to minimize spatial and temporal friction. Rigorous laboratory evaluations (N=30) demonstrate a highly stable average processing latency of 12.54 seconds (95% CI [11.61s, 13.47s]) under combined terrestrial-aerial disruption, with a complementary isolated terrestrial-only evaluation (N=30) confirming a predictable latency reduction of approximately 25.8% (9.30s) in lower-complexity scenarios. Furthermore, analytical assessment using the Keystroke-Level Model (KLM) confirmed a systemic reduction in user operational effort, decreasing required interactions from 7-10 scattered manual actions to a unified 1-2 click process. Ultimately, this research proves that

Submitted	confining LLM stochasticity within strict deterministic boundaries yields a highly reliable agent capable of drastically reducing cognitive load during dynamic travel disruptions.
Last update	Jun 22, 11:55
Address	quinta centenario, rua 25 de abril Entrada 7, apt P21, 2º andar Vila Real, 5000-062 Portugal
Author conflicts	none

Authors						
First name	Last name	Email	Country	Affiliation	Web page	Corresponding?
João	Cordeiro	1240466@isep.ipp.pt	Portugal	Instituto Superior de Engenharia do Porto		✓
Piedade	Carvalho	PBC@isep.ipp.pt	Portugal	Instituto Superior de Engenharia do Porto		