SISTEMA DE GERENCIAMENTO DO DEPARTAMENTO DE EXIBIÇÃO DE EMISSORA DE TV

TV STATION BROADCASTING DEPARTMENT MANAGEMENT SYSTEM

Augusto Navarro Infante de Andrade Vasconcelos[1*]
Bruno Santos de Lima[2**]

Resumo

Este trabalho apresenta o desenvolvimento do Sistema de Gerenciamento do Controle Mestre (SGCM), uma aplicação web criada para centralizar e otimizar a gestão operacional do departamento de exibição de uma emissora de televisão. O contexto inicial caracterizava-se pela gestão fragmentada de processos vitais, como a elaboração de escalas de trabalho, o controle de férias e a gestão da grade de programação, realizados por meio de múltiplas planilhas manuais e *softwares* locais isolados. A metodologia adotada baseou-se em um levantamento de requisitos junto aos operadores e no desenvolvimento iterativo da solução utilizando o framework Laravel. Como resultado, o sistema integrou essas funções em uma única plataforma, introduzindo módulos inovadores, como a "Afinação" de telejornais e cronômetros de estúdio ("*Timers*") sincronizados em tempo real através do navegador. Adicionalmente, a ferramenta automatiza a geração de escalas rotativas, permitindo a exportação em PDF e o envio direto por e-mail. Conclui-se que a transição para uma arquitetura web integrada resultou em uma melhoria na usabilidade, reduzindo o tempo gasto em tarefas administrativas, minimizando erros operacionais e garantindo uma maior rastreabilidade das ações através de registros de auditoria (logs).

Palavras-chave: sistema web, automação, escalas de trabalho, cronômetros de estúdio, gestão operacional.

Abstract

This paper presents the development of the Master Control Management System (SGCM), a web application created to centralize and optimize the operational management of a television broadcasting department. The initial context was characterized by the fragmented management of vital processes, such as shift scheduling, vacation tracking, and broadcast

[1*]Aluno do curso de Tecnologia em Análise e Desenvolvimento de Sistemas, da Fatec de Presidente Prudente. Email: augusto.vasconcelos@aluno.cps.sp.gov.br

[2**] Professor orientador Me. em Ciência da Computação, da Faculdade de Tecnologia de Presidente Prudente. E-mail: bruno.lima01@cps.sp.gov.br

grid management, which were performed using multiple manual spreadsheets and isolated local software. The adopted methodology was based on requirements gathering with operators and the iterative development of the solution using the Laravel framework. As a result, the system integrated these functions into a single platform, introducing innovative modules, such as real-time news program "Tuning" (Afinação) and studio "Timers" synchronized via the web browser. Furthermore, the tool automated the generation of rotating shift schedules, allowing for PDF export and direct distribution via email. It is concluded that the transition to an integrated web architecture resulted in a improvement in usability, reducing the time spent on administrative tasks, minimizing operational errors, and ensuring greater traceability of actions through audit logs.

***Keywords**: web system, automation, shift scheduling, studio timers, operational management*

1. INTRODUÇÃO

A gestão operacional em emissoras de televisão exige extrema precisão, especialmente no setor de exibição, onde o cumprimento de horários e a coordenação de equipes são vitais para a continuidade da programação. Tradicionalmente, muitos desses processos são geridos de forma fragmentada, utilizando ferramentas de uso geral que não se comunicam entre si.

No cenário que motivou este trabalho, a operação dependia de um ecossistema baseado em múltiplas planilhas eletrônicas isoladas (LAUDON; LAUDON, 2014). Havia um arquivo para o controle de escalas de trabalho, outro para a grade de programação local dos fins de semana e um terceiro para o registro de férias dos operadores. Essa fragmentação gerava atrasos operacionais, como a necessidade de preenchimento redundante de dados e o risco de erros de digitação e sincronização entre as informações de disponibilidade da equipe e as escalas geradas.

Além da gestão administrativa, as ferramentas operacionais de tempo real também apresentavam limitações. O sistema de contagem regressiva para estúdio (*timers*) consistia em um *software* local desenvolvido em C#, o que restringia o seu uso à máquina local e apresentava limitações, pois não contemplava o tempo de volta dos comerciais e dificultava a acessibilidade em diferentes pontos da rede interna da empresa. Já o processo de "afinação" de telejornais — o ajuste dos tempos das laudas conforme o jornal acontece ao vivo — era feita no próprio sistema de exibição, porém com a atualização recente do sistema, essa função foi perdida, logo, se fez necessária a busca por alternativas que suprissem essa necessidade do setor de exibição que permitisse o lançamento e monitoramento ágil pelo operador.

Para mitigar essas dificuldades, este trabalho apresenta o SGCM (Sistema de Gerenciamento do Controle Mestre). O sistema foi desenvolvido sobre o ecossistema Laravel 12, adotando uma arquitetura web modular que centraliza as funções antes dispersas. A inovação do projeto reside na unificação de processos administrativos e operacionais: o SGCM não apenas gerencia a grade e as férias, mas também oferece os módulos de Afinação e *Timers* acessíveis via navegador, eliminando a dependência de *softwares* locais.

O objetivo central deste artigo é detalhar o processo de modernização dessa infraestrutura, com foco na usabilidade e na centralização das informações. Demonstra-se como a automação da geração de escalas rotativas e a integração do envio de escalas por e-mail e PDF reduzem o trabalho manual dos operadores. Por fim, discute-se como a transição para uma arquitetura web hospedada em servidor interno (*self-hosted*) melhora a experiência do usuário e prepara o terreno para futuras integrações automáticas entre o sistema de exibição (*playout*) e a função de "Regressiva" do módulo "*Timers*" do sistema.

2. FUNDAMENTAÇÃO

A fundamentação teórica aborda os conceitos essenciais para compreender o contexto operacional de uma emissora de TV e as decisões técnicas de engenharia de *software* envolvidas no desenvolvimento do SGCM.

2.1 O Papel do Controle Mestre em Emissoras de TV

O Controle Mestre, ou *Master Control*, é o ponto final da cadeia de produção antes da distribuição do sinal ao telespectador. Segundo Albarran (2016), esse setor atua como o "funil" de uma emissora, onde a supervisão em tempo real é vital para a garantia da continuidade operacional. Qualquer interrupção pode resultar em perdas financeiras diretas e danos à credibilidade da marca. Portanto, sistemas que apoiam essa operação devem priorizar a precisão cronométrica e a rapidez na tomada de decisão (ZETTL, 2014).

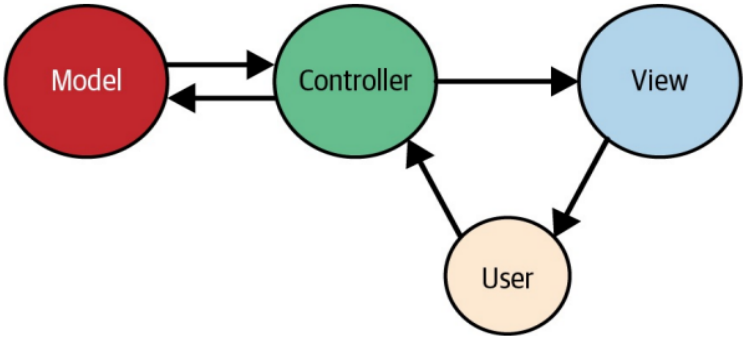
2.2 Da Gestão Fragmentada à Integração de Sistemas

Historicamente, a operação de emissoras de médio porte dependeu de "ilhas de informação" — planilhas e *softwares* locais que não se comunicam. Laudon e Laudon (2014)

explicam que a fragmentação de dados gera redundância e aumenta a probabilidade de erro humano. A transição para sistemas integrados permite que a informação flua entre os departamentos (como o RH inserindo férias e o Operacional gerando a escala), criando o que se define como uma "fonte única da verdade", essencial para a eficiência administrativa (PRESSMAN; MAXIM, 2016).

2.3 Desenvolvimento Web Moderno com o Padrão MVC

Para a construção do SGCM, utilizou-se o framework Laravel, que implementa o padrão de arquitetura *Model-View-Controller* (MVC). A Figura 1 apresenta esse modelo de maneira simplificada. De acordo com Stauffer (2019), o MVC separa a lógica de negócios da interface do usuário, o que facilita a manutenção e a escalabilidade do sistema. O Laravel eleva esse padrão ao oferecer recursos nativos como o *Eloquent ORM*, que simplifica a interação com o banco de dados, e o motor de templates *Blade*, que permite a criação de



interfaces dinâmicas e responsivas de forma eficiente (LARAVEL, 2026).

Figura 1 - Arquitetura Model-View-Controller (MVC) simplificada.
Fonte: Stauffer (2019)

2.4 Auditoria, Logs e Confiabilidade

Em sistemas que gerenciam operações críticas e escalas de trabalho, a rastreabilidade é um requisito não funcional indispensável. Sommerville (2019) destaca que a confiabilidade de um sistema está diretamente ligada à sua capacidade de auditoria. A implementação de registros detalhados (*logs*) não serve apenas para apurar erros, mas também para garantir o "não-repúdio" e a conformidade com as normas internas da organização, permitindo que cada

alteração na grade ou nos *timers* seja associada a um usuário e carimbo de tempo específico.

2.5 Arquitetura de Contêineres e Portabilidade

A adoção do Docker no desenvolvimento do SGCM justifica-se pela necessidade de paridade de ambientes. Segundo a literatura de infraestrutura moderna, diferente da virtualização, onde cada máquina virtual necessita de um sistema operacional completo, sendo gerenciado por um virtualizador, já o Docker possibilita a criação de ambientes independentes e isolados. A principal diferença é que o container compartilha o *Kernel* com o sistema operacional do *host*, aumentando o desempenho e diminuindo o consumo de memória, isolando a aplicação de dependências do sistema operacional hospedeiro (TURNBULL, 2014). A Figura 2 ilustra as diferenças da virtualização e conteinerização.

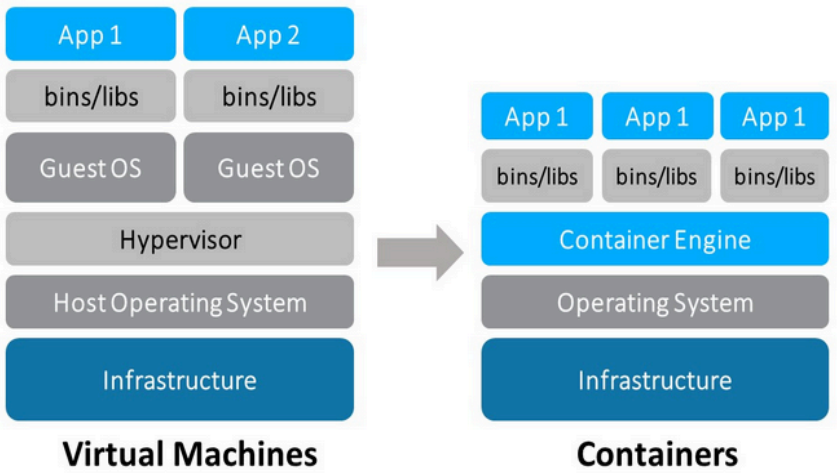


Figura 2 – Diferenças de Arquiteturas entre Máquinas Virtuais e Containers.
Fonte: Khan (2022)

Essa mudança de paradigma traz dois conceitos importantes:

- O container só irá consumir os recursos necessários para rodar o serviço.
- Cada container é uma aplicação isolada, sendo possível replicá-lo para qualquer ambiente que contenha o Docker instalado.

Com isso, aceleramos o *deploy* e a recuperação de desastres em ambientes de missão crítica (TURNBULL, 2014).

2.6 Usabilidade e Acessibilidade em Sistemas Web

Diferente de *softwares* locais que exigem instalação e configurações específicas em cada terminal, a arquitetura web baseada em navegadores democratiza o acesso à informação dentro da rede interna. Conforme apontam os princípios de UX (*User Experience*), a centralização em uma plataforma web melhora a usabilidade ao oferecer uma interface familiar e consistente em qualquer ponto de acesso, reduzindo a curva de aprendizado dos operadores (KRUG, 2014) e facilitando atualizações globais no sistema sem interrupção do serviço local (PRESSMAN; MAXIM, 2016).

3. METODOLOGIA

Esta seção descreve os procedimentos adotados para o desenvolvimento e implantação do SGCM. Trata-se de uma pesquisa aplicada, de abordagem qualitativa e quantitativa, com desenvolvimento tecnológico e avaliação por meio de questionário aplicado junto aos usuários do sistema.

3.1 Levantamento de Requisitos

A fase inicial do projeto consistiu em entrevistas e observação direta junto aos operadores do Controle Mestre da Band Paulista. Foram identificadas as principais necessidades:

- Controle automatizado de escalas de trabalho nos formatos de turnos de 6 e 8 horas;
- Padronização do processo de afinação de telejornais;
- Temporizador de estúdio compartilhado e em tempo real;
- Gestão centralizada de férias dos operadores;
- Cadastro de programas locais e independentes exibidos aos finais de semana;
- Geração de relatórios e envio de escalas por e-mail;
- Registro completo de ações dos usuários para auditoria.

3.2 Arquitetura de Tecnologias

O SGCM foi desenvolvido como uma aplicação web empregando as seguintes tecnologias, divididas em camadas:

- Backend: Laravel 12, PHP 8.2
- Frontend: Blade Templates, CSS (Bootstrap 5.3), JavaScript (Vanilla)
- Banco de Dados: MySQL / MariaDB
- Autenticação e Permissões: Laravel Auth + Spatie Laravel Permission
- Build: Vite + Node.js
- Containerização e *Deploy*: Docker + Docker Compose

A arquitetura segue o padrão **MVC (*Model-View-Controller*)**, aprimorado com a implementação da **Camada de Serviços (*Service Layer*)** para garantir o Princípio da Responsabilidade única (*SRP*). A Figura 3 mostra como o sistema está dividido nas seguintes camadas lógicas.

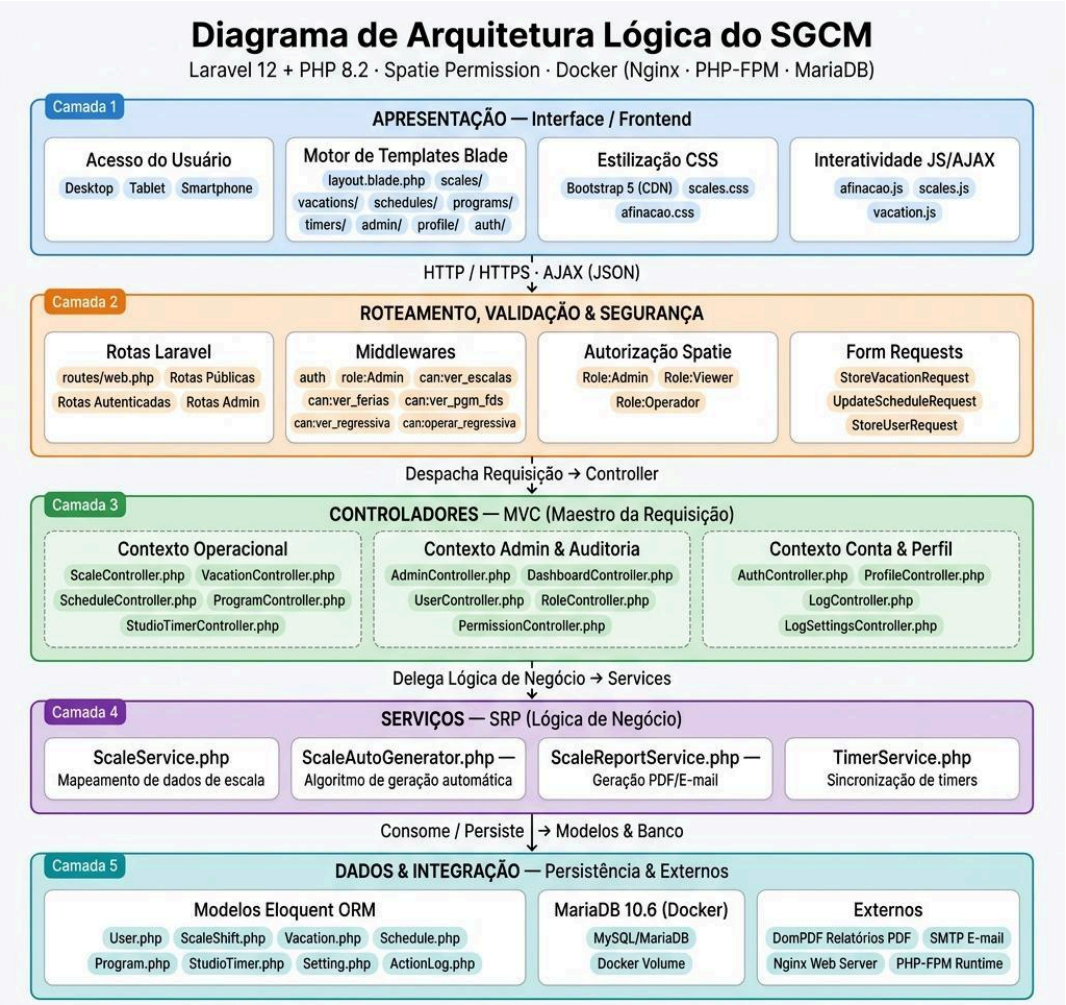


Figura 3 – Arquitetura Lógica do Sistema.

Fonte: Autor.

3.2.1 Camada de Apresentação (Interface com o Usuário / *Frontend*)

Responsável por renderizar a interface e gerenciar a interatividade no navegador do cliente.

- **Motor de Templates:** Blade (padrão Laravel), organizado em layouts e componentes (ex: layout.blade.php)
- **Estilização:** Bootstrap 5 (via CDN) combinado com CSS customizado (ex: scales.css, afinação.css).
- **Interatividade e Assincronismo:** JavaScript Vanilla (ex: afinacao.js, scales.js, vacation.js) responsável por fazer requisições AJAX (via Fetch API) para atualizar os *Timers* e Afinação em tempo real sem recarregar a página.

3.2.2 Camada de Roteamento, Validação e Segurança

Atua como a "porta de entrada" e o segurança do sistema.

- **Roteamento:** O arquivo routes/web.php mapeia URLs HTTP para os respectivos controladores.
- **Autorização e Autenticação:** Uso do *Spatie Laravel Permission* para proteger rotas baseado em *Roles* (Admin, Viewer, Operador) e *Permissions* (ex: ver_escalas, usar_afinacao).
- **Form Requests (Validação Segura):** Classes dedicadas que limpam os Controladores validando os dados antes deles entrarem no sistema (ex: StoreVacationRequest, UpdateScheduleRequest, StoreUserRequest).

3.2.3 Camada de Controle (*Controllers*)

Atuam como os maestros do sistema. Recebem a requisição, chama os Serviços ou Modelos adequados, e devolvem a resposta (HTML ou JSON). O sistema possui 14 Controladores principais, divididos por contexto.

Contexto Operacional:

- **ScaleController:** Gerencia a visualização, edição e fluxos da geração/criação

de escalas de trabalho.

- **VacationController:** Gerencia o CRUD e aprovações de férias.
- **ScheduleController** e **ProgramController:** Gerenciam a grade de programação local de fins de semana (PGMs FDS).
- **StudioTimerController:** Fornece os *endpoints* JSON para os cronômetros regressivos/progressivos e de tempo do break (intervalo).

Contexto Administrativo e Auditoria:

- **AdminController** e **DashboardController:** Painéis com métricas do sistema (Admin) e visão geral (Operador/Viewer) respectivamente.
- **UserController**, **RoleController** e **PermissionController**: Gestão completa de usuários, grupos e permissões sistêmicas.
- **LogController** e **LogSettingsController:** Visualização e exportação (PDF) do rastreamento de ações de auditoria.

Contexto de Conta:

- **AuthController** e **ProfileController:** Gestão de login, logout e alteração de senha/dados do usuário logado.

3.2.4 Camada de Regras de Negócio (Services Layer)

Esta camada isola a complexidade pesada, evitando que os Controladores fiquem sobrecarregados (o que evita o "Fat Controller").

- **ScaleService:** Processa a lógica de buscar os turnos e operadores no banco e organizar a estrutura visual dos dias na memória.
- **ScaleAutoGenerator:** Contém o algoritmo matemático complexo que faz a rotação de operadores (turnos 1, 2, 3, Madrugada e Folga) analisando o dia anterior.
- **ScaleReportService:** Cuida de empacotar as escalas, convertê-las em PDF e despachar os e-mails por SMTP.
- **TimerService:** Gerencia a lógica de tempo servidor vs. tempo local para

manter os relógios dos *timers* sincronizados em múltiplos computadores.

3.2.5 Camada de Dados (Modelos *Eloquent ORM*) e Integração (Infraestrutura, Serviços Externos)

Utiliza o Eloquent ORM para mapear o banco de dados relacional em objetos PHP, utilizando recurso de *Soft Deletes* (para evitar exclusão acidental de dados operacionais).

- **Modelos principais:** User, ScaleShift, Vacation, Schedule, Program, StudioTimer, Setting, ActionLog (este último sendo crucial para a auditoria contínua de ações no sistema).
- **Banco de Dados:** MySQL / MariaDB contendo as tabelas/dados do sistema.
- **Geração de Relatórios:** Biblioteca DomPDF (utilizada para Escalas e Logs).
- **Comunicação:** Servidor SMTP integrado para envio de e-mails via classe ScaleShipped.
- **Conteinerização (Ambiente):** O sistema inteiro opera encapsulado no Docker (*docker-composer.yml*, *Dockerfile*), utilizando um proxy reverso Nginx e Contêineres PHP-FPM, garantindo que o ambiente de produção seja idêntico ao de desenvolvimento.

3.3 Estrutura Modular

O sistema foi organizado nos seguintes módulos conforme o Quadro 1:

Quadro 1 - Estrutura Modular do SGCM

Módulo	Descrição
Dashboard	Visão geral dos turnos do usuário logado e datas de folga/retorno.
Afinação de Jornais	Gerenciamento da afinação de telejornais.
Timers	Cronômetro e regressiva em tempo real para operadores e diretores de imagem.
Escalas	Criação, edição e geração automática de escalas de turnos (6h/8h), geração de PDF e envio de escalas por e-mail.

Programação	Cadastro de programas locais e independentes da grade.
Férias	Controle de períodos de férias.
Logs	Rastreamento completo de ações dos usuários (com filtros por módulo, usuário e data) e geração de relatórios em PDF.
Administração	Dashboard administrativo com métricas, gerenciamento de usuários, papéis e permissões.

Fonte: Autor

Ressalta-se que o módulo de Escalas possui uma limitação operacional deliberada. O algoritmo de geração automática de escalas (ScaleAutoGenerator) foi desenvolvido para o cenário padrão da emissora, no qual todos os operadores ativos cumprem turnos de 6 horas. Nesse contexto, o sistema distribui os turnos de forma rotativa e automática, respeitando folgas e evitando conflitos.

Todavia, quando há operadores em férias ou de atestado, a equipe disponível fica reduzida, e o regime de trabalho de todos os envolvidos é automaticamente convertido para turnos de 8 horas em alguns dias específicos. Nessa situação, a complexidade do escalonamento aumenta significativamente, e o sistema não oferece geração automática. A escala deve então ser montada manualmente por um operador, utilizando as ferramentas de edição do módulo de Escalas. Essa restrição é conhecida e está prevista para ser tratada em versões futuras do SGCM.

3.4 Testes Unitários

Em um ambiente crítico como o Controle Mestre de uma emissora de televisão, falhas sistêmicas ou erros na atribuição de escalas podem resultar em prejuízos operacionais graves e comprometer a continuidade da transmissão. Para mitigar esses riscos e garantir a estabilidade do SGCM, foi implementada uma robusta suíte de testes automatizados utilizando o PHPUnit, a ferramenta de testes padrão integrada no ecossistema Laravel (LARAVEL, 2026).

A estratégia de testes foi dividida em duas abordagens principais: testes unitários (Unit Tests) e testes de funcionalidade/integração (Feature Tests), garantindo a cobertura tanto da lógica de negócio isolada quanto do fluxo completo de interação do usuário.

No âmbito dos **testes unitários**, o foco principal recaiu sobre as classes de serviço, com destaque especial para o "ScaleAutoGenerator" e o "ScaleService". A geração automática de escalas rotativas é o processo matematicamente mais complexo do sistema. Os testes unitários desenvolvidos para esse módulo simulam diversos cenários críticos — como a ausência de um operador no turno da madrugada anterior, a identificação do operador que está de folga e a validação de equipes incompletas, como visto na Figura 4.

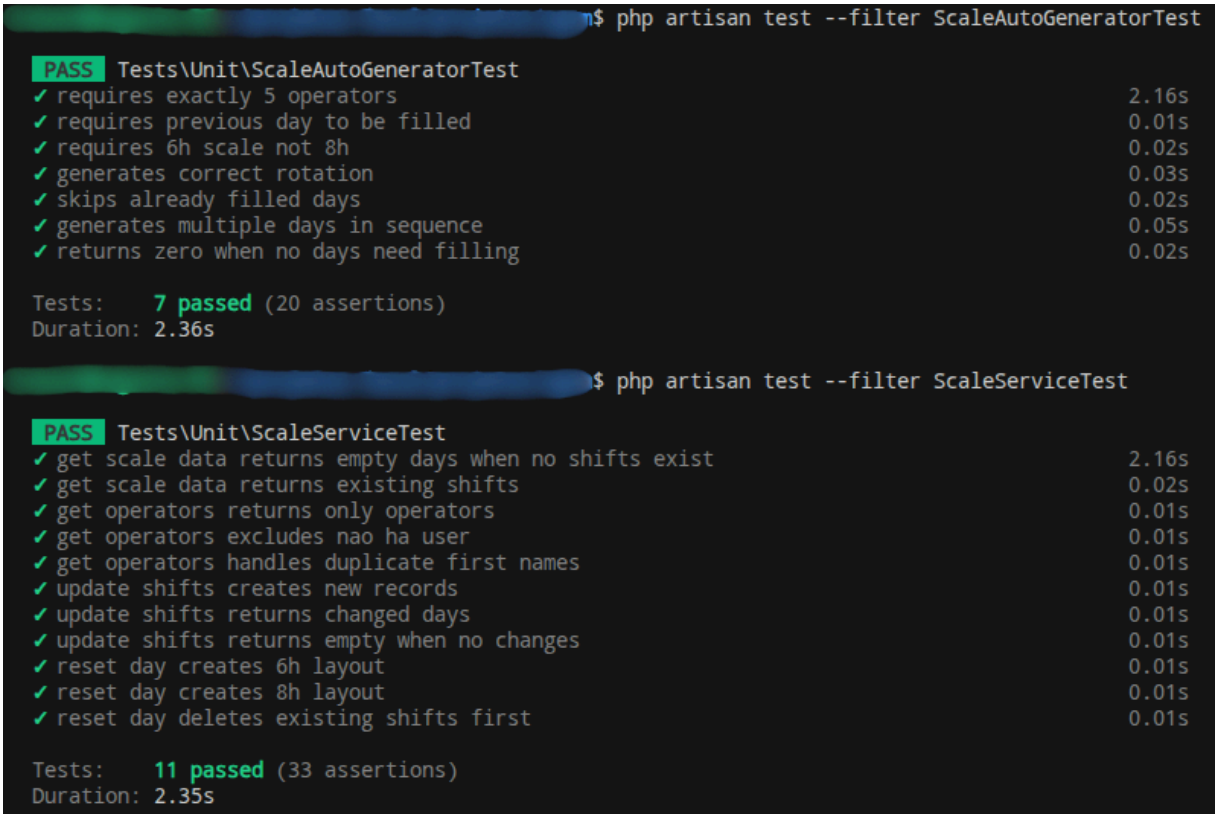


Figura 4 – Testes Unitários das Classes "ScaleService" e "ScaleAutoGenerator".

Fonte: Autor.

Essas validações automatizadas asseguram que o algoritmo distribua os turnos de forma justa e em total conformidade com as regras trabalhistas e de descanso da equipe, impedindo que o sistema gere escalas inválidas.

Por outro lado, os **testes de integração (Feature Tests)** foram aplicados para validar o comportamento dos controllers, rotas e regras de autorização. Módulos sensíveis como o de Férias (VacationControllerTest) e o de Escalas (ScaleControllerTest) foram submetidos a

testes que simulam requisições HTTP reais. Esses testes garantem, por exemplo, que apenas usuários com privilégios de "Administrador" ou o próprio titular do registro consigam editar um período de férias, validando a eficácia dos *Form Requests* e das políticas de segurança do Laravel. A Figura 5 mostra os testes realizados.

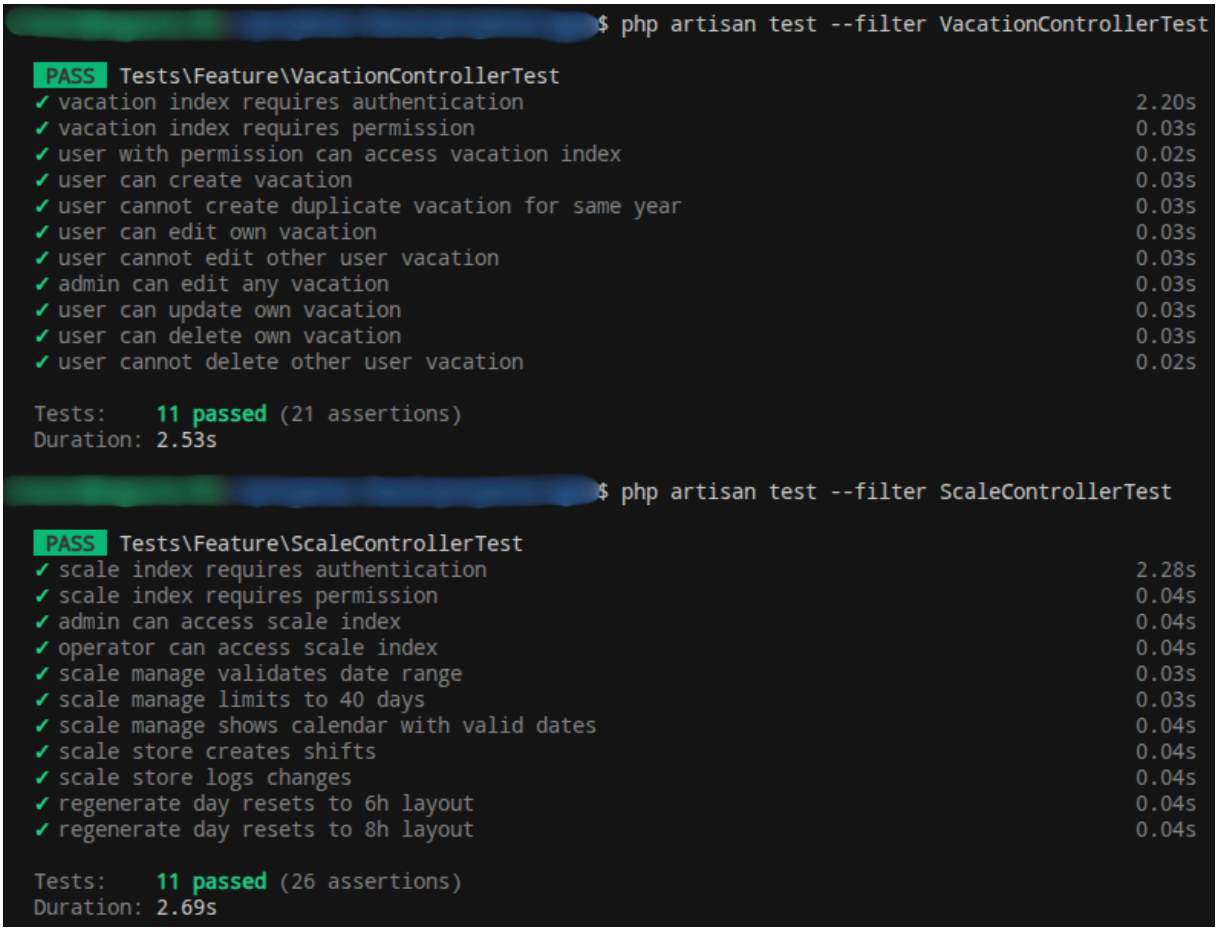


Figura 5 – Testes de Integração dos controladores “VacationController” e “ScaleController”.

Fonte: Autor.

A implementação dessa camada de testes revelou-se fundamental durante o ciclo de desenvolvimento. Ao garantir que as regras de negócio basilares estão protegidas por testes automatizados, obteve-se a segurança necessária para refatorar o código — como a transição de lógicas complexas dos *Controllers* para uma arquitetura baseada em *Services* — sem o receio de introduzir *bugs* no ambiente de produção (FOWLER, 2019).

3.5 Processo de Desenvolvimento e Implantação

O desenvolvimento seguiu uma abordagem iterativa, com entregas incrementais de módulos. Cada módulo foi testado em ambiente de testes, com a participação de operadores do Controle Mestre, antes de finalização e liberação para produção. Ao final da implantação do sistema, foi realizado um treinamento presencial com a equipe, e o sistema entrou em produção, recebendo feedback e alguns ajustes posteriores.

4. RESULTADOS

Após a implantação do SGCM no Controle Mestre da Band Paulista, foram observados ganhos significativos em eficiência, confiabilidade e rastreabilidade. Para mensurar esse impacto, foi aplicado um questionário de avaliação (ver Apêndice A)(VASCONCELOS, 2026) junto aos 5 operadores do setor, englobando profissionais com experiência variando de 1 a mais de 5 anos na emissora. Embora os resultados indiquem elevada aceitação do sistema, ressalta-se que a avaliação foi realizada com um grupo reduzido de usuários, correspondente à equipe diretamente envolvida na operação do Controle Mestre.

4.1 Aceitação e Impacto Operacional

Os resultados da pesquisa de satisfação demonstraram uma aceitação unânime da nova ferramenta. Quando questionados sobre a avaliação geral do sistema, 100% dos respondentes atribuíram a nota máxima (10/10). Além disso, todos afirmaram que recomendariam o SGCM para outros setores ou emissoras "com certeza", como mostra a Figura 6.

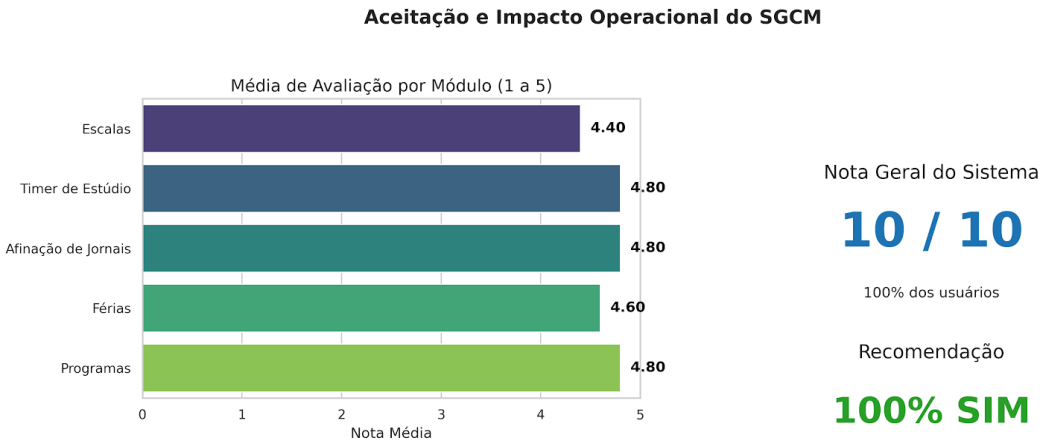


Figura 6 – Resultados gerais de aceitação e avaliação média dos módulos do SGCM.
Fonte: Elaborado pelo autor com dados da pesquisa.

Na avaliação específica por módulos (em uma escala de 1 a 5, onde 5 representa "Muito Satisfeito"), todos os componentes do sistema — incluindo *Timers*, Afinação, Escalas e Férias — obtiveram médias de aprovação entre 4,40 e 4,80 (Figura 6). Nas respostas discursivas, os usuários destacaram que a "agilidade" e a "concentração de tudo no mesmo local" foram os maiores benefícios trazidos pela plataforma.

4.2 Automação das escalas de trabalho

Anteriormente, a elaboração da escala semanal de turnos era feita manualmente, utilizando planilhas isoladas. Segundo os dados levantados no questionário, os responsáveis chegavam a gastar entre 30 minutos e mais de 1 hora semanalmente apenas para elaborar, conferir e ajustar essas escalas. Além do tempo despendido, os operadores relataram que erros como sobreposição de horários ou conflitos com férias ocorriam "às vezes" ou "frequentemente".

Com a adoção do SGCM e sua geração automática, o tempo gasto para essas mesmas tarefas foi drasticamente reduzido, com relatos de conclusão em "menos de 5 minutos". O impacto mais expressivo, contudo, deu-se na confiabilidade dos dados: 100% dos usuários relataram que, após a implantação do sistema, a frequência de erros nas escalas geradas caiu para "Nunca", conforme visto na Figura 7.

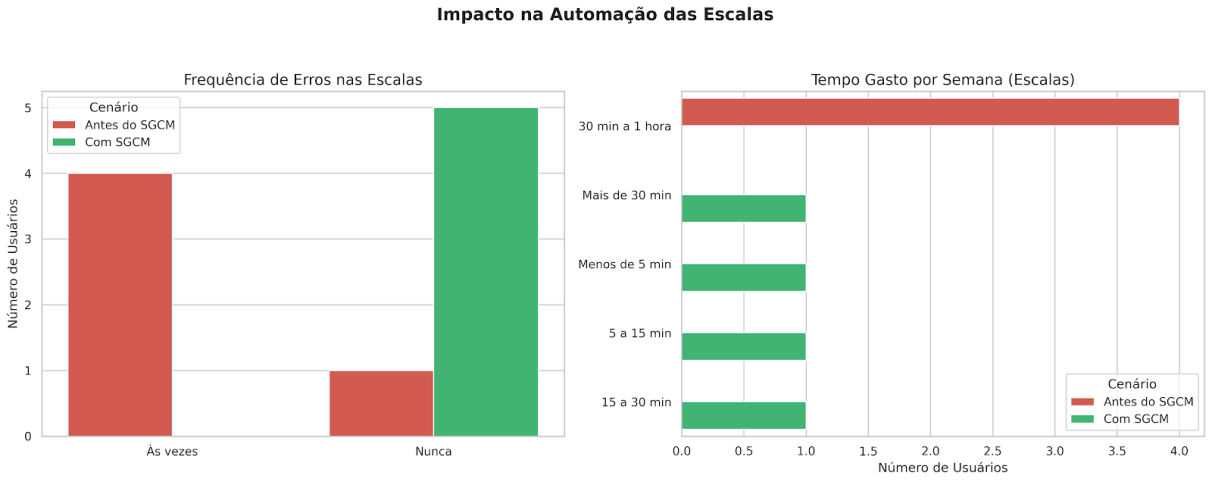


Figura 7 – Comparativo de incidência de erros e tempo gasto na elaboração de escalas antes e após o SGCM.

Fonte: Elaborado pelo autor com dados da pesquisa.

A opção de enviar a escala por e-mail em PDF também eliminou a necessidade de impressão e distribuição manual, consolidando o fluxo digital. A Figura 8 mostra a tela de interface de geração de escalas com dias especificados e turnos já preenchidos. Cabe notar que a geração automática descrita é aplicável apenas ao cenário padrão de turnos de 6 horas sem afastamentos. Quando há férias ou atestados, a equipe opera em regime de 8 horas, e a escala deve ser elaborada manualmente pelos operadores.

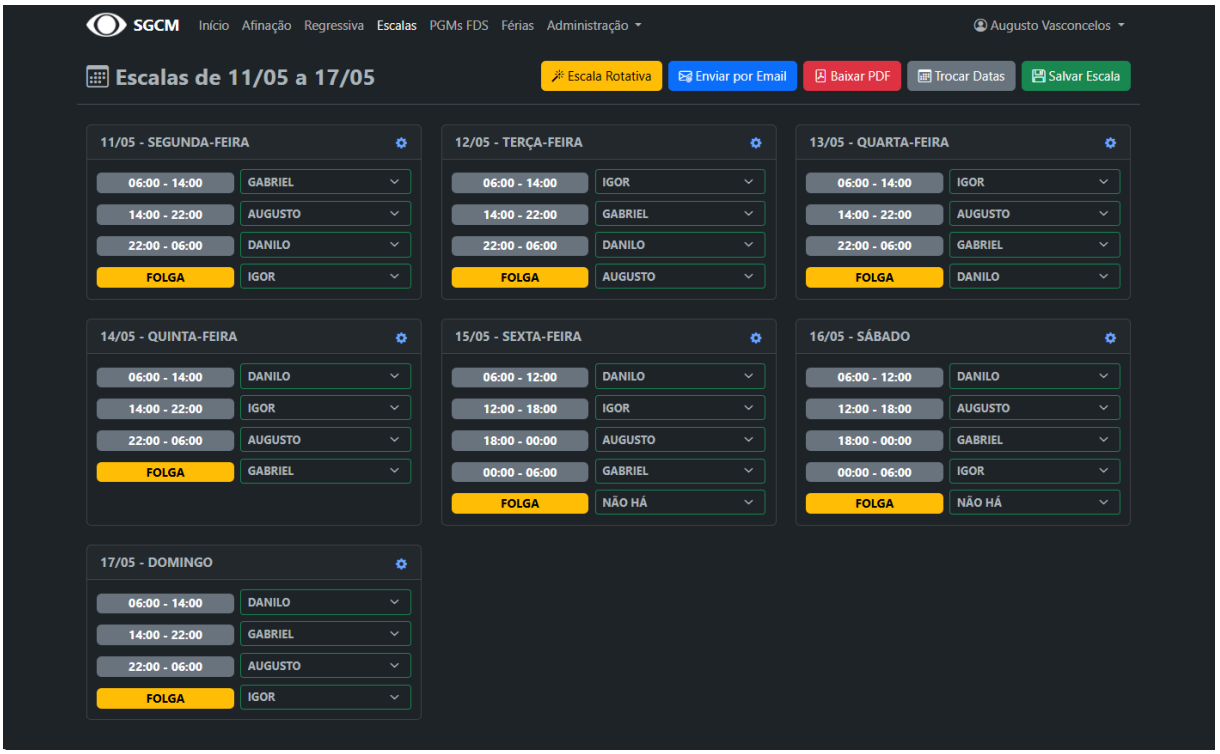


Figura 8 – Interface do Módulo de Escalas com datas já definidas e turnos preenchidos.

Fonte: Autor.

4.3 Timer de estúdio

A disponibilização de um cronômetro e contagem regressiva acessível via navegador (qualquer computador da rede interna) substituiu o uso de cronômetros físicos independentes. Utilizando o recurso de "Visualização Dividida" disponibilizado por navegadores que utilizam o "motor" do navegador de código aberto Chromium, é possível utilizar os módulos de Afinação e *Timers* durante os jornais, como visto na Figura 9.

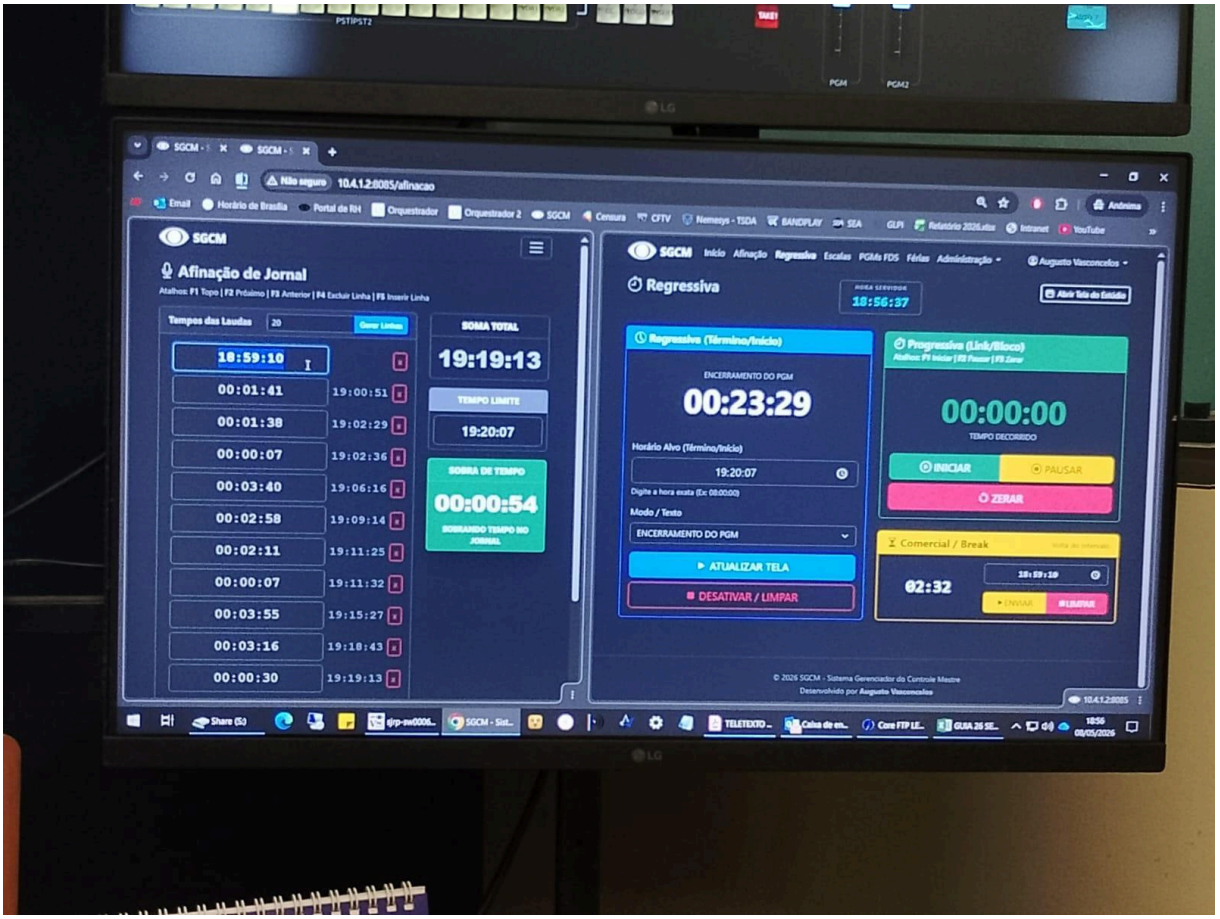


Figura 9 – Módulos de Afinação e *Timers* sendo usados no jornal ao vivo.

Fonte: Band Paulista.

Os operadores e diretores de imagem passaram a compartilhar o mesmo *timer*, sincronizado pelo servidor, reduzindo a comunicação excessiva entre o Controle Mestre e a Switcher de Produção, mantendo-a relacionada apenas a problemas ou avisos durante os programas ao vivo. A Figura 10 mostra o timer regressivo ativo durante um programa ao vivo dentro da Switcher de Produção.

Figura 10 – Timer regressivo indicando quanto tempo falta para acabar o programa ao vivo.
Fonte: Band Paulista.

4.4 Centralização de informações e logs de auditoria

A unificação dos registros de férias, programas locais e afinação dos jornais em um único banco de dados eliminou a redundância de preenchimento. O módulo Logs permite à coordenação identificar rapidamente quem realizou determinada alteração na grade ou no timer.

5. CONCLUSÃO

O Sistema de Gerenciamento do Controle Mestre (SGCM) cumpriu com êxito o seu objetivo de modernizar, centralizar e automatizar a gestão operacional do departamento de exibição da emissora. A substituição de um ecossistema fragmentado — antes composto por planilhas manuais suscetíveis a falhas e *softwares* locais isolados — por uma plataforma web integrada provou ser uma estratégia eficaz.

Os resultados obtidos evidenciam ganhos práticos substanciais. A automação da geração de escalas rotativas reduziu um processo que consumia horas para apenas alguns minutos, eliminando sobreposições e a necessidade de retrabalho. A introdução de ferramentas acessíveis via navegador, como a "Afinação" de telejornais e o Timer de estúdio

compartilhado, melhorou a sinergia e a comunicação entre o Controle Mestre e a Switcher de Produção. Além disso, a implementação de uma sólida camada de testes automatizados e o registro completo de ações (*logs*) conferiram segurança, rastreabilidade e alta manutenibilidade ao sistema.

Apesar dos avanços significativos, o projeto apresenta limitações inerentes à sua arquitetura atual. O funcionamento contínuo do timer de estúdio em tempo real depende fortemente da estabilidade da rede interna da emissora e do desempenho dos navegadores utilizados nos terminais. Outro ponto limitante é a necessidade de acionamento manual para iniciar e pausar os cronômetros, uma vez que o SGCM ainda atua de forma independente em relação aos equipamentos primários de automação de *playout*. Adicionalmente, a geração automática de escalas não cobre cenários com afastamentos (férias ou atestado), pois estes exigem regime de 8 horas para toda a equipe, situação em que a escala deve ser elaborada manualmente.

Como propostas para trabalhos futuros e evolução contínua da ferramenta, sugerem-se: (i) a integração via API com os *softwares* de automação da emissora para o disparo automático dos timers; (ii) a implementação de notificações *push* visuais e sonoras para alertar os operadores sobre o início de uma afinação; (iii) a otimização da interface (responsividade avançada) para uso em dispositivos móveis, como *tablets*; e (iv) a adoção de tecnologias de *WebSockets* nativas (como o Laravel Reverb) para aprimorar a sincronização de rede em ambientes com maior latência.

Por fim, a experiência documentada neste artigo demonstra que a aplicação de princípios rigorosos de Engenharia de *Software*, aliados a *frameworks* web modernos e ferramentas de conteinerização, resulta em soluções corporativas robustas. O SGCM não apenas resolveu os passivos operacionais imediatos da emissora, mas também estabeleceu um padrão tecnológico sustentável que pode ser perfeitamente replicado em outras operações de *broadcast* de médio e grande porte.

REFERÊNCIAS

ALBARRAN, Alan B. **Management of Electronic and Digital Media**. 6. ed. Boston: Cengage Learning, 2016.

FOWLER, Martin. **Refatoração: aperfeiçoando o design de códigos existentes**. 2. ed. São Paulo: Novatec Editora, 2019.

KHAN, Ahmed. Secure PAAS environment over hybrid cloud using load-balanced Docker containers. **International Journal of Advanced and Applied Sciences**, v. 9, n. 3, p. 133-141, 2022. DOI: 10.21833/ijaas.2022.03.015.

KRUG, Steve. **Não me faça pensar: atualizado: uma abordagem de bom senso à usabilidade na Web**. 3. ed. Rio de Janeiro: Alta Books, 2014.

LARAVEL. **Laravel: The PHP Framework for Web Artisans**. 2026. Disponível em: https://laravel.com/docs. Acesso em: 29 abr. 2026.

LARAVEL. **Testing: Getting Started**. 2026. Disponível em: https://laravel.com/docs/testing. Acesso em: 29 abr. 2026.

LAUDON, Kenneth C.; LAUDON, Jane Price. **Sistemas de informação gerenciais**. 11. ed. São Paulo: Pearson Prentice Hall, 2014.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software: uma abordagem profissional**. 8. ed. Porto Alegre: AMGH, 2016.

SOMMERVILLE, Ian. **Engenharia de software**. 10. ed. São Paulo: Pearson Education do Brasil, 2019.

STAUFFER, Matt. **Laravel: Up & Running: A Framework for Building Modern PHP Apps**. 2. ed. Sebastopol: O'Reilly Media, 2019.

TURNBULL, James. **The Docker Book: Containerization is the new virtualization**. [S.l.]: James Turnbull, 2014.

VASCONCELOS, Augusto. **Avaliação do Sistema SGCM – Controle Mestre** [Questionário online]. Google Forms, 2026. Disponível em: https://forms.gle/HmZH8B7vHx17QiBm6. Acesso em: 15 mai. 2026.

ZETTL, Herbert. **Manual de produção de televisão**. 11. ed. São Paulo: Cengage Learning, 2014.

APÊNDICE A – Questionário de Avaliação do SGCM

O presente questionário foi elaborado pelo autor e aplicado junto aos 5 operadores do Controle Mestre da Band Paulista em abril de 2026. As respostas foram coletadas de forma anônima por meio de formulário digital. O instrumento completo pode ser acessado em: https://forms.gle/HmZH8B7vHx17QiBm6 (acesso em 15 maio 2026).

SEÇÃO 1 – PERFIL

1. Há quanto tempo você trabalha no setor?
( ) Menos de 1 ano
( ) 1 a 3 anos
( ) 3 a 5 anos
( ) Mais de 5 anos

2. Qual turno você mais atua?
( ) Manhã
( ) Tarde
( ) Noite
( ) Variável

SEÇÃO 2 – COMPARATIVO: ANTES DO SGCM (PLANILHAS) vs. DEPOIS

3. Antes do SGCM, quanto tempo você gastava por semana para lidar com escalas (elaboração, conferência, ajustes)?
( ) Menos de 30 min
( ) 30 min a 1 hora
( ) 1 a 2 horas
( ) Mais de 2 horas
( ) Não se aplica (não lidava com escalas)

4. Atualmente, com o SGCM, quanto tempo você gasta por semana para as mesmas tarefas de escala?
( ) Menos de 5 min
( ) 5 a 15 min
( ) 15 a 30 min
( ) Mais de 30 min

5. Com que frequência você percebeu erros (sobreposição de horários, conflito com férias, etc.) nas escalas geradas pelo SGCM?
( ) Nunca
( ) Raramente (1 vez por mês)

( ) Às vezes (2-3 vezes por mês)
( ) Frequentemente (1 vez por semana ou mais)

6. E antes do SGCM, com que frequência ocorriam erros nas escalas feitas manualmente?
( ) Nunca
( ) Raramente
( ) Às vezes
( ) Frequentemente

SEÇÃO 3 – AVALIAÇÃO DOS MÓDULOS ESPECÍFICOS (1 = Muito Insatisfeito; 5 = Muito Satisfeito)

7. Para cada módulo abaixo, assinale a nota correspondente:

Módulo	Nota (1 a 5)
Dashboard (visão geral dos turnos)	( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5
Afinação de Jornais	( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5
Regressiva (Timers de estúdio)	( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5
Escalas (criação/edição/envio)	( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5
Programação (grade local)	( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5
Férias (controle de períodos)	( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5
Administração (gestão de usuários)	( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

8. Logs de Ações (transparência, consulta) – Avalie se já usou.
( ) 1 (Muito Insatisfeito)
( ) 2
( ) 3
( ) 4
( ) 5 (Muito Satisfeito)
( ) Não usei ainda

SEÇÃO 4 – PERGUNTAS ABERTAS

9. Na sua opinião, qual a maior vantagem do SGCM em relação ao método anterior (planilhas)?

_____

10. Você já encontrou alguma dificuldade ou problema técnico ao usar o SGCM? Se sim, descreva brevemente.

11. Que funcionalidade nova você gostaria que o sistema tivesse?

12. De 0 a 10, qual nota você daria para o SGCM no geral?
( ) 0 ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 ( ) 8 ( ) 9 ( ) 10
(0 – Péssimo / 10 – Excelente)

13. Você recomendaria o SGCM para outro setor da emissora ou para outra emissora?
( ) Sim, com certeza
( ) Sim, com algumas ressalvas
( ) Não

14. Se quiser deixar seu e-mail para contato (opcional):

Fonte: Elaborado pelo autor.

Agradecimentos

O autor agradece à equipe do Controle Mestre da Band Paulista pela colaboração nos testes e no preenchimento do questionário, bem como à Fatec Presidente Prudente pelo suporte institucional.