UNIVERSIDADE DE PASSO FUNDO

INSTITUTO DE CIÊNCIAS EXATAS E GEOCIÊNCIAS

PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO APLICADA

UMA PLATAFORMA PARA ACOPLAMENTO DE MODELOS DE SIMULAÇÃO UTILIZANDO WEBSOCKETS

Jeancarlo Sartori

Passo Fundo

2017

UNIVERSIDADE DE PASSO FUNDO

INSTITUTO DE CIÊNCIAS EXATAS E GEOCIÊNCIAS

PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO APLICADA

UMA PLATAFORMA PARA ACOPLAMENTO DE MODELOS DE SIMULAÇÃO UTILIZANDO WEBSOCKETS

Jeancarlo Sartori

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Computação Aplicada na Universidade de Passo Fundo.

Orientador: Prof. PhD. José Mauricio Cunha Fernandes

Coorientador: Prof. Dr. Willingthon Pavan

Passo Fundo

2017

CIP – Catalogação na Publicação

S251u Sartori, Jeancarlo
Uma plataforma para acoplamento de modelos de simulação utilizando websockets / Jeancarlo Sartori. – 2017.
70 f. : il. color. ; 30 cm.

Orientador: Prof. Dr. José Mauricio Cunha Fernandes.
Coorientador: Prof. Dr. Willingthon Pavan.
Dissertação (Mestrado em Computação Aplicada) – Universidade de Passo Fundo, 2017.

1. Simulação (Computadores). 2. Linguagem de programação (Computadores). 3. Programação (Computadores). I. Fernandes, José Mauricio Cunha, orientador. II. Pavan, Willingthon, coorientador. III. Título.

CDU: 004.414.23

Catalogação: Bibliotecário Luís Diego Dias de S. da Silva – CRB 10/2241

UPF Universidade de Passo Fundo | PPGCA Programa de Pós-Graduação em Computação Aplicada
Instituto de Ciências Exatas e Geociências - ICEG

ATA DE DEFESA DO TRABALHO DE CONCLUSÃO DE CURSO DO ACADÊMICO

JEANCARLO SARTORI

Aos vinte dias do mês de março do ano de dois mil e dezessete, às 16 horas, realizou-se, no Instituto de Ciências Exatas e Geociências, prédio B5, da Universidade de Passo Fundo, a sessão pública de defesa do Trabalho de Conclusão de Curso **"Uma plataforma para Acoplamento de Modelos de Simulação utilizando Web Sockets"**, de autoria de Jeancarlo Sartori, acadêmico do Curso de Mestrado em Computação Aplicada do Programa de Pós-Graduação em Computação Aplicada – PPGCA/UPF. Segundo as informações prestadas pelo Conselho de Pós-Graduação e constantes nos arquivos da Secretaria do PPGCA, o aluno preencheu os requisitos necessários para submeter seu trabalho à avaliação. A banca examinadora foi composta pelos doutores José Maurício Cunha Fernandes, Willingthon Pavan, Carlos Amaralo Hölbig e Diego Noleto Luz Pequeno. Concluídos os trabalhos de apresentação e arguição, a banca examinadora considerou o candidato APROVADO. Foi concedido o prazo de até quarenta e cinco (45) dias, conforme Regimento do PPGCA, para o acadêmico apresentar ao Conselho de Pós-Graduação o trabalho em sua redação definitiva, a fim de que sejam feitos os encaminhamentos necessários à emissão do Diploma de Mestre em Computação Aplicada. Para constar, foi lavrada a presente ata, que vai assinada pelos membros da banca examinadora e pela Coordenação do PPGCA.

Prof. Dr. José Maurício Cunha Fernandes
Presidente da Banca Examinadora
(Orientador)

Prof. Dr. Willingthon Pavan
(Coorientador)

Prof. Dr. Carlos Amaral Hölbig
(Avaliador Interno)

Prof. Dr. Diego Noleto Luz Pequeno
(Avaliador Externo)

Prof. Dr. Rafael Rieder
Coordenador do PPGCA

À minha esposa Flavia e à minha filha Maria Eduarda, pelo amor, incentivo e inspiração.
Aos meus pais, Jerônimo e Lucí, pelo exemplo de vida, pelo amor e pelo apoio.
Amo vocês.

AGRADECIMENTOS

Agradecer a Deus primeiramente, pela vida, pela saúde e por me guiar sempre.

Ao meu orientador José Maurício Cunha Fernandes pelos ensinamentos, pelas sugestões e constante apoio na orientação deste trabalho.

Ao professor, amigo e co-orientador Willingthon Pavan pelo incentivo, pelos conhecimentos compartilhados, pelos conselhos, por estar sempre disponível.

Aos membros da banca examinadora, composta pelos professores Dr. Carlos Amaral Hölbig e Dr. Diego Noleto Luz Pequeno, pela atenção com que corrigiram este trabalho e pelas valiosas sugestões.

À minha esposa Flavia pelo amor, carinho e paciência. Por entender a minha ausência em muitos momentos e por sempre ter uma palavra de incentivo nos momentos difíceis.

À minha filha Maria Eduarda por compreender os vários finais de semana que ficamos em casa ou que não pude me fazer presente nas tuas atividades, pelo carinho e pelo respeito.

Aos meus pais, Jerônimo e Lucí, por tudo, pelos ensinamentos, pelo carinho, pela compreensão, pela força e pela tranquilidade que sempre me passaram. Vocês são meu espelho, esta conquista tem muito de vocês.

Às minha irmãs, Carlana e Tanara, pela força, companheirismo e amor. Pelos momentos de alegria e de parceria que sempre me proporcionam quando estamos juntos.

À equipe do projeto de pesquisa Mosaico da Universidade de Passo Fundo, pela amizade, auxílio e companheirismo.

Aos professores do PPGCA, pelos conhecimentos e apoio.

Muito obrigado a todos.

UMA PLATAFORMA PARA ACOPLAMENTO DE MODELOS DE SIMULAÇÃO UTILIZANDO WEBSOCKETS

RESUMO

Modelos de simulação vem sendo utilizados como forma de entender o funcionamento de um sistema complexo, vislumbrar ganhos, diminuir custos e/ou como forma de predizer eventos futuros. Na simulação computacional de um modelo complexo, como os que envolvem fenômenos ambientais e biológicos, sempre existirá uma quantidade considerável de agentes que, em algum momento, interferirão no desenvolvimento natural do experimento. É comum, portanto, que cientistas investiguem as inúmeras variações de maneira multifacetada, e à medida que o conhecimento científico avança, novos modelos são desenvolvidos e os existentes são atualizados. O resultado são modelos caros de desenvolver e difíceis de manter e melhorar. Então, porque não utilizar este conhecimento legado acoplando modelos já existentes em uma execução concomitante? O objetivo principal deste estudo é a construção de uma plataforma computacional para acoplamento, onde se possa configurar execuções de diferentes modelos em uma simulação única, independente das linguagens de programação envolvidas na construção desses modelos. Modelos próprios ou conhecidos do pesquisador também poderão ser acoplados remotamente. Obteve-se como resultado deste estudo uma ferramenta que possibilita a integração de diferentes modelos de simulação de maneira muito simples e transparente, por meio da Web. Além do conhecimento legado proporcionado pelos modelos, buscou-se o uso de base de dados de experimentos como fonte de dados para a configuração das execuções. O acoplamento se dá por meio de um servidor de *WebSockets*. Rotinas de acoplamento, adicionadas ao código dos modelos, gerenciam a comunicação com o servidor.

Palavras-Chave: acoplamento, linguagens de programação, modelos de simulação, rotinas de acoplamento, WebSockets.

A PLATFORM FOR COUPLING OF SIMULATION MODELS USING WEBSOCKETS

ABSTRACT

Simulation models have been used as a way to understand the functioning of a complex system, to envisage gains, to reduce costs and / or as a way to predict future events. In the computational simulation of a complex model, such as those involving environmental and biological phenomena, there will always be a considerable amount of agents that, at some point, will interfere in the natural development of the experiment. It is common, therefore, for scientists to investigate the numerous variations in a multifaceted way, and as scientific knowledge advances, new models are developed and existing ones are updated. Resulting in expensive models to develop and difficult to maintain and improve. So why not use this legacy knowledge by coupling already existing models into a concomitant execution? The main objective of this study is the construction of a computational platform for coupling, where it is possible to configure executions of different models in a single simulation, independent of the programming languages involved in the construction of these models. The researcher's own or known models may also be coupled remotely. The result of this study was a tool that allows the integration of different simulation models in a very simple and transparent way, through the Web. In addition to the legacy knowledge provided by the models, we sought to use the data base of experiments as data source for the configuration of the executions. The coupling takes place through a WebSockets server. Coupling routines, added to model code, manage communication with the server.

Keywords: coupling, coupling routines, programming languages, simulation models, WebSockets.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

ACE – AgMIP Crop Experiment

AGMIP – Agricultural Model Intercomparison and Improvement Project

AGRODB – Agronomy Database

DSSAT – Decision Support System for Agrotechnology Transfer

HTML5 – Hypertext Markup Language, versão 5

HTTP – Hypertext Transfer Protocol

ICASA – International Consortium for Agricultural Systems Applications

JNI – Java Native Interface

JPA – Java Persistence API

JSON – JavaScript Object Notation

MAE – Mean Absolute Error

MSE – Mean Standard Error

MVC – Modelo/Model, Visão/View e Controle/Controller

PMF – Período de Molhamento Foliar

REST – REpresentational State Transfer

RMSE – Root Mean Square Error

SGBD – Sistemas Gerenciadores de Banco de Dados

SQL – Structured Query Language

TCP/IP – Transmission Control Protocol/Internet Protocol

URI – Uniform Resource Identifier

URL – Uniform Resource Locator

XML – eXtended Markup Language

SUMÁRIO

1. INTRODUÇÃO

O rendimento máximo de plantas, determinado pelo potencial genético, raramente é alcançado porque fatores como insuficiência de água ou de nutrientes, condições climáticas adversas, danos causados por insetos e, especialmente, as doenças das plantas vão limitar o crescimento em algum momento. Portanto, a produtividade resultante do plantio de culturas para o consumo humano é de risco considerável devido à incidência de pragas, especialmente as ervas daninhas, patógenos e pragas animais. Perda de colheitas devido a estes organismos prejudiciais pode ser substancial, contudo, pode ser prevenida ou reduzida, através de medidas de proteção das culturas.

Pragas agrícolas e doenças são responsáveis por perdas de rendimento diretos que variam entre 20 e 40% da produtividade agrícola global e regularmente ameaçam a segurança alimentar mundial [1, 2]. No entanto, as perdas da colheita se mantêm pouco reconhecidas como um fator preponderante em matéria de segurança alimentar, ao passo que as doenças das plantas tiveram um enorme impacto sobre os meios de vida ao longo da história humana [1]. O trigo (*Triticum aestivum*) é um dos cereais com maior área de produção, atingindo cerca de 215 milhões de hectares em todo o mundo [3], e as doenças continuam a ser um dos principais entraves para a produção de trigo.

Devido a crescente demanda por produtos agrícolas, o aumento das pressões sobre a terra, a água e outros recursos naturais, a informação para tomada de decisão agrícola precisa estar presente em todos os níveis. Ao longo dos tempos a ciência vem tentando antever fenômenos e prevenir epidemias com objetivo de minimizar riscos e aumentar a produtividade. A geração de novos dados através de métodos de investigação agronômica tradicionais e publicações relacionadas não são suficientes para atender tais necessidades que são crescentes. Cabe destacar que experimentos agronômicos tradicionais são realizados em pontos específicos no tempo e no espaço, tornando os resultados, além de demorados e caros, locais e específicos de cada temporada.

Ferramentas importantes neste processo, os modelos de simulação computacional, auxiliam na criação de cenários, identificação de situações atípicas e geração de alertas, geralmente baseados em dados históricos. Modelos de simulação de culturas capazes de predizer o rendimento final são estudados, testados, calibrados e validados, intensivamente, por pesquisadores em várias partes do mundo [4]. Representam a simulação dinâmica do crescimento das culturas, por intermédio da integração numérica, com o auxílio de computadores. Em essência, esses modelos de simulação são programas de computadores que representam, matematicamente, o crescimento das plantas em relação ao ambiente [5]. No entanto, a maioria dos modelos de culturas opera no nível de rendimento atingível (considerando-se radiação solar, temperatura, CO_2 e precipitação), e não conta para efeitos de pragas e doenças, quer mecanicamente ou através de pós-modelo.

Os simuladores de epidemias de doenças ou surtos de praga, por outro lado, são modelos criados em função da forte influência que representam no rendimento final de uma cultura. Estas doenças ou pragas tem mais propensão de se desenvolverem em algumas regiões do que em outras. Equipes tem dedicado um longo tempo para estudo e aprimoramento de modelos que simulam uma

situação específica. O uso destes modelos vêm sendo vislumbrados como uma ferramenta de apoio à tomada de decisão com relação ao controle de epidemias. Os resultados apresentados, através das saídas geradas, buscam auxiliar na escolha de estratégias de manejo, visando manter um baixo nível de dano para uma determinada cultura e evitar maiores prejuízos econômicos.

A maioria dos modelos de simulação de epidemias, necessita, no entanto, dos dados gerados por um modelo de cultura para poder ser executado. Apesar do esforço, são poucos os modelos que conseguem rodar em tempo real, tanto o modelo de cultura, quanto um modelo de epidemia. Este processo costuma ser trabalhoso e exige o conhecimento de todos os modelos envolvidos. O uso desses modelos acoplados, mesmo por pesquisadores mais experientes, não costuma ser simples e demanda muitos passos. Estas dificuldades vão desde a condução do experimento até a análise e interpretação dos dados.

É inviável um modelo de simulação considerar sozinho, o crescimento e o desenvolvimento de plantas em paralelo com todas as doenças que podem afetar essa cultura, em qualquer tempo ou espaço. Isto se deve à complexidade envolvida e ao caráter experimental que estes modelos representam para pesquisadores locais. Da mesma forma o uso de uma única tecnologia/linguagem de programação para o desenvolvimento desses modelos não pode ser considerado. Ao longo dos anos temos visto o surgimento e o desaparecimento de diversas linguagens de programação, cada qual pode fornecer mais ou menos recursos e/ou facilidades na hora de se implementar um modelo de simulação. Dispomos de modelos construídos e mantidos há mais de 40 anos em linguagens de programação tradicionais, casos do Fortran e do C, ou modelos mais recentes construídos utilizando linguagens de programação modernas, entre elas o Java.

Questiona-se, então, o por quê de não fazer uso deste conhecimento legado acoplando diferentes modelos de simulação em uma execução concomitante. Na literatura encontram-se várias experiências de acoplamento de modelos, no entanto, o fato de várias tecnologias serem empregadas no desenvolvimento torna a integração de modelos custosa. Isto porque, é sempre necessário entender como o modelo foi desenvolvido, para que se identifique os pontos de acoplamento. Somente a partir da identificação destes locais é que será possível a comunicação com outros modelos, por meio de uma rotina adicionada ao código.

Ao mesmo tempo, uma característica muito presente no nosso dia a dia, e que as tentativas de acoplamento geralmente não abrangem, é a execução de modelos de forma remota. Ou seja, que utilizem todo o poder da internet para execução de dois, ou mais modelos, em tempo real e localmente distantes. Avaliou-se o uso de banco de dados e de *Sockets*[1] antes de definir-se pela comunicação via *WebSockets*. Desenvolvida como parte do *Hypertext Markup Language*, versão 5 (HTML5), esta tecnologia define um canal de comunicação *full-duplex* sobre o qual podem ser enviadas mensagens entre os clientes e o servidor. A arquitetura *full-duplex* permite que diferentes modelos, mesmo remotamente, desenvolvidos em diferentes linguagens de programação, se conectem e troquem mensagens entre si rapidamente.

[1] Um *Socket* é um ponto final (*endpoint*) de um canal bidirecional de comunicação entre dois programas rodando em uma rede. Ele é a "porta" na qual os processos enviam e recebem mensagens. Cada *socket* possui um endereço único na internet formado por um número IP e por um número de porta.

Diante do exposto, todo este conhecimento legado nos permitirá, de imediato, fazer uso de modelos de simulação novos e/ou atualizados, bastando identificar os pontos de acoplamento destes modelos. Abstrair a complexidade envolvida na inclusão de rotinas de acoplamento em modelos desenvolvidos por terceiros torna-se o principal desafio.

Este trabalho tem como objetivo o desenvolvimento de uma plataforma para execução de modelos de simulação de culturas, acoplados a modelos com capacidades para a contabilização de estresses bióticos. Pretende-se aliar as metodologias já consagradas, como os modelos de simulação, com os recentes avanços tecnológicos nas linguagens de programação e comunicação em rede via a internet.

Os próximos capítulos estão divididos em fundamentação teórica, descrição da plataforma e estudo de caso. Na fundamentação teórica são apresentados conceitos sobre modelos de simulação de culturas e de doenças. Abordagens de acoplamento também são expostas fazendo referências a tentativas de integração de modelos encontradas na literatura. Na descrição da plataforma são apresentadas as tecnologias e a abordagem de desenvolvimento utilizadas, bem como a própria estrutura da plataforma. Por fim, um estudo de caso, que teve como objetivo comprovar que o acoplamento de um modelo de cultura a modelos de doença, proporcionado pela plataforma em questão, produziram resultados condizentes com o que seria encontrado em uma situação real, sob as mesmas condições. Dados de campo foram obtidos para um grupo de cultivares, utilizando-se a severidade de duas doenças para comparação dos dados simulados frente aos dados observados.

2. FUNDAMENTAÇÃO TEÓRICA

2.1 MODELOS DE SIMULAÇÃO DE CULTURAS

Pensar em simulação ou em simular algo está diretamente relacionado à necessidade de entender um sistema. Conceber um modelo de simulação não é um processo simples, exige aplicação, foco nos objetivos e discussão com especialistas [6]. É essencial decidir com clareza qual o escopo do modelo, as hipóteses e o nível de detalhamento antes de qualquer implementação computacional [7]. A abstração de uma situação real para um modelo computacional visa a obtenção de resultados em menor tempo e com menor custo. Uma técnica eficiente para facilitar o entendimento de um sistema complexo é fracioná-lo em várias partes a fim de representá-lo com maior precisão [8]. Kelton et al. [9] relatam que no início a simulação tinha um custo muito elevado e só era utilizada por grandes corporações. A partir da década de 70, tornou-se possível para muitas companhias, mas mesmo com computadores mais rápidos e mais baratos só era considerada para determinar as razões de um desastre. Durante a década de 80 esta situação começa a mudar com simuladores sendo requisitados antes do início de uma produção. A partir da década de 90, os simuladores se tornaram uma ferramenta, não só para uso na fase de projeto, mas como ferramenta de auxílio na tomada de decisões [9]. A partir de então, muitos esforços foram realizados e muitos resultados de sucesso obtidos.

Na agricultura, os modelos de simulação vem sendo utilizados para produzirem informações que experiências de campo levariam muito tempo para fornecer. Os modelos de sistemas agrícolas são cada vez mais utilizados para vários tipos de propósitos, além de comporem um sistema maior, no qual cada um é responsável por determinadas tarefas. Pavan [10] cita algumas tarefas que podem ser desempenhadas por estes modelos, como: desenvolvimento de culturas (planta); aparecimento/ataque de pragas e doenças; determinação de rendimentos; dinâmica da água no solo, desenvolvimento de sistemas radiculares e absorção de nutrientes, entre outros.

Estes modelos agrícolas têm sido muito importantes, também, na questão da sustentabilidade. Motivados pela demanda crescente de comida e pela diminuição das áreas de plantio em todo o mundo, estes modelos, buscam identificar as mais apropriadas e efetivas práticas de manejo. De acordo com Jones e Porter [11], desde que informações de solo, manejo, clima e sócio-econômicos estejam disponíveis, os modelos terão condições de auxiliar no gerenciamento de áreas de plantio, bem como fornecer subsídios que motivem decisões políticas na questão da sustentabilidade. Estes modelos estão disponíveis para a comunidade em geral e preveem o crescimento e a produção de culturas nos seguintes níveis: potencial, atingível ou produção real. O potencial de produção é previsto pelos modelos em função do clima local (radiação solar, temperatura, CO_2 e comprimento do dia), mas assumindo que não há limitações de déficit hídrico, a falta de fertilidade, ou pragas e/ou doenças. A produção atingível é a que prevê onde as limitações de água e fertilidade são simuladas. Enquanto que a Produção real é o que os agricultores obtiveram, inclui todos e quaisquer fatores limitantes no terreno, incluindo os aspectos de fertilidade do solo e aspectos de pragas e/ou doenças. No entanto,

a maioria dos modelos de culturas opera no nível de rendimento atingível (considerando-se radiação solar, temperatura, CO_2 e precipitação), e não leva em consideração efeitos de pragas e doenças, quer mecanicamente ou através de pós-modelo.

Apesar da considerável sobreposição de focos (mesmo propósito) que encontramos em modelos agrícolas, não se justifica diminuir a importância de nenhuma das alternativas. Cada um deles é importante em virtude das diferenças entre as culturas e a região onde são cultivadas.

Cabe salientar, relativo aos modelos de simulação, que passado o processo de implementação o mesmo precisa ser validado. Neste processo, os resultados obtidos precisam estar dentro de um intervalo aceitável. No caso dos modelos agrícolas estes são calibrados, comparando-se os dados simulados com os dados experimentais observados. O processo de análise dos resultados é a terceira grande etapa do ciclo de construção de um modelo de simulação, a primeira etapa é a de entendimento e de concepção do modelo, enquanto que a segunda etapa é a de implementação [7]. Caso o resultado não seja satisfatório o modelo pode ser modificado ou até mesmo descartado, iniciando-se o ciclo de vida novamente. Portanto, as três etapas desta metodologia não podem ser vistas como uma sequência linear, mas sim como forma espiral [7] (apud Robinson 2004).

A construção de um modelo de simulação para a agricultura não costuma ser um processo simples, envolve altos custos, exige uma equipe multidisciplinar, e muito tempo de trabalho dedicado. Nesta perspectiva, encontramos alguns modelos de simulação de cultura sendo desenvolvidos e atualizados há mais de 40 anos [6]. E, por datarem da década de 60, é compreensível encontrarmos modelos criados em linguagens de programação tradicionais como o Fortran. O uso de diferentes linguagens de programação, aliás, não é desencorajado, pelo contrário, toda linguagem de programação possui pontos fortes. O Fortran, por exemplo, se destacou na área de análise numérica por empregar velocidade e precisão ao trabalhar com cálculos numéricos.

Dentre os modelos desenvolvidos com as características citadas temos o *Decision Support System for Agrotechnology Transfer* (DSSAT), que é formado por uma plataforma de vários modelos. Entre os sub-módulos encontram-se modelos que simulam o crescimento e desenvolvimento de diversas culturas: de cereais (trigo, milho, arroz, sorgo, etc), de leguminosas (soja, amendoim, etc), de raízes (batata e mandioca), entre outras (tomate, cana de açúcar, girassol, etc) [12, 13]. Mantido pela Fundação DSSAT, é o resultado dos esforços de um grupo de pesquisadores de várias universidades e centros de pesquisa do mundo. Amplamente utilizado no mundo [12] este modelo opera com cálculos diários, integra arquivos contendo os efeitos de solo, as características da espécie, os coeficientes genéticos, os dados climáticos e as opções de manejo, na parametrização de entradas. Esta flexibilidade, ao trabalhar com arquivos de entrada, permite a adaptação do modelo para uso nas mais diferentes condições e localidades. E é com base nestes parâmetros que o desenvolvimento da cultura é afetado caso as condições não sejam as ideais. As saídas, ou seja, os resultados da simulação, podem alimentar outros modelos, como, por exemplo, os que são responsáveis pela simulação de epidemias em plantas [14, 6].

O *CropSim-Wheat*, um dos modelo do DSSAT, simula o crescimento e desenvolvimento do trigo. Opera com cálculo diário e inclui sub-rotinas para simular o crescimento, a fenologia, o balanço

hídrico e a disponibilidade de nitrogênio no solo. Outro fator importante a destacar no modelo *Cropsim-Wheat* é a simulação da área foliar e do número de folhas existentes a cada dia do ciclo da cultura do trigo [15]. Possui uma sub-rotina denominada *Disease* [16], criada para calcular o progresso da doença, que pode ser utilizada como ponto de acoplamento para penalizar a área foliar.

A incapacidade de contabilizar os efeitos de pragas em modelos de culturas motivou a criação do módulo PEST por Batchelor et al [17]. Adicionado inicialmente ao modelo CROPGRO, pertencente a plataforma do DSSAT, este módulo visa considerar situações em que a cultura pode ser acometida pela ação de pragas e/ou doenças. A incidência dessas representa danos que são aplicados a variáveis do modelo por meio de pontos de acoplamento [17].

A expressão de um dano depende da unidade de medida, a qual representa a variável a ser afetada. Geralmente, é calculada em unidades de massa por unidade de área, exemplo m^2 x m^{-2}. Outra maneira de dano pode ocorrer pelo desfolhamento, na qual uma diferença percentual na massa foliar é subtraída do tratamento. As taxas de danos são sempre calculadas por dia.

Batchelor et al [17] consideraram o tipo de dano causado por insetos. Como esses competem por locais de alimentação, se a demanda for alta os danos podem ser distribuídos para outras áreas da planta. Ou seja, qualquer dano calculado comprometerá o crescimento e desenvolvimento da cultura e a área subtraída não estará mais disponível para a sequência da simulação no próximo cálculo diário.

Vale destacar que a execução do módulo de PEST é parametrizada a partir do arquivo de configuração da execução. Caso opte-se por não utilizá-lo, esta rotina não será chamada na execução do modelo de simulação do crescimento e desenvolvimento. Outros dois arquivos definem, respectivamente, cada praga/dano e o nível medido de cada praga/dano em um determinado dia do ano [17]. Definições de praga/dano incluem a identificação, o método que a caracteriza, a variável de acoplamento, o coeficiente de taxa de alimentação e as unidades de medida.

A partir do módulo PEST e respectivos arquivos de configuração, é possível realizar várias combinações de pragas e danos sobre um modelo de cultura, mesmo que de forma genérica, diferentemente de outros modelos de simulação de culturas que não possuem módulo similar ao PEST, do DSSAT. Em razão disso, vários modelos de doenças e pragas vêm sendo concebidos e atualizados ao longo dos anos. O objetivo é entender melhor o comportamento de uma determinada praga/dano e buscar alternativas para minimizar efeitos. Esses modelos, geralmente, necessitam de um modelo de culturas para simular o crescimento e desenvolvimento de uma cultura específica.

2.2 MODELOS DE SIMULAÇÃO DE DOENÇAS

Na área agrícola, além do preço de venda, que é definido pelo momento do mercado mundial, o que definirá a lucratividade de uma colheita são a produtividade e o custo de produção. A produtividade está baseada em inúmeros fatores como as próprias necessidades bióticas exigidas pela cultura em termos de nutrientes, água no solo, entre outros; as condições climáticas e a incidência de pragas e/ou doenças. Quanto ao custo de produção este é um ponto determinante e altamente variável, visto que aumentar ou diminuir este custo pode não alterar necessariamente a produtividade [18]. No custo

de produção estão incluídas entre outras despesas os gastos com defensivos, bem como os custos de aplicação. Oliveira et al [18] em uma análise experimental comprovaram que aplicações tardias de fungicidas propiciam o aumento no nível de severidade da doença e como consequência reduções significativas de produtividade.

Entender melhor os fatores que influenciam a dinâmica de uma doença pode auxiliar no manejo da cultura visando minimizar os riscos de epidemias [19]. Modelos de simulação e de previsão de risco de doenças ou pragas auxiliam nos processos de entendimento e de tomada de decisão [6]. Auxiliam na escolha de medidas eficazes no controle e tratamento de epidemias, levando em consideração os dados de localização do cultivo e indicando a melhor época para uma ação de proteção a cultura. Identificar antecipadamente a possibilidade de ocorrência de uma determinada doença, pode levar a uma melhor relação custo e benefício [6] e, diminui o impacto que as aplicações desnecessárias de fungicidas causariam ao meio ambiente [18, 19].

Modelos de simulação de doenças são concebidos, geralmente, para o estudo de uma doença específica. Isto se deve ao fato de as epidemias estarem diretamente relacionadas aos fatores climáticos extremos em um determinado estádio de desenvolvimento da cultura. A propensão de desenvolvimento de uma doença, portanto, é maior em algumas regiões do que em outras.

Segundo Fernandes et al [6] os primeiros modelos de simulação de epidemias serviram para avaliar aspectos básicos da epidemiologia, passando a considerar aspectos de manejo de doenças e à tomada de decisão somente mais tarde. Os primeiros relatos são dos modelos EPIDEM, para a Alternariose em tomate e o EPIMAY, para a queima da folha do milho, já no final da década de 60 [6, 10, 20]. A partir daí, inúmeros modelos têm sido desenvolvidos, dos mais simples aos mais complexos, visando entender o funcionamento destas epidemias e a implementação de sistemas de simulação capazes de auxiliar na tomada de decisões [10].

Pode-se citar modelos desenvolvidos na cultura da soja, em que se destacam os esforços para a ferrugem asiática, causada pelo fungo *Phakopsora pachyrhizi*, ao passo que na cultura do trigo destacam-se os modelos para a ferrugem da folha, causada pelo fungo *Puccinia triticina Erikss* [21]. Somente para o modelo da ferrugem, Delponte et al [21] relacionam doze modelos, dentre os quais estão o SOYRUST e o CLIMEX, como modelos epidemiológicos, e o HYSPLIT e o SRAPS, como modelos aéreo-biológicos. “Os modelos aéreo-biológicos mostram-se importantes quando se necessita controlar a entrada de uma nova praga numa determinada área” [10].

Os esforços de Pavan e Fernandes [10], renderam o Modelo Genérico de Doenças. Modelo que nasceu com o objetivo de representar o maior número possível de epidemias. Arquivos no formato *eXtended Markup Language* (XML), permitem a parametrização destas doenças a fim de representar o mais realisticamente possível as observações feitas em campo [21]. Pode ser utilizado para a avaliação de danos provocados pelas doenças, utilizando para isso dados de clima, solos e cultivar, o que pode predizer danos à cultura por meio de dados históricos e/ou de estações meteorológicas. O modelo tomou como base o clico de vida do patógeno, descrito amplamente por Pavan [10], de maneira a obter os resultados esperados.

Este modelo genérico foi desenvolvido de maneira modular, na linguagem de programação orientada a objetos Java. Esta linguagem foi escolhida por facilitar a interpretação e simulação do ciclo de vida do patógeno, já que cada componente do modelo é entendido como um objeto ou um grupo de objetos no sistema [10].

A grande capacidade de integração que caracteriza os modelos de simulação proporciona à ciência agrícola grandes resultados por meio da simulação dinâmica de processos biológicos [6]. Representa uma prática comum os modelos de simulação de epidemias se valerem das saídas geradas pelos modelos de cultura. A partir do momento que os modelos de cultura geram saídas padronizadas, em arquivos textuais, os modelos de epidemias passam a se valer desta fonte, como forma de entrada para execução da própria simulação. Comumente, os resultados obtidos neste formato de acoplamento, ao serem analisados, ficam dentro dos limites aceitáveis e o modelo passa a ser utilizado como um importante recurso para análise e monitoramento de uma epidemia específica.

2.3 EXPERIÊNCIAS DE ACOPLAMENTO

O uso de acoplamento vem sendo utilizado corriqueiramente como forma de aproveitar o conhecimento legado. O grande número de modelos computacionais simulando sistemas complexos e multifacetados propicia a integração dos mesmos como forma de alavancar a programação complementar. Ou seja, direcionando o tempo para o desenvolvimento de novas necessidades. Em novos modelos este processo tende a uma facilidade maior, visto que costumam seguir o caminho da modularização, onde uma integração de módulos não comprometará a estrutura global do modelo [20].

Até recentemente, não existiam padrões de desenvolvimento para a integração de modelos de simulação, pois a cada estudo criava-se uma nova estrutura de acoplamento [20]. Qualquer tentativa de integração exigia um amplo conhecimento das estruturas dos modelos a acoplar. Com o tempo, algumas abordagens envolvendo integração de sistemas foram sendo descritas. Ainda que hoje, os avanços nas tecnologias computacionais contribuam para o aumento da viabilidade de sistemas acoplados [22], esse processo costuma ter alto grau de complexidade devido ao ambiente heterogêneo dos sistemas envolvidos. Buscar a abordagem mais apropriada para os modelos em estudo torna-se essencial.

Na sequência serão descritas algumas dessas abordagens mais utilizadas, segundo Bulatewicz [23]. Casos de sucesso envolvendo modelos de simulação, encontrados na literatura, serão apresentados para enriquecimento dos conceitos sobre abordagens de acoplamento. A discussão sobre estes trabalhos também é o resultado de um levantamento sobre o que já foi tentado com relação a integração de modelos de simulação. Esta análise é importante, visto que, apesar de terem atingido o objetivo, estes estudos trouxeram algumas questões frente ao formato e tecnologias escolhidas para o desenvolvimento. Pretende-se discutir a abordagem de acoplamento a ser escolhida como forma de minimizar os riscos e evitar a mudança de rumo durante o desenvolvimento. É importante salientar que não existe uma receita pronta, mas alternativas melhores para determinadas situações.

2.3.1 Abordagem Monolítica

A criação de um novo modelo customizado a partir fragmentos de código fonte de dois ou mais modelos, caracteriza a abordagem monolítica, sistematizada na Figura 1.

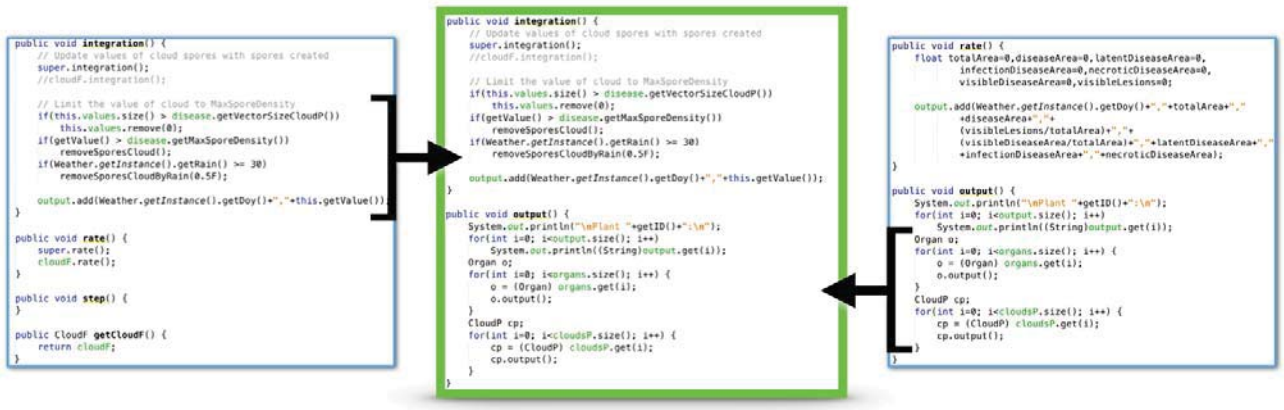

Figura 1. Códigos de dois ou mais modelos são utilizados para criação de novos modelos (modificado de [23]).

Deve-se destacar a importância da discussão de alguns aspectos antes da escolha de qualquer abordagem. Estes aspectos são: identificação de modelos apropriados, especificação das interações (conversões de dados, tipos de variáveis, entre outras), e a verificação da possibilidade de integração dos códigos fontes [20]. A integração de códigos fonte, por exemplo, é muito importante na abordagem monolítica em virtude de se trabalhar apenas com uma linguagem de programação. Estando todos os aspectos entendidos e sem restrições a detalhes do código fonte, obtém-se um certo grau de facilidade na integração dos códigos fonte dos modelos.

A prática deste tipo de abordagem foi adotada por Koo [24], onde o autor realizou o acoplamento de dois modelos de simulação desenvolvidos em uma mesma linguagem. Os modelos utilizados foram o CROPGRO-Tomato, o qual simula o crescimento e desenvolvimento do tomate em conjunto com o modelo de doenças do tomate FOBIS-TMED. O acoplamento deste modelo foi realizado com a inclusão dos códigos fontes do modelo FOBIS-TMED dentro do modelo CROPGRO-TOMATO, que assim passa a contabilizar danos causados por duas doenças no crescimento do tomate.

A aplicação deste tipo de abordagem, portanto, exige um completo entendimento dos modelos envolvidos. A dificuldade aumenta em função da maioria destes modelos possuírem pouca ou nenhuma documentação, além de terem o código fonte pobremente comentado. Dificuldades na aplicação de técnicas de engenharia de software, envolvendo testes, verificações, atualizações, entre outras, também é um ponto a se considerar na escolha desta abordagem de acoplamento [20].

2.3.2 Abordagem Programada

Um caso de sucesso no uso de acoplamento de modelos através de *Sockets* é descrito por Pavan [10]. No estudo utilizou-se do modelo de simulação *CROPSIM-Wheat*, como modelo de si-

mulação de cultura, em conjunto com o modelo de genérico de doença RUSTSIM[2]. A integração dos modelos se dá por meio da utilização das saídas do modelo *CROPSIM-Wheat* como entrada de dados para o modelo de simulação RUSTSIM (Figura 2).

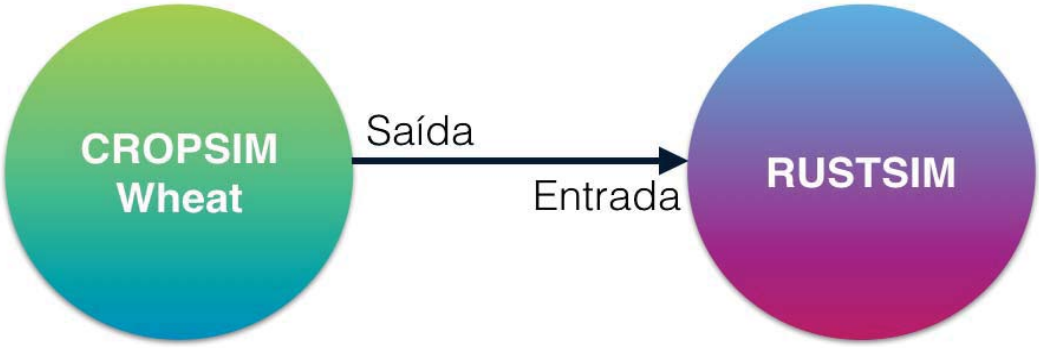

Figura 2. Integração dos modelos Cropsim e RUSTSIM

Esta abordagem em que a saída de dados de um modelo integrado é utilizada como entrada de dados para outro modelo, é conhecida como abordagem de acoplamento programada, na qual os modelos permanecem como programas independentes, não afetando um ao outro em tempo de execução. A aplicação desta abordagem automatiza o processo de seleção dos modelos e dos conjuntos de dados pelo cientista. Além disso, pode-se especificar a distribuição dos conjuntos de dados e a ordem de execução dos modelos na integração.

Entretanto, optar por esta abordagem tende a exigir um desenvolvimento adicional, para que a comunicação entre os modelos ocorra. Por exemplo, diferentes interfaces entre modelos necessitam do desenvolvimento de padrões (para que seja possível a troca de dados através das interfaces), tais como a transformação de dados (mesmas unidades de medidas), que exige a substituição de todas as entradas e saídas dos modelos que não sigam os mesmos padrões.

Por fim, entende-se que onde um modelo (cultura) precise finalizar a execução para que outro modelo (doença) inicie, pode não representar a produção real ao final da simulação. Visto que, a taxa de crescimento não estaria sendo afetada pela evolução da doença calculada no segundo modelo (doenças e/ou pragas).

2.3.3 Abordagem Orientada a Componentes

O resultado final da integração, usando esta abordagem, constitui-se num modelo único como o resultado da abordagem monolítica. A diferença é que na abordagem Orientada a Componentes, os códigos dos modelos são decompostos em componentes de *software* [23], de forma que cada componente possa ser reutilizado em uma variedade de situações sem a necessidade de alterações. Estas sub-rotinas, modulares e reutilizáveis, devem ser escritas de maneira a simplificar o processo de alteração de um componente isoladamente, ou mesmo a substituição deste. Na Figura 3 pode-se

[2]RUSTSIM: modelo genérico de doenças parametrizado para a Ferrugem da Folha do Trigo.

observar esta divisão de componentes de maneira clara e objetiva; conexões entre estes componentes resultarão no modelo acoplado.

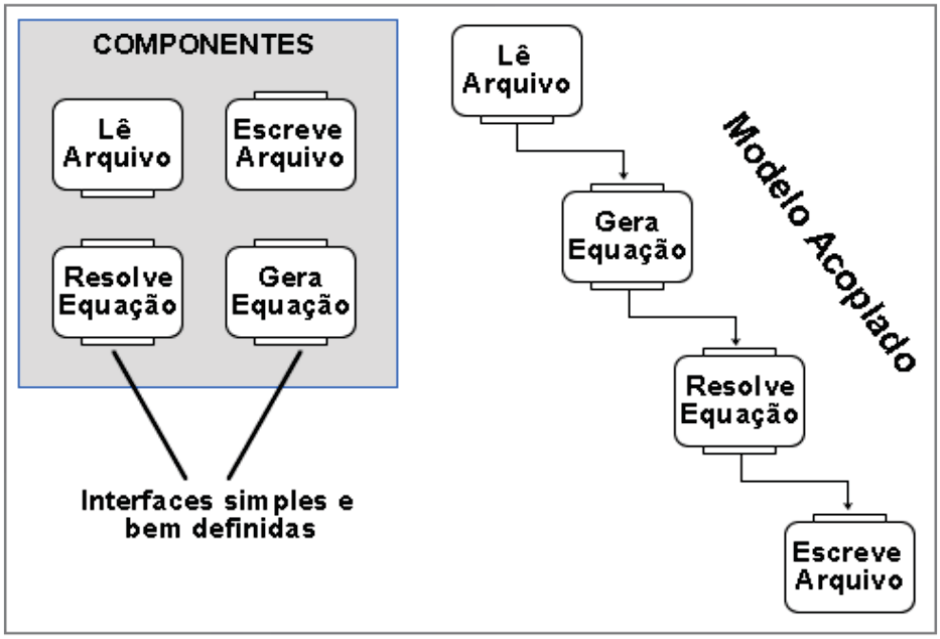

Figura 3. Componentes são conectados para criação de modelos (modificado de [23])

O processamento de cada componente está na utilização das entradas para geração das próprias saídas. Cabe, portanto, a quem for utilizá-los conhecer os tipos de dados exigidos na entrada, e os tipos de dados que serão recebidos no retorno do componente [20].

Rech et al [25] descrevem a criação de um modelo genérico de simulação de culturas, baseado no modelo proposto por Porter et al [26]. O modelo foi desenvolvido na linguagem R, sendo dividido em três partes principais: módulos, modelos de plantas e modelos de doenças. Cada parte possui diversos submódulos como apresentado na Figura 4.

Como o resultado é um modelo único, uma das desvantagens desta abordagem implica na necessidade de conhecimento geral e detalhado dos códigos, e da ordem de execução dos modelos envolvidos. Enquanto que, como vantagens podemos citar a possibilidade de aplicação de técnicas de engenharia de *software* (testes, verificações, atualizações, entre outras), e a facilidade de utilização dos componentes gerados em novas construções e composições futuras.

A abordagem orientada a componentes pode ser o caminho ideal para a construção de novos modelos, se componentes estiverem disponíveis para tal construção. Já que devido a grande reprogramação dos códigos existentes esta conversão, muitas vezes, torna-se inviável [20].

2.3.4 Abordagem de Comunicação

Na abordagem de comunicação os modelos interagem entre si por meio da troca de mensagens, durante todo o processo de execução [24]. Trata-se da abordagem mais utilizada, porém mais complexa, já que busca evitar a reescrita de códigos fontes, ao contrário da abordagem monolítica e

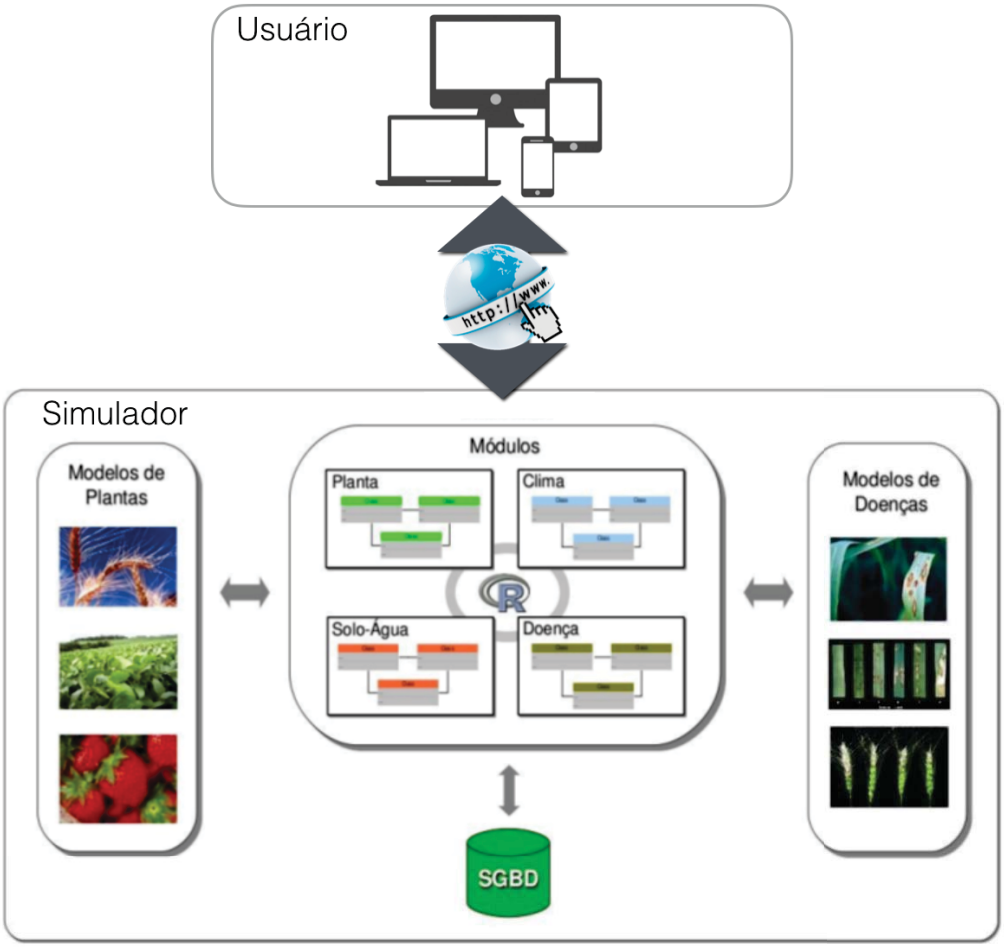

Figura 4. Abordagem orientada a componentes utilizada por Rech et al [25]

de componentes. A ideia de manter os modelos independentes torna mais fácil a experimentação com diversos modelos e reduz esforços.

Modelos desenvolvidos nesta abordagem podem fazer uso de um *middleware*, aplicações independentes que intermediam a comunicação na execução dos modelos, ou não [20]. Em qualquer das classificações escolhidas haverá a necessidade dos dados trocados entre os modelos seguirem um padrão.

Por mais que se busque a integração de modelos com mínimas mudanças nos códigos fontes, ainda existe a necessidade de conhecer bem os modelos envolvidos. O processo de acoplamento exige muita atenção, visto que, inconsistências na comunicação podem interferir nos resultados finais. O desempenho é outra questão que necessita de atenção, já que a execução de modelos acoplados pode acarretar uma drástica desaceleração no tempo de execução do modelo [20]. Como as interfaces de troca de dados são requisitadas frequentemente necessitam ser pensadas para minimizar o tráfego e a negociação destes dados.

Esta abordagem não requer que os modelos sejam desenvolvidos na mesma linguagem de programação ou arquitetura de *software*. Propicia a ampla utilização do conhecimento legado proporcionado por modelos de simulação tradicionais e amplamente difundidos. Além do mais, pode-se

fazer uso de tecnologias atuais na implementação de um novo acoplamento, disseminando o uso e aumentando a vida útil destes modelos.

A seguir apresenta-se algumas tentativas bem sucedidas de acoplamento, que seguem a abordagem de comunicação, utilizando diferentes tecnologias e modelos de simulação nos acoplamentos.

2.3.4.1 Acoplamento via RServe

A abordagem de comunicação foi utilizada na validação do Modelo Genérico de Doenças, proposto por Pavan [10]. No estudo Pavan propõe a integração de um modelo de simulação do crescimento e desenvolvimento do trigo ao Modelo Genérico de Doenças, parametrizado para simular a ferrugem da folha do trigo. O modelo de simulação do crescimento e desenvolvimento escolhido foi o *CropSim-Wheat*.

O principal desafio encontrado neste acoplamento diz respeito as diferentes linguagens de programação empregadas no desenvolvimento destes modelos. O *CropSim-Wheat* desenvolvido em Fortran, uma linguagem de programação estruturada, e o Modelo Genérico de Doenças desenvolvido em Java, uma linguagem de programação orientada a objetos.

A conexão entre as duas linguagens (Java e Fortran), ficou a cargo da linguagem R, "a qual permite a integração de diferentes linguagens de programação, além de oferecer ótimos recursos computacionais para operações estatísticas e geração de gráficos" [10]. No módulo de integração entre o *CropSim-Wheat* e o Modelo Genérico de Doenças, um servidor *Transmission Control Protocol/Internet Protocol* (TCP/IP) RServe é o responsável pelas conexões remotas, autenticação e transferência de arquivos (Figura 5). Para que o Modelo Genérico de Doenças rodasse com cálculos diários sem ter que aguardar o final da simulação, o modelo de simulação do crescimento e desenvolvimento foi modificado e recompilado novamente para se integrar ao novo modelo. Assim, ao final de cada cálculo diário há uma comunicação entre os modelos, em que o modelo que está sendo executado passa a vez para o outro modelo. Os dados recebidos do *CropSim-Wheat*, como quantidade de órgãos, área total e área senescente de cada órgão são utilizados pelo Modelo Genérico de Doenças, para o cálculo de novas lesões, produção de esporos e expansão da lesão [10].

Figura 5. Conexão entre o modelo genérico de doenças e o CropSim-Wheat

Toda a integração entre os modelos se dá pela troca de dados entre os dois modelos. O modelo de cultura, enviando informações sobre o número de folhas criadas e a área disponível (sadia) de cada uma, naquele instante [10]. Enquanto que o modelo que simula o progresso da doença retroalimenta o modelo de cultura com informações sobre a área infectada.

2.3.4.2 Acoplamento via Java Native Interface (JNI)

Outro exemplo deste tipo de abordagem foi desenvolvida por Pedrini [20], onde um modelo genérico de doenças [27] é integrado com o modelo *CROPGRO-Soybean*, que simula o crescimento e desenvolvimento da soja, possuindo como objetivo, contabilizar o impacto de dano que a doença ferrugem asiática causa nesta cultivar. A interface desenvolvida por Pedrini [20], consiste na utilização de *Java Native Interface* (JNI), para comunicação entre a linguagem Java, do modelo genérico de doenças, com a linguagem Fortran, do modelo *CROPGRO-Soybean*.

JNI é uma especificação do Java, que permite que um código Java chame ou seja chamado por aplicações nativas de um sistema, fornecendo uma interface para a programação híbrida [20]. Esta execução de código nativo dentro do programa Java, permite a interação e troca de dados entre os modelos desenvolvidos em diferentes linguagens de programação de forma dinâmica.

O diagrama que exemplifica todo o fluxo de dados no acoplamento dos dois modelos e as linguagens de programação envolvidas está representado na Figura 6. O fluxo de execução tem dez passos, com alguns sendo executados uma série de vezes, enquanto outros são executados uma única vez durante a simulação [20].

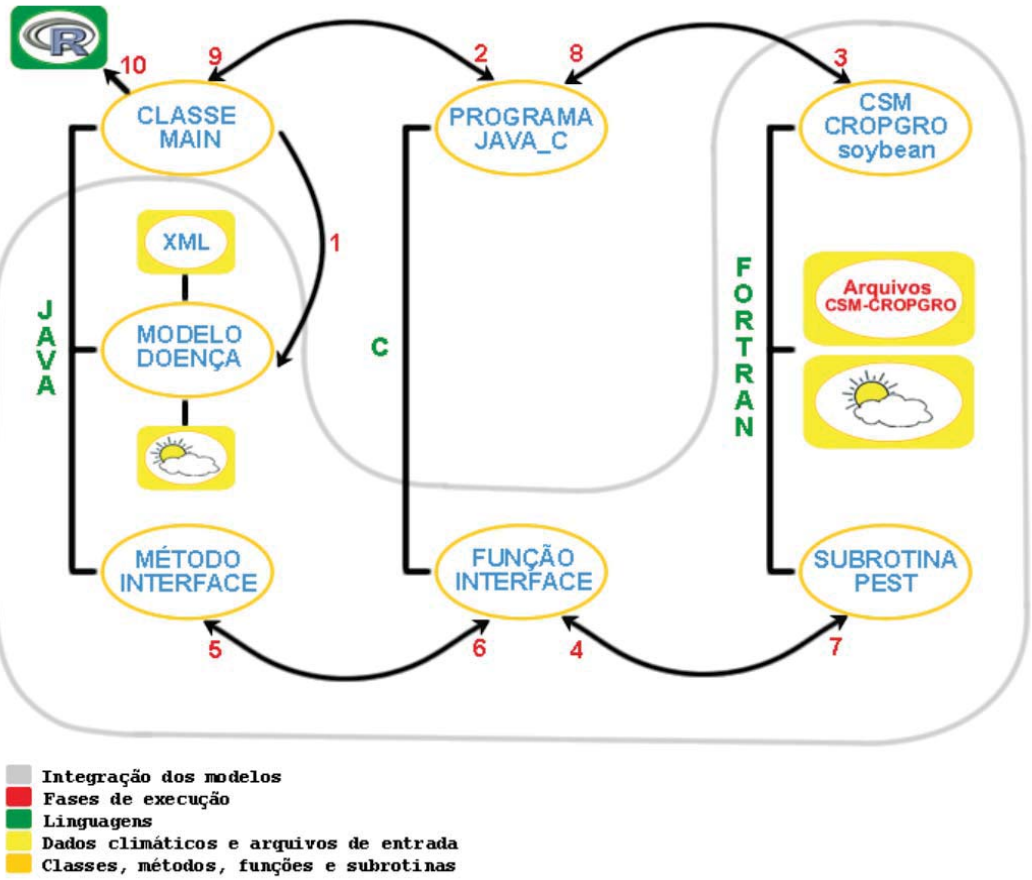

Figura 6. Abordagem de comunicação utilizando JNI, por Pedrini [20]

Esta estrutura de acoplamento depende de um amplo conhecimento dos dois modelos acoplados, mas funciona muito bem rodando os dois modelos localmente. Qualquer alteração nesta comunicação ou na troca de um dos modelos exigirá conhecimentos em programação, muito tempo de estudo e implementação.

2.3.4.3 Acoplamento via Banco de Dados

O acoplamento de modelos também pode se dar por meio do banco de dados. Solano et al [22] utilizam este expediente para que os modelos se comunicam uns com os outros indiretamente. Ou seja, os modelos ao serem executados, separadamente, armazenam informações em um banco de dados, tornando possível a recuperação posterior por outros modelos de simulação.

Todo o processo é sincronizado pelo relógio do banco de dados, que também monitora os agendamentos. Uma das poucas exigências dessa abordagem é que a linguagem de programação, em que foi construído o modelo, forneça meios de conexão, leitura e escrita no banco de dados escolhido para o acoplamento.

O referido estudo trabalha com rotinas de inserção e de extração de dados (Figura 7). Para enviar um dado TI1 do Modelo M1 para o Modelo M2, o processo passará por uma rotina de inserção, denominada I1, que inserirá os dados no banco. Do banco de dados uma rotina de extração é chamada gerando os dados convertidos TE2, que ficarão disponíveis para uso pelo Modelo M2. Após o tratamento do dado recebido o Modelo M2 precisa enviar um novo dado (TI2) para o Modelo M1 e o processo se inicia novamente com a inserção dos dados no banco e posterior extração destes dados pelo Modelo M1.

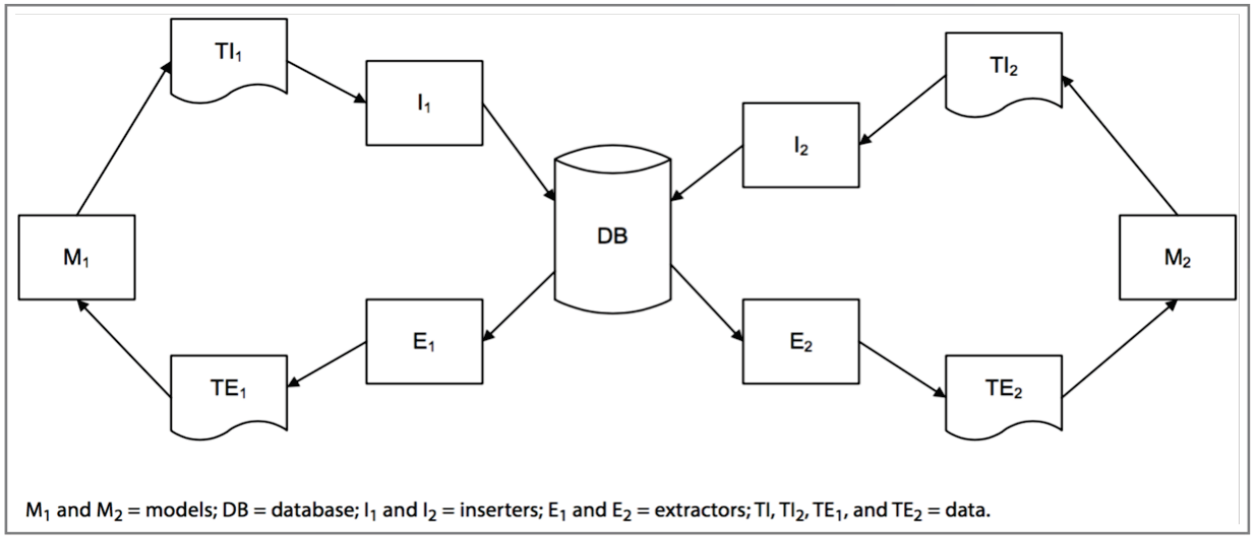

Figura 7. Troca de dados, entre modelos, por meio de um sistema de gerenciamento de banco de dados relacional [22]

Entre outras exigências de desenvolvimento, definiu-se não haver restrições ao número de modelos que podem se comunicar com o banco de dados. Outra preocupação era que os modelos

fossem fracamente acoplados, ou seja, um modelo sabe pouco, ou quase nada, sobre o outro modelo. O autor relata que essas exigências foram cumpridas, mesmo o acoplamento tendo sido testado, inicialmente, somente com dois modelos trocando informações.

Acoplar sistemas em qualquer abordagem diferente da monolítica [23] exigirá alterações nos módulos para que se conectem e troquem informações. Buscou-se, no trabalho de Solano et al [22], minimizar essas mudanças, inserindo somente o código estritamente necessário em locais adequados para a exportação dos dados para sistema de importação.

O autor deixa bem claro que problemas de desempenho podem ocorrer, mas que além da capacidade de máquina, neste formato, o único jeito de acelerar o processo seria fazendo ajustes de desempenho diretamente no banco de dados. Este é um dos principais problemas, visto que, muitos modelos rodam a simulação rapidamente quando executados sozinhos. Se com o acoplamento este tempo aumentar, consideravelmente, pode haver uma insatisfação por parte de quem está fazendo uso do acoplamento.

As características citadas até aqui, no que diz respeito ao acoplamento via banco de dados, são comuns também no trabalho de Toebe [28] que propõem o uso de Sistemas Gerenciadores de Banco de Dados (SGBD) como intermediários no acoplamento. O MBAPraga, um modelo de simulação de insetos e pragas agrícolas desenvolvido em Java, foi escolhido para acoplar a um modelo de cultura escrito em Fortran. Ambos os modelos são executados de maneira paralela.

Devido ao Fortran não permitir o acesso a banco de dados, foram adicionadas sub-rotinas no código Fortran direcionando para rotinas codificadas em C. O código C implementa a conexão, consulta e atualização dos dados no próprio banco. Laços de repetição foram utilizados para monitorar o recebimento de dados em ambos os modelos.

É fato que o uso do SGBD para o acoplamento de modelos sobrecarrega o processamento, pois, será o responsável por gerenciar os acessos simultâneos a um mesmo registro. Essa função de bloqueio e liberação de um registro, ora para um modelo, ora para outro, acarretará numa maior latência na comunicação entre os modelos. Outro fator determinante é que o acoplamento tende a ser local, visto que a execução por meio de redes diferentes pode inviabilizar a execução, seja pela qualidade da conexão, pela estrutura de redes, por questões de segurança, entre outros.

Cada uma das experiências apresentadas possui um certo grau de dificuldade na implementação, na execução ou na evolução. De maneira geral exigem um conhecimento detalhado dos modelos envolvidos. O presente estudo se propõe a utilizar ferramentas atuais que abstraiam a complexidade envolvida no acoplamento de modelos de simulação e permitam a execução remota de dois ou mais modelos, trocando informações com baixa latência e, independente das linguagens de programação envolvidas na construção dos modelos. É condição indispensável também que se permita aos pesquisadores acoplarem modelos próprios com facilidade, seguindo uma especificação simples.

3. DESENVOLVIMENTO DA PLATAFORMA DE ACOPLAMENTO

As tentativas de integração de modelos encontradas na literatura apesar de lograrem êxito, geralmente, focam no acoplamento de modelos específicos, em uma estrutura não distribuída e difícil de manter e melhorar. Usuários sem conhecimentos em programação desejando ajustar um modelo acoplado a outro, usando dessas abordagens, teriam imensas dificuldades em obter sucesso.

Para atender a possibilidade de acoplamento, a reescrita destes modelos em uma linguagem mais moderna e única é inviável. Implementar o que estes modelos consolidados e respeitados no meio já proporcionam demandaria muito tempo e alto custo. A ideia de simplesmente integrar novas funcionalidades a modelos existentes, na mesma linguagem de programação, também não parece muito inteligente. Por exemplo, o uso de um modelo desenvolvido em Fortran, em que se assume esta linguagem de programação como único meio para o desenvolvimento, restringe o acesso há novos recursos de distribuição, de compartilhamento e de apresentação de resultados.

Buscou-se neste estudo a construção de uma plataforma distribuída que permitisse o acoplamento de dois ou mais modelos de simulação agrícola. Os modelos podem ser executados, dentro desta estrutura, estando em locais distintos geograficamente. A utilização de tecnologias atuais de comunicação em rede viabilizou o uso dos esforços, do tempo e dos resultados proporcionados por vários pesquisadores multidisciplinares na construção desses modelos, propiciando assim um aumento na vida útil do código legado [20].

A plataforma, nomeada *Coupling*, utiliza a arquitetura Web e permite a seleção de modelos de simulação de cultura e de doenças pré-configurados no ambiente. Modelos próprios ou conhecidos do pesquisador também poderão ser acoplados remotamente, por meio da inserção de rotinas de acoplamento no código próprio. Essas rotinas são fornecidas pela plataforma, via e-mail, a partir da configuração da execução.

3.1 ARQUITETURA DA PLATAFORMA

Para avaliar os potenciais impactos de pragas e doenças na produção de culturas, alguns modelos de culturas incluem pontos de acoplamento; variáveis especiais do modelo que podem ser utilizadas para representar danos de pragas e doenças aos órgãos da cultura, ou processos de crescimento. Exemplos de variáveis de ponto de acoplamento incluem massa foliar ou área, massa de caule, massa de raiz, comprimento de raiz e massa ou número de sementes, todos podem ser negativamente influenciadas por pragas e doenças. Ao identificar vias e taxas de dano, usando estas variáveis (por exemplo, uma infestação de lagarta pode ser definida como a redução de área foliar em 10% até o 40º dia após o plantio), modelos de crescimento podem quantificar como o desenvolvimento das culturas seriam afetados e, em última análise, qual seria o impacto sobre o rendimento da cultura.

O conceito de ponto de acoplamento foi introduzido pela primeira vez em 1983 [29], e mais tarde formalmente implementado na plataforma de modelagem de culturas DSSAT [12, 13] como um

Módulo de *Pest* [30]. O Módulo de *Pest* permite aos usuários a entrada de dados observados e a aferição de dados sobre danos causados por insetos, intensidade da doença, danos físicos às plantas e aos diferentes órgãos (por exemplo, folhas, colmos, vagens e grãos). Permite, ainda, simular os prováveis efeitos das pragas e doenças no crescimento das culturas e no rendimento econômico.

Para a implementação do presente estudo, dois modelos de simulação desenvolvidos em linguagens de programação distintas foram escolhidos visando a inclusão dessas rotinas de acoplamento. São eles o modelo de cultura de cereais *Cropsim-Wheat*, desenvolvido em Fortran, e o modelo genérico de simulação de doenças [10], desenvolvido em Java.

Em se tratando de acoplamento de modelos é extremamente importante definir-se, inicialmente, o tipo de abordagem a ser utilizada na implementação. Isto porque o grau de complexidade envolvido nas interações entre os modelos é elevado. Pode-se citar a necessidade de conhecimento dos modelos, linguagens de programação distintas, parâmetros incompatíveis, pouca documentação, entre outros.

Definiu-se o uso da abordagem de comunicação para o acoplamento dos modelos de doenças parametrizados. Nesta abordagem, os modelos são executados como programas independentes e interagem entre si, durante a execução, por meio da troca de dados via mensagens [22]. Esta troca de dados envolve os parâmetros necessários para que os modelos consigam dar sequência na própria execução. Portanto, não fez-se necessária uma reprogramação significativa para acoplar os modelos. Usou-se uma função de *middleware* escrita em C para permitir a comunicação entre as linguagens Java e Fortran.

3.1.1 Aplicações independentes

As primeiras definições, da abordagem de comunicação, se referem a definição dos fluxos de comunicação, se há necessidade de transformação de dados e como se dará a inicialização dos modelos. Outra questão a analisar é se na interação entre os modelos serão utilizadas aplicações independentes ou não.

Para intermediar a comunicação entre os modelos algumas tecnologias foram analisadas, visando identificar a que melhor atenderia aos desafios propostos por este estudo. Ou seja, que tivesse um padrão de arquitetura web e que fornecesse suporte ao uso por diferentes linguagens de programação. Chegou-se, então, até o *WebSockets*, desenvolvido como parte do HTML5. Este protocolo proporciona uma conexão de *Sockets* para a internet com sobrecarga mínima [31]. A comunicação via *WebSockets*, diferentemente da atual, baseada em *Hypertext Transfer Protocol* (HTTP), define um canal de comunicação *full-duplex* sobre o qual podem ser enviadas mensagens entre o cliente e o servidor.

No modelo de requisição e resposta padrão, implementado pelo protocolo HTTP, o cliente faz uma requisição ao servidor iniciando a comunicação, o servidor ao devolver a resposta finaliza a conexão. Para simular uma ligação *full-duplex*, neste cenário, é necessária a utilização de outras tecnologias como Ajax, Flash ou Comet mantendo duas conexões HTTP.

Para o uso de *WebSockets*, nenhum *software* adicional, biblioteca ou *plugin* precisa ser instalado ou configurado. Basta que o cliente, no caso um navegador ou uma linguagem de programação suporte este protocolo. Uma nova instância cliente de *WebSocket* é inicializada por meio de uma *Uniform Resource Locator* (URL). Esta URL representa o ponto final, ou *endpoint*, do servidor, inicia com ws:// ou wss://. O “s” a mais na segunda URL torna a comunicação criptografada, aumentando a segurança no tráfego dos dados.

A maioria dos navegadores atuais fornecem suporte a esta tecnologia por utilizar o protocolo HTTP. Possui implementações para C++, Java, Ruby, dentre outras linguagens de programação. À medida que se busca abranger vários modelos, desenvolvidos em diferentes linguagens de programação, este é um fator importantíssimo.

WebSockets vem sendo considerado o próximo passo evolutivo em comunicação via Web, visto que simplifica bastante a complexidade em torno da comunicação bidirecional e gerenciamento de conexões [32]. Oferece uma enorme redução no tráfego de rede e latência em comparação com outras tecnologias aplicadas na implementação da comunicação em tempo real entre cliente e servidor [31]. Trata-se, portanto, de uma excelente tecnologia de comunicação em tempo real na Web. Esta arquitetura permitirá que diferentes modelos, em diferentes localizações, desenvolvidos em diferentes linguagens de programação, se conectem e troquem mensagens entre si rapidamente.

Esta tecnologia nos entrega a capacidade de preservar o desenvolvimento de sistemas especializados individuais, ao mesmo tempo em que os transforma em um poderoso sistema distribuído. Cada subsistema, portanto, pode empregar plenamente os serviços de outro subsistema. Mais importante nesta estrutura é que os usuários finais não se preocuparão com a resolução de tarefas individuais em diferentes sistemas ou com diferentes dados ou requisitos de entrada.

3.1.2 Fluxo de comunicação

O fluxo de comunicação segue o estrutura disponibilizada pelo *WebSockets*, onde qualquer modelo pode enviar mensagens aos demais modelos conectados. Cada modelo é um cliente que se conecta a um servidor de *WebSockets*. Toda a mensagem passa pelo servidor que verifica para quem deve direcioná-la. Para que a mensagem seja entregue para o destinatário correto, todo o modelo que conecta ao servidor utiliza uma chave. Esta chave, que é única, é gerada na configuração da execução. Modelos rodando acoplados na mesma simulação possuem a mesma chave.

Os modelos de simulação de doenças, baseados em dados de um modelo de cultura, verificam a incidência de uma determinada doença, executam equações e devolvem as informações para um modelo de cultura. Esse processo é sincronizado pelo servidor de *WebSockets*, que direcionada as mensagens ora para um tipo de modelo, ora para outro tipo de modelo. Este fluxo é repetido até o final da execução da simulação.

3.1.3 Transformação dos dados

De acordo com a abordagem escolhida para o acoplamento, onde os modelos devem se manter independentes, buscou-se não fazer transformações de dados durante o processo de interação dos modelos. Os dados necessários são repassados pelos modelos no formato *JavaScript Object Notation* (JSON), conforme especificação fornecida.

A escolha do formato JSON se dá, principalmente, em virtude de o mesmo ser leve, condição indispensável para troca de dados em estruturas distribuídas. Criado como alternativa ao uso de XML, objetiva a troca de informações entre sistemas de uma maneira intuitiva, como é o XML, mas ao mesmo tempo menos burocrática e com menos dados trafegando [33]. O formato JSON disponibiliza três tipos básicos de dados [34]: boleano (*true* ou *false*), texto (*string*) e numérico (*number*). A partir da combinação destes três tipos pode-se representar objetos (JSON *Object*) e listas (*arrays*). Para seres humanos o formato texto torna a leitura e a escrita fáceis. Para máquinas, é simples de interpretar e de gerar [35]. De caráter adaptável e completamente independente de linguagem de programação fornece aos usuários grande autonomia, pois permite que as informações se adaptem de um acoplamento para outro, caracterizando-se num formato ideal de troca de dados [35].

Além do formato escolhido a adoção de um padrão de mensagens é necessária para facilitar o intercâmbio de informações entre os modelos. Essa padronização envolve nomes e códigos de variáveis, definições de conjuntos de dados e suas relações. O *International Consortium for Agricultural Systems Applications* (ICASA)[3] definiu normas que descrevem detalhadamente cada um desses códigos e nomes, que se destinam a apoiar qualquer atividade relacionada às experiências de campo, ou situações de produção comercial.

As normas ICASA representam uma tentativa de promover um consenso na forma de organizar os dados [36]. Tais normas difundidas no meio, vêm sendo adotadas pelo *Agricultural Model Intercomparison and Improvement Project* (AgMIP)[4] na construção de ferramentas. O principal uso tem sido a modelagem de simulação, em que são utilizadas para representar condições iniciais, insumos climáticos, insumos e manejo de solo, em situações de cultivo específicas [36]. A definição completa desta norma está acessível no site da DSSAT *Foundation* (dssat.net/data/standards).

Optou-se pelo uso desde padrão, pois necessitava-se de um vocabulário uniforme. Facilitou a escolha o fato deste ser adotado por modelos e ferramentas utilizadas nesse trabalho. Pôde-se focalizar os esforços no problema de pesquisa principal, quando se teria que formular nomeações de campos das mensagens. Os dados, portanto, são compartilhados entre os modelos utilizando a nomeação de variáveis definida por esta norma, como um conjunto de chave e valor, no formato JSON.

[3] ICASA (International Consortium for Agricultural Systems Applications): formado em 1993, reúne um grande número de cientistas de diversas universidades e centros de pesquisa do mundo, os quais trabalham de forma colaborativa para a construção dos modelos e como aplicá-los. O grupo encerrou as atividades, sob esta definição, em 2011 sendo os avanços do conhecimento repassados para outros grupos, como o AgMIP, por exemplo.

[4] O AGMIP (Agricultural Model Intercomparison and Improvement Project) é um esforço internacional importante ligando as comunidades das áreas de climatologia, modelagem e simulação de cultivos e econometria, com tecnologia de ponta para produzir modelos de simulação agrícolas e econômicos mais avançados, incluindo a próxima geração de projeções de impacto climático para o setor agrícola.

Não havendo a necessidade de transformação nos dados, e definido o formato e padrão das mensagens, é fundamental definir como os modelos se comunicarão com o servidor de *WebSockets*. Para estabelecer esta comunicação, rotinas de acoplamento foram criadas para que os detentores do código fonte alterem, conforme orientações, modelos próprios ou conhecidos. Nestas rotinas são incluídos os parâmetros necessários para que os demais modelos consigam dar sequência a própria execução. Isso faz com que os modelos falem a mesma língua e se estabeleça uma comunicação entre eles.

Tendo-se definidos os modelos de simulação e criadas as rotinas de acoplamento para estes modelos fez-se necessário o teste desta integração como forma de aferir e validar as rotinas. Como resultado dessa validação, a rotina é disponibilizada conforme o modelo selecionado. Tomando como exemplo um modelo em Fortran, esta rotina direcionará para um código em C++, também disponibilizado. Esse código em C++ contém toda a implementação necessária para realizar a conexão com o servidor de *WebSockets*. Outros arquivos parametrizáveis serão disponibilizados, como o que define a URL de conexão, a chave de acesso única, o tipo de modelo e de execução. Esta rotina também está disponível na linguagem de programação Java.

3.1.4 Inicialização dos Modelos

O ponto central é um servidor de *WebSockets* conhecido por todos os modelos configurados para a execução. A comunicação inicia quando todos os modelos, de mesma chave, estiverem conectados ao servidor de *WebSockets*. Cada modelo a seu tempo executa instruções com base nos dados de entrada e gera um saída que será compartilhada com os demais modelos, de mesma chave, conectados no servidor.

A inicialização da simulação sempre necessitará que um primeiro modelo solicite a conexão ao servidor. A plataforma, utilizando a chave fornecida por este modelo, verifica quais modelos estão configurados nesta execução e inicializa os mesmos. Quando a execução envolver um modelo do usuário, executando remotamente, este modelo será o responsável por iniciar a execução da simulação acoplada. O fluxo de comunicação entre os modelos dar-se-á entre as fases de execução 5 e 8 do diagrama de funcionamento da plataforma *Coupling* (Figura 8).

3.2 FONTES DE DADOS DE EXPERIMENTOS

Seguindo a estratégia de aproveitamento do conhecimento legado, proporcionado por outros pesquisadores, buscou-se a utilização de bases de dados de experimentos conhecidas para o fornecimento dos dados necessários às execuções de modelos acoplados.

Dentre estas bases de dados de experimentos estão o *AgMIP Crop Experiment* (ACE) [37] e o *Agronomy Database* (AgroDB) [38]. Ambos os bancos de dados contêm dados de experimentos mais ou menos detalhados para uma variedade de cultivares. Os dados do ACE são originados por centros internacionais de pesquisa agrícola, universidades e o setor privado [39]. Implementado sobre

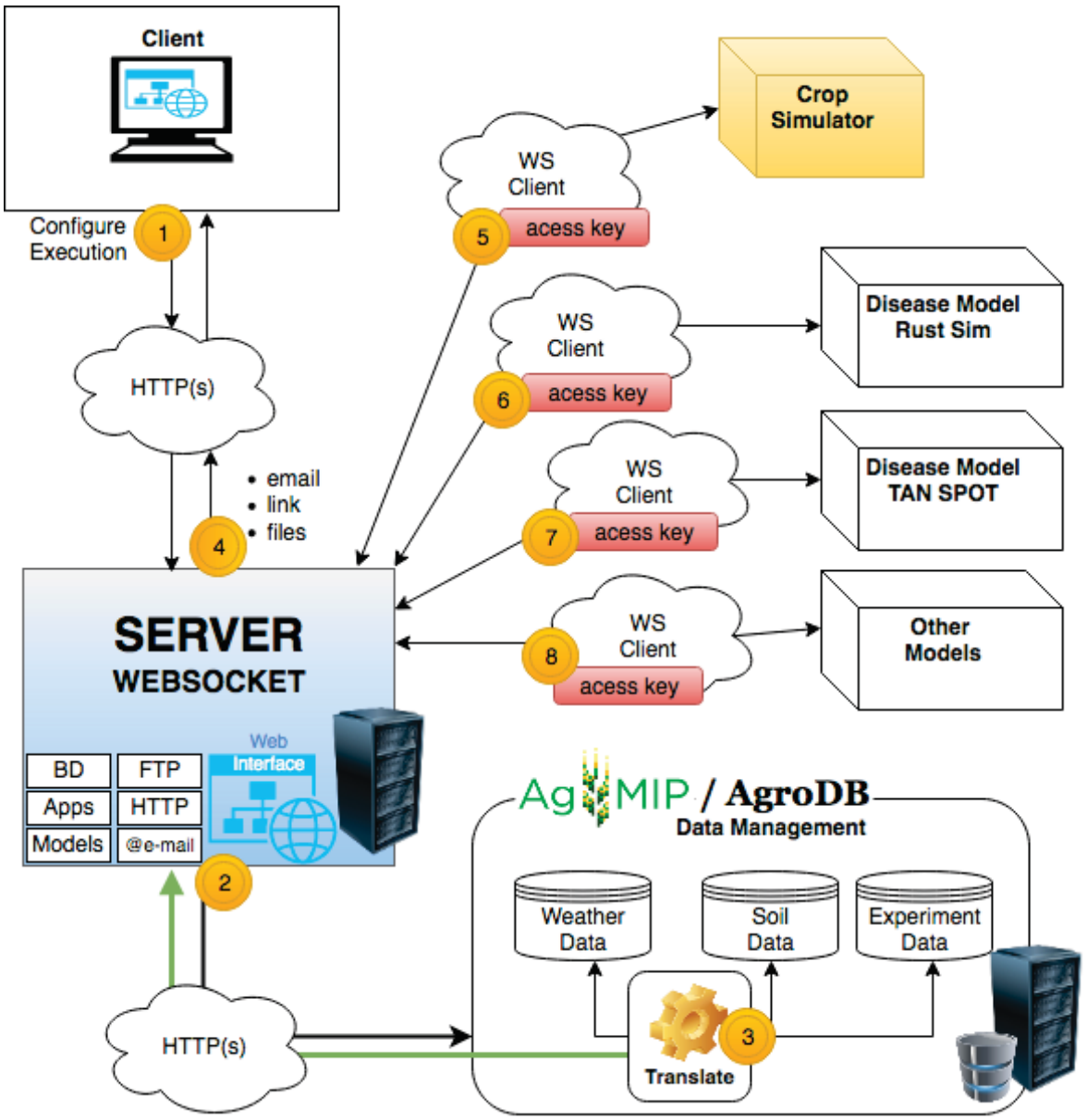

Figura 8. Estrutura de comunicação da Plataforma Coupling

uma plataforma não relacional, o ACE prevê o armazenando de milhares de experimentos e o acesso simultâneo por uma comunidade global de usuários [36]. Os dados são classificados em *buckets*[5], em vez de várias tabelas, que armazenam chaves e valores. As chaves que são únicas, combinadas com o armazenamento e indexação dos valores em um subconjunto pesquisável, proporcionam um rápido retorno para uma pesquisa dentro do banco de dados [39]. No ACE, os dados são armazenados usando estruturas JSON, onde o padrão ICASA é aplicado [36].

Já o AgroDB é um esforço de Lazzaretti [38], constituindo-se como um repositório de dados, que possibilita a importação e exportação de arquivos contendo dados em diversos formatos. A exportação, em um formato específico, é útil na execução de modelos de simulação, já que estes trabalham com dados de entrada. Enquanto que a importação representaria os resultados da simulação, por

[5]Bucket: O bucket é considerado uma unidade de armazenamento. Um bucket normalmente armazena um bloco de disco completo, que por sua vez, pode armazenar um ou mais registros.

meio da integração de um modelo com o AgroDB. Implementado no SGBD PostgreSQL, segue o paradigma objeto-relacional, em que os dados são armazenados em estruturas denominadas tabelas. Para auxiliar na manipulação e interpretação dos dados, utiliza-se das tecnologias fornecidas pelo SGBD escolhido, como linguagem *Structured Query Language* (SQL) e linguagens procedurais [38].

Bavaresco [40], disponibiliza uma ferramenta Web para consulta e inserção de dados de experimentos no banco de dados AgroDB. O cadastro de um novo experimento segue o formato passo a passo. Ao todo são onze passos, nos quais os dados são organizados por relação, são eles: dados principais, datas do experimento, cultivar do experimento, datas de semeadura, localidade, anos de experimento, resultados desejados, parâmetros de controle, condições iniciais do solo, fertilizantes e finalizar. O esforço de Bavaresco [40], permite expandir outros recursos do AgroDB que não são relevantes neste estudo. Cabe ressaltar, no entanto, a facilidade oferecida por esta ferramenta na importação de dados de experimentos para o AgroDB, incluindo ainda a possibilidade de clonar Experimentos já existentes como base para uma nova inclusão.

Para acesso aos dados do AgroDB via Web, Bavaresco disponibilizou um serviço, o qual chamou de *AgroService*. Tratam-se de *WebServices* baseados em *REpresentational State Transfer* (REST), que através de duas *Uniform Resource Identifier* (URIs), manipuladas por meio de operações de HTTP, devolvem, respectivamente, todos os experimentos cadastrados para uma determinada cultura e os dados completos de um experimento específico, incluído dados de clima e solo. O retorno é no formato JSON, seguindo os padrões definidos pelo ICASA.

O uso da tecnologia de *WebServices* REST, permitiu o consumo transparente das informações contidas no AgroDB, pela plataforma *Coupling*, sem a necessidade de acesso especial e direto ao Banco de Dados. Estes serviços representam o elo de integração entre os três sistemas, dando dinâmica à inserção e ao uso de novos experimentos. São utilizados na plataforma *Coupling* na área de cadastro de uma nova execução.

3.3 CONFIGURANDO EXECUÇÕES NA WEB

Parte essencial no processo, o usuário, em uma área privada da plataforma, configura as suas execuções (Fase 1 da Figura 8). Para cada execução deve-se informar qual a cultura e o modelo de cultura será utilizado, bem como o(s) modelo(s) de doenças que deseja-se acoplar. Dados de experimentos, pré-configurados, estão disponíveis por meio de duas fontes de dados: ACE e AgroDB. Os dados dos experimentos incluem dados de clima e de solo. Após a escolha da origem dos experimentos, é necessária a seleção de um dentre os listados. Acessível por meio de um navegador Web, da preferência do usuário, a tela de inserção de uma nova execução pode ser visualizada na Figura 9.

Utilizando-se de padrões de projetos a plataforma, além do servidor de *WebSockets*, disponibiliza para usuários cadastrados uma área pessoal para gerenciamento das próprias execuções de modelos acoplados. Para o acesso como usuário registrado, basta acessar o endereço da plataforma e preencher um pequeno cadastro. O gerenciamento envolve a criação, a listagem, a exclusão e a seleção de execuções para que rodem acopladas em uma mesma simulação.

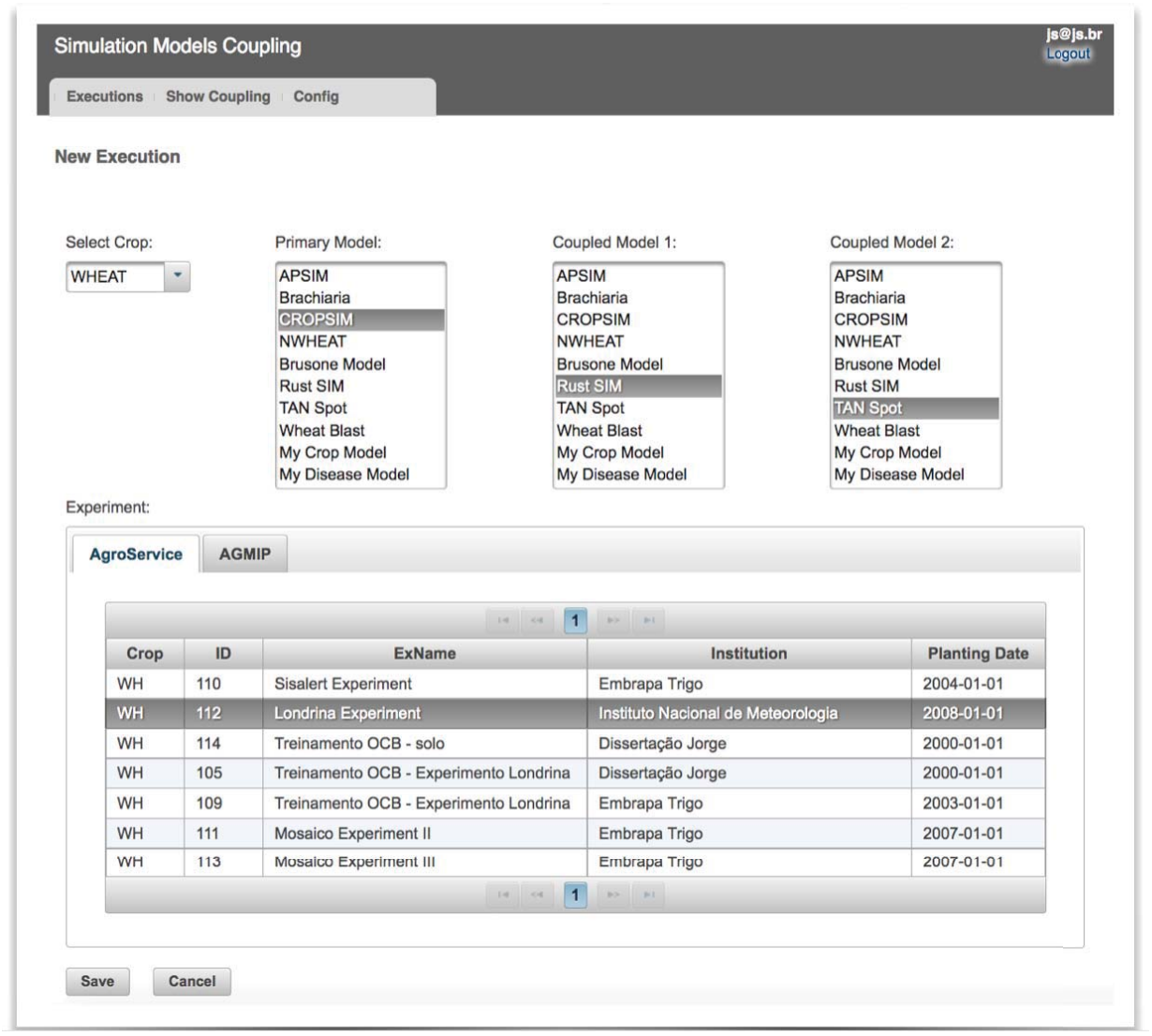

Figura 9. Tela de adição de uma nova execução de modelos acoplados

Na concepção da plataforma toda e qualquer interação do usuário foi estruturada utilizando-se camadas, no padrão Modelo/*Model*, Visão/*View* e Controle/*Controller* (MVC). Esse padrão de arquitetura tem como ideais centrais a reusabilidade de código e a separação de conceitos. A estruturação da plataforma de maneira modular, visa que partes funcionem independentemente e que novos módulos sejam integrados facilmente no futuro. Nas suas respectivas camadas utilizou-se as seguintes tecnologias: *Java Persistence API* (JPA), como padrão para a persistência de dados; *Java Server Faces* e *Bean Validator*, para tratar e validar as requisições do usuário; e a biblioteca *Primefaces* para componentes da interface.

As tecnologias escolhidas para o desenvolvimento permitem que qualquer Gerenciador de Banco de Dados venha a ser utilizado. Inicialmente, optou-se pelo uso do PostgreSQL para armazenamento das configurações da plataforma e das execuções definidas pelos usuários.

Na submissão de uma nova execução (Figura 9), após a validação dos dados obrigatórios, a plataforma inicializará um processo de montagem da estrutura de execução. Este processo inclui a geração de uma chave de acesso única para a execução cadastrada. A chave e os dados informados

são, então, persistidos no banco de dados. Esta chave, como já comentado, será utilizada pelos modelos vinculados a execução como forma de identificarem-se para o servidor de *WebSockets*. O uso de chaves permitirá que várias execuções possam rodar ao mesmo tempo, no mesmo servidor.

Ainda, no processo de preparação do ambiente de execução, providencia-se na fonte de dados especificada as informações do experimento selecionado (Fase 2 da Figura 8). A alimentação de diferentes modelos de simulação, de maneira dinâmica e transparente, só é possível por meio da conversão dos dados de experimentos para arquivos que os modelos conheçam e consigam interpretar. Para esta tarefa de tradução dos experimentos (Fase 3 da Figura 8), em arquivos de entrada para modelos de simulação de culturas conhecidos, utilizou-se a ferramenta QuadUI, desenvolvida pelo AgMIP.

O QuadUI integra a funcionalidade de tradutores e com base nos parâmetros recebidos providencia modificações dos dados brutos para atender a necessidade do(s) modelo(s) informado(s) [41]. Os tradutores são responsáveis pela geração dos arquivos para cada modelo. Além do arquivo representando o Experimento, a ferramenta, também disponibiliza os arquivos de Solo e Clima a partir dos dados do experimento. Entre outros, são arquivos essenciais para execução do modelo de simulação de cultura. O uso do QuadUI pela plataforma *Coupling*, permitiu focalizar a pesquisa em questões próprias da estrutura, ao invés de assuntos específicos de cada modelo. A versão de linha de comando do QuadUI foi a escolhida e os dados dos experimentos são passados para a ferramenta no formato JSON.

Na área de seleção dos modelos (Figura 9) dois representam modelos próprios dos usuário. São eles: *My Crop Model*, indicado para o acoplamento de um modelo próprio de cultura e *My Disease Model*, indicado para o acoplamento de um modelo próprio de doenças. Caso algum desses modelos seja selecionado uma condição adicional é realizada. Localizam-se as rotinas e instruções de acoplamento, bem como os arquivos compilados para diferentes plataformas e linguagens de programação. Junto a esses arquivos se somam os arquivos de configuração, que incluem URL de conexão, chave de acesso da execução, entre outros. Tudo é compactado em um único arquivo e enviado para o e-mail informado no cadastro do usuário que configurou a nova execução. Um exemplo de e-mail enviado ao usuário, contendo os arquivos e as instruções para adição da rotina de acoplamento ao modelo próprio, pode ser visualizado na Figura 10.

Ao receber o e-mail o responsável, conforme instruções, precisa adicionar a rotina ao modelo próprio e compilá-lo juntamente com os arquivos disponibilizados. Ao término desta etapa poderá, então, executar o modelo próprio alterado, que solicitará a conexão ao servidor de *WebSockets*, iniciando o processo de acoplamento. De onde quer que esteja, somente necessitando de uma conexão com a internet.

3.4 CONECTORES JAVA E ROTINAS DE ACOPLAMENTO

Com o objetivo de simplificar o processo de acoplamento de modelos construídos em Java, dois conectores distintos foram desenvolvidos. Um para o uso com modelos de cultura e outro para

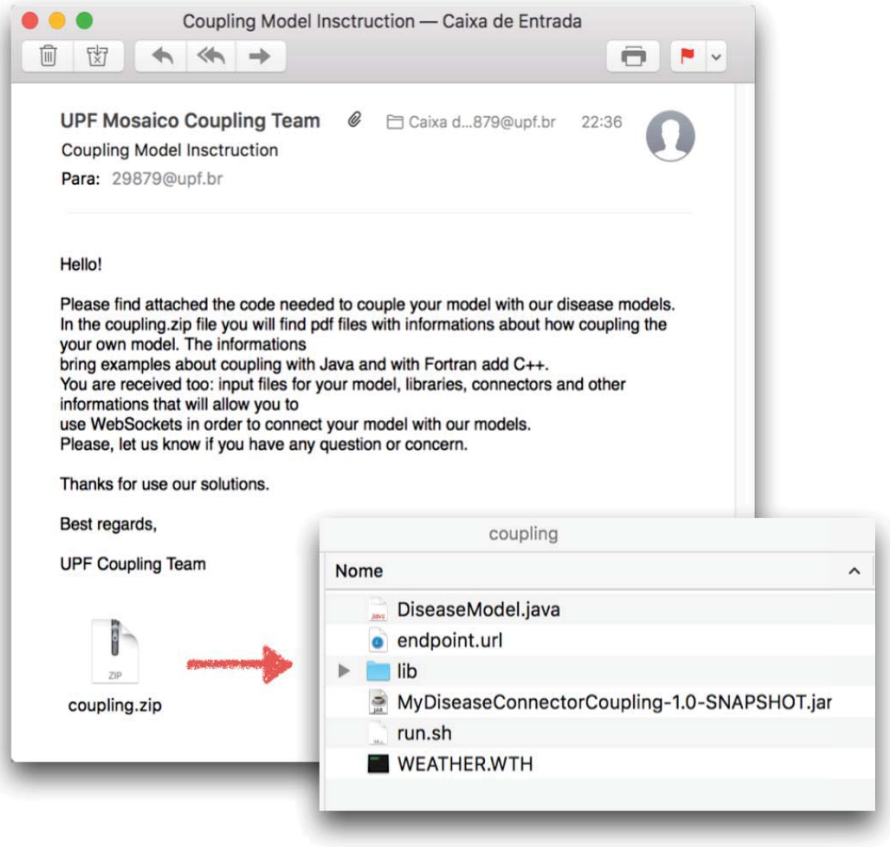

Figura 10. Exemplo de e-mail contendo instruções e rotinas de acoplamento a modelos próprios

o uso com modelos de doença. A diferença principal está nos parâmetros que um tipo ou outro deve enviar e receber. Esses conectores se encarregam de iniciar e finalizar a conexão com o servidor de *WebSockets*, além de monitorar as trocas de mensagens.

De acordo com o tipo de modelo próprio selecionado o usuário receberá o conector adequado, bem como as bibliotecas necessárias, no e-mail informado. Pode-se visualizar na Figura 10, o recebimento do conector para um modelo de doenças. Ambos os modelos podem ser executados através do arquivo run.sh, como forma de visualizar o funcionamento dos modelos acoplando e trocando mensagens. Os dados dessa execução são totalmente aleatórios, sem valor científico.

Um arquivo com a extensão java também é enviado no intuito de exemplificar como o conector deve ser utilizado. Na listagem dos anexos(Figura 10), este arquivo aparece com o nome de DiseaseModel.java. Seguindo a lógica do arquivo, o usuário necessita alterar o modelo próprio de forma que as variáveis esperadas sejam informadas a seu tempo. Para garantir que estas variáveis sejam enviadas/recebidas no número e formato corretos, criou-se uma Interface no Conector. Esta Interface exige a implementação do método '*coupler*'. Neste método, os parâmetros recebidos devem ser utilizados para dar sequência ao funcionamento do modelo próprio e devolver o resultado que dará continuidade ao processo de integração.

Cabe ao usuário identificar, no modelo próprio ou conhecido, onde a Interface deve ser implementada, para que os dados recebidos sejam utilizados de forma correta. Da mesma forma que o retorno da informação permita ao modelo acoplado, do outro lado, continuar a execução.

Quanto aos modelos desenvolvidos em Fortran, por estes não implementarem recursos de comunicação HTTP, um *middleware* em C++ foi desenvolvido. Da mesma forma que os conectores Java, este *middleware*, inicializa e finaliza a conexão com o servidor de *WebSockets*, além de enviar as mensagens ao servidor e gerenciar os retornos. Para o uso de *WebSockets* no C++ utilizou-se a biblioteca *Easywsclient* (disponível em: https://github.com/dhbaird/easywsclient), que implementa esses recursos de comunicação utilizando apenas bibliotecas padrões do C++ [42].

Os códigos em C++ e as bibliotecas necessárias, são anexadas ao e-mail juntamente com uma rotina de acoplamento. Esta rotina precisa ser incluída no código Fortran, depois de declarada. Cabe ao detentor ou conhecedor do modelo identificar o local correto para a adição desta rotina. É ela que fará a comunicação com o código em C++. Por fim, o modelo Fortran e o *middleware* C++ precisam ser compilados em conjunto. Passos para a compilação do modelo são disponibilizados para o sistema operacional Linux e MacOS. Instruções de acoplamento em um arquivo PDF também são disponibilizadas.

3.5 EXECUÇÃO DE MODELOS ACOPLADOS

Passada a fase de criação das execuções, que rodarão os modelos acoplados, o usuário, através de uma área restrita, visualiza suas execuções, tendo acesso às principais informações configuradas para cada execução (Figura 11). Estas informações incluem a data de criação, a chave única, a cultura, a data de início da simulação, a data de plantio, os modelos que acoplarão, entre outros. O usuário, ainda pode fazer a chamada ao formulário de criação de uma nova execução ou excluir uma já existente. O recurso principal, no entanto, é a seleção de uma ou mais execuções para que executem modelos acoplados. Nesta ação os modelos vinculados a cada execução conectam-se ao servidor de *WebSockets*, iniciando-se o processo de troca de mensagens entre eles. O botão *Run*, em destaque na Figura 11, dá início ao processo de acoplamento.

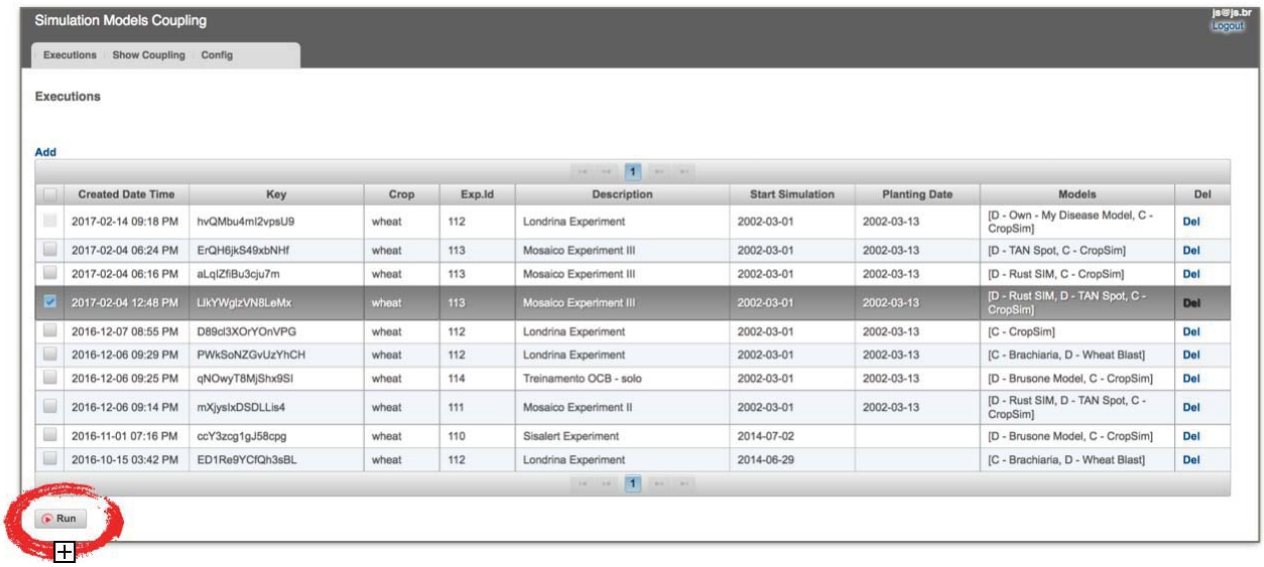

Figura 11. (a) Listagem das execuções configuradas. (b) Iniciando o processo de acoplamento.

Execuções envolvendo modelos próprios não estão disponíveis para seleção e início da execução pela plataforma. Estes modelos é que, remotamente, inicializarão a comunicação conectando-se ao servidor de *WebSockets*.

Ao solicitar o início de uma execução o usuário será direcionado para a tela representada na Figura 12. Criou-se esta interface para que se pudesse acompanhar a execução e as trocas de mensagens entre os modelos conectados. No exemplo (Figura 12), os modelos conectam-se ao servidor de *WebSockets* por meio da URL ws://localhost:8080/coupling/couplingendpoint, utilizando a chave única da execução que tem o valor 'LlkYWglzVN8LeMx'. O instante representado na Figura 12 é o final da execução. O quadrante da esquerda é destinado ao modelo de simulação de cultura, no qual pode-se visualizar informações de conexão e desconexão, além das entradas e das saídas envolvendo este modelo. Enquanto que o quadrante da direita é destinado para os modelos de simulação de doenças. As informações representadas são as mesmas do modelo de cultura.

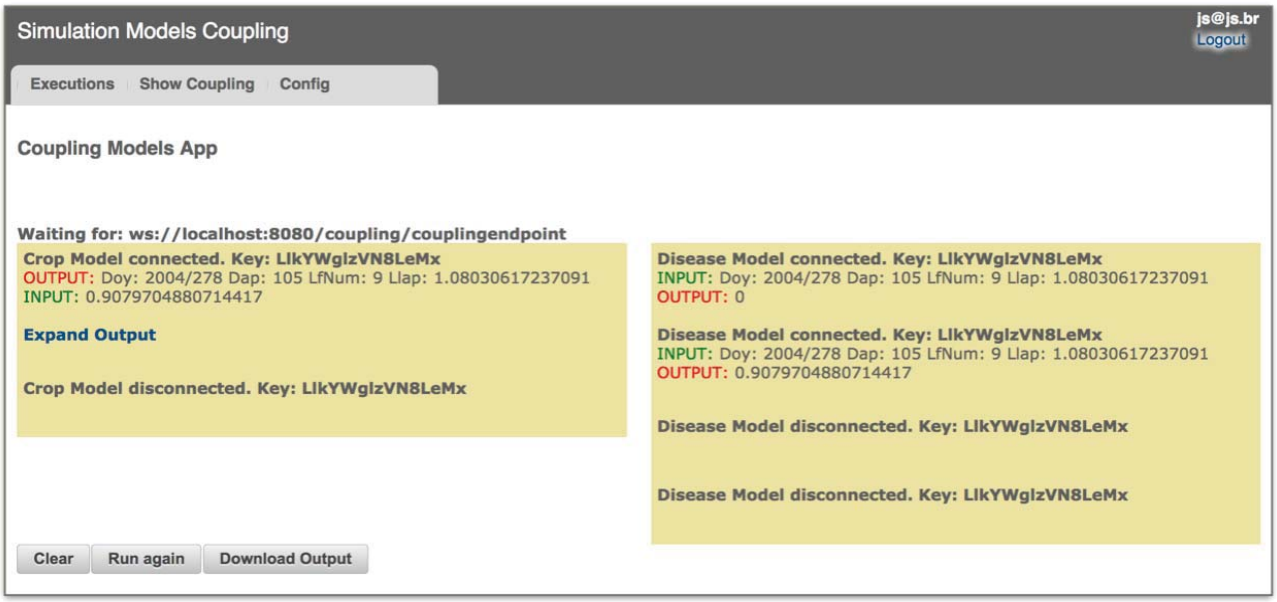

Figura 12. Tela de acompanhamento da execução de modelos acoplados

As mensagens enviadas caracterizam a comunicação entre eles. A saída de um modelo de cultura será a entrada para os modelos de doenças acoplados, e vice-versa. Cada mensagem que o modelo de cultura envia (*crop output*) é encaminhada para os modelos de doenças no mesmo instante. Os modelos de doença recebem (*disease input*) estes dados e os utilizam para calcular a incidência da doença que representam, devolvendo informações para o modelo de cultura (*disease output - crop input*). A frequência de troca de mensagens será determinada pelo modelo de cultura, geralmente com cálculo diário. Caso o modelo de cultura simule o ganho de massa foliar por meio de várias folhas, as trocas serão multiplicadas pelo número de folhas simuladas por este modelo. Cada modelo após concluir a execução é desconectado do servidor.

Expandindo as saídas do modelo de cultura, através do link '*Expand Output*' (Figura 12), visualiza-se as mensagens trocadas entre os modelos a cada cálculo diário. Essas mensagens podem

ser baixadas através do botão '*Download Output*', juntamente com os arquivos de saída do modelo de simulação de cultura.

A seleção de mais de uma execução, e a posterior inicialização destas, considera a possibilidade de as execuções trocarem informações entre si. Nestes casos as execuções serão ordenadas pela data de início da simulação. Somente quando a primeira atingir a data de início da simulação da segunda execução é que esta será iniciada, e assim sucessivamente. Enquanto as execuções estivem rodando no mesmo intervalo de tempo, as que rodarem depois poderão fazer uso dos dados obtidos pelas execuções anteriores. Esta situação é útil quando se deseja considerar, por exemplo, a existência de uma nuvem de esporos no campo. Ou seja, o que se vislumbra aqui é a possibilidade desta nuvem ter sido gerada e se desenvolvido por outra cultura próxima ao local onde a cultura de interesse está sendo cultivada. Cada valor obtido é armazenado e disponibilizado para as demais execuções concomitantes. No entanto, o uso destas variáveis compartilhadas precisa ser considerado por cada modelo. A ilustração deste funcionamento pode ser observada na Figura 13, na qual se vê a primeira execução conectando e gerando dados. Mais tarde a segunda execução conecta-se e passa a fazer uso dos dados produzidos pela execução anterior nos mesmos dias de simulação.

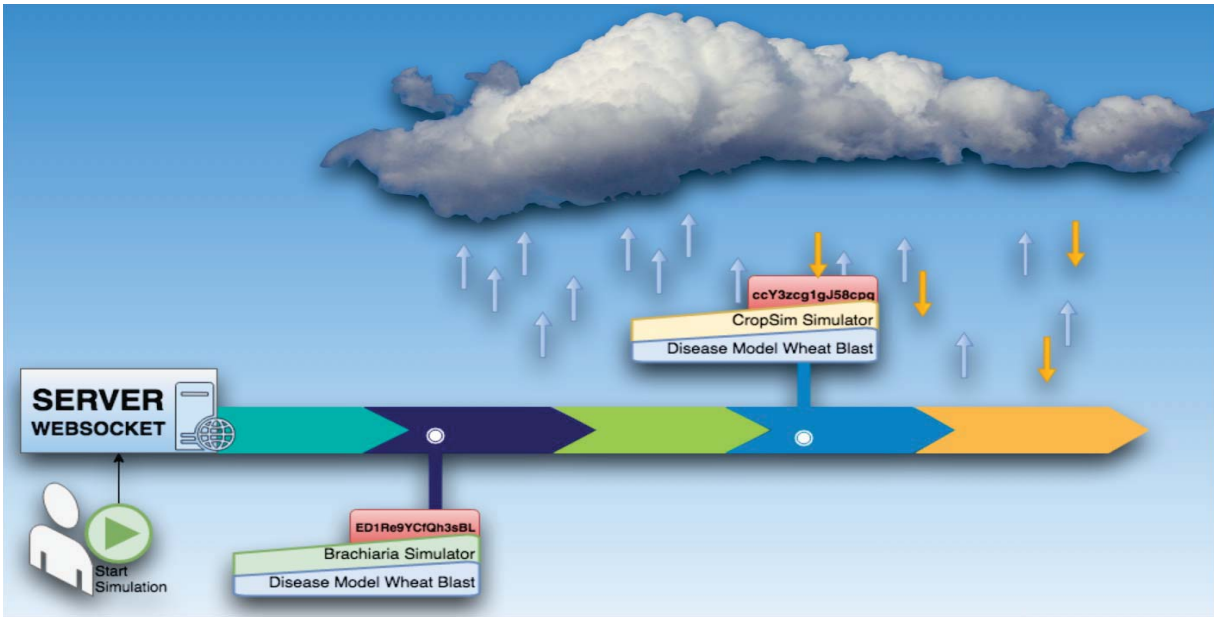

Figura 13. Representação de várias execuções rodando ao mesmo tempo e trocando informações

4. ACOPLAMENTO E TESTE DE MÓDULO DE DOENÇAS NO MODELO DSSAT CROPSIM-WHEAT PARA A PRODUÇÃO DE TRIGO, NO PLANALTO MÉDIO DO RIO GRANDE DO SUL

4.1 RESUMO

Esse trabalho tem como objetivo testar o acoplamento de um modelo de simulação de cultura a modelos de simulação de doenças, por meio da plataforma *Coupling*. A validação dos resultados necessitou de dados de experimentos de campo, que foram obtidos para um grupo de cultivares, indicadas para cultivo no Planalto Médio do Rio Grande do Sul. Os ensaios de campo foram realizados nos anos de 2003 e 2004, em Passo Fundo, RS. Danos causados pela ferrugem da folha do trigo, causada por *Puccini triticina*, e mancha amarela, provocada por *Pyrenophora tritici-repentis*, foram medidos nos ensaios e utilizados na comparação com os danos gerados por meio do acoplamento de modelos, que simulam essas doenças. Os modelos envolvidos precisaram ser calibrados para que simulassem as condições encontradas em campo e representassem as doenças incidentes. Os resultados mostraram que as parametrizações realizadas nos modelos tiveram uma projeção muito próxima da evolução da doença na cultura, descrita pela observação em campo. Conclui-se que o acoplamento proporcionado pela plataforma *Coupling*, auxilia no processo de avaliação do impacto de doenças no crescimento e no rendimento final de grãos de uma planta.

4.2 INTRODUÇÃO

Pragas agrícolas e doenças são responsáveis por perdas de rendimento diretos que variam entre 20 e 40% da produtividade agrícola global e regularmente ameaçam a segurança alimentar mundial [1, 2]. No entanto, as perdas da colheita se mantem pouco reconhecidas como um fator preponderante em matéria de segurança alimentar, ao passo que as doenças das plantas tiveram um enorme impacto sobre os meios de vida ao longo da história humana [1]. O trigo é um dos cereais com maior área de produção, atingindo cerca de 215 milhões de hectares em todo o mundo [3], e as pragas e doenças continuam a ser um dos principais entraves para a produção de trigo.

Devido a crescente demanda por produtos agrícolas e o aumento das pressões sobre o uso da terra, da água e de outros recursos naturais, a informação para tomada de decisão agrícola é essencial em todos os níveis. Ao longo dos tempos a ciência vem tentando antever fenômenos e prevenir epidemias com o objetivo de minimizar riscos e aumentar a produtividade. A geração de novos dados através de métodos experimentais agronômicos tradicionais, apesar de sua utilidade para responder questões específicas, não são suficientes para atender tais necessidades que são crescentes. Experimentos agronômicos tradicionais são realizados em pontos específicos no tempo e no espaço, o que torna os resultados, além de demorados e caros, locais e específicos de cada temporada.

Ferramentas importantes neste processo, os modelos de simulação computacional, auxiliam na criação de cenários, identificação de situações atípicas e geração de alertas, geralmente baseados em dados históricos. Modelos de simulação de culturas capazes de predizer o rendimento final são estudados, testados e calibrados, intensivamente, por pesquisadores em várias partes do mundo [4]. Representam a simulação dinâmica do crescimento das culturas, por intermédio da integração numérica, com o auxílio de computadores. Em essência, esses modelos de simulação são programas de computadores que representam, matematicamente, o crescimento das plantas em relação ao ambiente [5]. No entanto, a maioria dos modelos de cultura opera no nível de rendimento atingível (considerando-se radiação solar, temperatura, CO_2 e precipitação), e não conta para efeitos de pragas e doenças, quer mecanicamente ou através de pós-modelo (ou seja, depois do término da simulação do crescimento/desenvolvimento da cultura).

Os simuladores de epidemias de doenças ou surtos de praga, por outro lado, são modelos criados em função da forte influência que representam no rendimento final de uma cultura. Estas doenças ou pragas têm mais propensão de se desenvolverem em algumas regiões do que em outras. Equipes têm dedicado um longo tempo para estudo e aprimoramento de modelos que simulam uma situação específica. O uso destes modelos vêm sendo vislumbrados como uma ferramenta de apoio a tomada de decisão com relação ao controle de epidemias. Os resultados apresentados, através das saídas geradas, buscam auxiliar na escolha de estratégias de manejo, visando manter um baixo nível de dano para uma determinada cultura e evitar maiores prejuízos econômicos. A maioria dos modelos de simulação de epidemias, necessita, no entanto, dos dados gerados por um modelo de cultura para poder ser executado.

O uso do conhecimento legado proporcionado por esses modelos motivou a criação da Plataforma *Coupling*. A função principal é executar modelos de simulação de culturas, acoplados a modelos com capacidades para a contabilização de estresses bióticos. Rotinas de acoplamento governam a integração de modelos de simulação construídos em diferentes linguagens de programação, desenvolvidas utilizando tecnologias recentes de comunicação via Internet, permitindo o acoplamento de modelos remotos. A estrutura da plataforma é baseada na troca de mensagens entre os clientes de uma mesma execução e gerenciada por um servidor de *WebSockets*. Adotou-se um padrão de nomenclatura dos campos, muito utilizado na área de modelos de simulação agrícola, para que o intercâmbio de mensagens siga um vocabulário uniforme. As mensagens estão no formato JSON, o que as torna intuitivas, padronizadas, expansíveis e leves. Toda a comunicação é realizada quase que em tempo real, com baixíssima latência.

Na literatura encontram-se várias experiências de acoplamento, no entanto, o fato de várias tecnologias serem empregadas no desenvolvimento torna a integração custosa e muito específica em função dos modelos envolvidos. O que a plataforma *Coupling* pretende é abstrair a principal complexidade, que é comunicação entre diferentes implementações, bem como, abranger modelos que ainda não foram explorados em um processo de acoplamento, como os construídos pelo próprio usuário, visto que estes fomentam a pesquisa na área agrícola. Permitir a integração de mais de dois modelos em uma mesma execução também é um diferencial da plataforma.

Diante do exposto, o estudo teve como objetivo comprovar que o acoplamento de um modelo de cultura a modelos de doença, proporcionado pela plataforma em questão, produziram resultados condizentes com o que seria encontrado em uma situação real, sob as mesmas condições. Para efeito de comparação dados de experimentos de campo foram obtidos para um grupo de cultivares, indicadas para cultivo no Planalto Médio do Rio Grande do Sul. Os ensaios de campo foram realizados nos anos de 2003 e 2004, em Passo Fundo, RS [43]. Como prova de conceito utilizou-se a severidade de duas doenças para comparação dos dados simulados com os dados observados. Os modelos de simulação envolvidos precisaram ser parametrizados e calibrados para representar as condições encontradas no campo na época do ensaio.

4.3 MATERIAIS E MÉTODOS

4.3.1 Experimentos de Campo

Para o comparativo, dados dos experimentos de campo foram obtidos para um grupo de cultivares do trigo indicadas para cultivo no sul do Brasil. Os ensaios, que produziram estes dados, são descritos por Funck et al. [43] onde os autores buscaram determinar a relação entre a severidade das doenças foliares com a duração da área verde fotossinteticamente ativa e com o peso de grãos de trigo. As cultivares escolhidas foram BR 23, Embrapa 16 e BRS 179, e semeadas em uma área de Latos-solo Vermelho Distrófico típico, na estação experimental da Embrapa Trigo em Passo Fundo, Rio Grande do Sul. O campo experimental está localizado a 28°25' de latitude Sul e a 52°40' de longitude Oeste. A semeadura foi realizada nos dias 27 e 21 de junho, nos anos de 2003 e 2004, respectivamente, para cada cultivar, em duas faixas de 5 x 25 metros. Em uma parcela dessas faixas, para cada cultivar, visando garantir a ausência das doenças foliares, uma mistura dos fungicidas foi aplicada aos 30, 60 e 90 dias após semeadura, resultando em experimentos com e sem tratamentos.

A severidade das doenças foliares incidentes, bem como a quantidade de área foliar verde dos colmos e das folhas fotossinteticamente ativas, por meio do respectivo peso seco, foram medidas semanalmente. A cada avaliação estes valores eram estimados para todas as folhas e colmos de cinco plantas coletadas ao acaso, de cada parcela de cultivar. O percentual de área foliar com sintomas de doenças era estimado para oídio (*Blumeria graminis f. sp. tritici*), ferrugem da folha (*Puccinia triticina*) e para manchas foliares, através da observação visual, considerando infectada a folha com sinais visíveis de colonização [43].

Os experimentos observados no campo foram reproduzidos de maneira a serem simulados, com o uso do computador, usando o modelo de simulação da cultura do trigo *Cropsim-Wheat* acoplado aos modelos de doenças. Os resultados foram tabulados em arquivo de dados observados, para uso pelo modelo de cultura escolhido. Dentre as doenças consideradas no ensaio, duas foram parametrizadas no modelo de doenças para simular a sua incidência sobre a cultura. São elas: a ferrugem da folha do trigo, causada por *Puccini triticina*, e a mancha amarela, provocada por *Pyrenophora tritici-repentis*.

4.3.2 Modelo de simulação Cropsim-Wheat

Escolheu-se o modelo de cultura DSSAT *CROPSIM-Wheat* para validação da plataforma. Desenvolvido na linguagem Fortran, este modelo simula os impactos dos principais fatores ambientais, como o clima, tipo de solo e principais características, e manejo da cultura no crescimento, desenvolvimento e rendimento do trigo. Requisitos de entrada para o *CROPSIM-Wheat* incluem clima e condições do solo, características da planta e manejo da cultura [16]. Estes requisitos são informados/modificados por meio de arquivos textuais [44, 14], evitando assim a alteração de qualquer código do modelo. As extensões identificadas a seguir são de arquivos de entrada, que foram usados na parametrização do modelo, visando representar o cenário de um experimento real.

- .WTH: possui os dados meteorológicos. Descritos em cálculo diário os dados informados são: data (formato juliano) - DATE, radiação solar global (MJ m^{-2}) - SRAD, temperatura máxima do ar (^oC) - TMAX, temperatura mínima do ar (^oC) - TMIN e precipitação pluvial (mm) - RAIN. Além de um cabeçalho relacionado ao local de coleta dos dados, contendo: latitude - LAT, longitude - LON, altitude - ELEV e concentração de CO_2 na atmosfera (ppm) - CCO2.

  Os dados meteorológicos, para o experimento, foram obtidos de uma estação automática, nas coordenadas latitude -28,250, longitude -52,400 e altitude 684m, do Instituto Nacional de Meteorologia (INMET).

- .SOIL: contém os dados sobre as propriedades de solo. Os primeiros campos identificam o solo (textura, profundidade do solo, entre outros), coordenadas geográficas e outras propriedades que não variam com a profundidade. A partir desses cada linha representará uma camada do solo, com variáveis representando espessura, nutrientes, entre outros [12, 13].

  Os dados de solo para os anos de 2003 e 2004, já adaptados ao formato do *Cropsim-Wheat* são mostrados na Figura 14.

- Arquivos .CUL, .ECO e .SPE são utilizados para tratar das características morfológicas e fisiológicas de um genótipo particular [12, 13], em que o arquivo .SPE contém características específicas da cultura. As variáveis presentes nestes arquivos determinam a composição dos tecidos e processos da planta, como fotossíntese, respiração, assimilação de nitrogênio, partição de fotoassimilados, senescência, fenologia e crescimento. O arquivo .ECO contém os coeficientes de resposta da planta em relação ao ambiente (ecotipo). Enquanto que o arquivo .CUL contém as variáveis genéticas associadas às cultivares [45].

- .WHX: arquivo que gerencia a execução da simulação. Cada cabeçalho do arquivo, identificado pelo asterisco "*", indica um controle. Os principais controles são: informações sobre os tratamentos (*TREATMENTS), cultivares (*CULTIVARS), arquivos com dados de clima e solo (*FIELDS), condições iniciais do solo - água, amônio e nitrato - por camada (*INICIAL CONDITIONS), adubação de base e de cobertura - nitrogênio e fósforo (*FERTILIZERS (INORGANIC)), resíduos e fertilizantes orgânicos - matéria orgânica e nitrogênio (*RESIDUES AND ORGANIC

FERTILIZER) e parâmetros de controle do modelo - gerenciamento de água, nitrogênio, controle de doenças, módulo de cálculo da matéria orgânica, dentre outros (*CONTROLS).

```
*EBPF930001  S22PF1 150  S       150 OX
@SITE        COUNTRY          LAT     LONG SCS FAMILY
 PASSO FUNDO BRAZIL         -28.15  -52.24 -99
@ SCOM  SALB  SLU1  SLDR  SLRO  SLNF  SLPF  SMHB  SMPX  SMKE
   -99   .14     6    .6    60    .8     1   0.1  IB00  IB00
@  SLB  SLMH  SLLL  SDUL  SSAT  SRGF  SSKS  SBDM  SLOC  SLCL  SLSI  SLCF  SLNI  SLHW  SLHB  SCEC  SADC
     5   -99  .126  .226  .276     1    21  1.08  2.49     1   6.2   -99   -99   -99   -99   -99   -99
    10   -99  .126  .226  .276     1    21  1.08  2.49     1   6.2   -99   -99   -99   -99   -99   -99
    20   -99  .131  .231  .281     1    21  1.16  2.38    .4   6.3   -99   -99   -99   -99   -99   -99
    30   -99  .141  .241  .291   .73    21  1.13   2.2    .2   5.5   -99   -99   -99   -99   -99   -99
    40   -99  .141  .241  .291   .15    21  1.13  1.86    .2   4.7   -99   -99   -99   -99   -99   -99
    50   -99  .182  .282  .332   .13    21  1.12  1.74    .1   4.7   -99   -99   -99   -99   -99   -99
    60   -99  .215  .315  .365    .1    21  1.09  1.33    .1   5.2   -99   -99   -99   -99   -99   -99
    75   -99  .207  .307  .357  .005    21  1.09   .81     0   4.9   -99   -99   -99   -99   -99   -99
    90   -99  .228  .328  .378  .001    21   1.1   .52     0     5   -99   -99   -99   -99   -99   -99
   105   -99  .194  .294  .344  .001    21  1.13   .29     0     5   -99   -99   -99   -99   -99   -99
   120   -99  .218  .318  .368     0    21  1.11   .23     0     5   -99   -99   -99   -99   -99   -99
   135   -99  .215  .315  .365     0    21  1.12   .23     0   4.9   -99   -99   -99   -99   -99   -99
   150   -99  .241  .341  .391     0    21   1.1   .23     0     5   -99   -99   -99   -99   -99   -99
```

Figura 14. Configurações do arquivo de solo (.SOL), formato *Cropsim-Wheat*, usado nos experimentos simulados nos anos de 2003 e 2004

Arquivos com extensão .OUT armazenam os resultados gerados pelo modelo *Cropsim-Wheat*. Tratam-se de arquivos textuais com formatos pré-definidos, assim como os descritos anteriormente. No arquivo OVERVIEW.OUT são colocados os principais resultados das simulações, como por exemplo: o desenvolvimento da planta através dos estádios da fenologia e o rendimento de grãos. Cada variável é determinada pela unidade dias após a semeadura. Outro arquivo de saída é o LEAVESDI.OUT, que apresenta detalhes da distribuição das folhas, como área produzida, área doente e área senescente, para cada dia da simulação.

4.3.3 Modelo Genérico de simulação de Doenças

Como modelo de simulação de doenças a ser acoplado, utilizou-se o Modelo Genérico de Doenças desenvolvido por Pavan [10]. Modelo este construído de maneira modular, na linguagem de programação orientada a objetos Java, foi parametrizado para simular os ciclos das doenças da ferrugem da folha do trigo, causada por *Puccini triticina*, e da mancha amarela, provocada por *Pyrenophora tritici-repentis*. Os valores dos parâmetros para ambos os modelos de doença foram extraídos a partir da literatura ou empiricamente definidos por especialistas em doenças do trigo. Variáveis climáticas, tais como, temperatura, umidade relativa, precipitação e radiação solar, fornecidas em base horária, governam o progresso da doença. O modelo genérico de doenças foi estruturado seguindo os princípios para o acoplamento do hospedeiro e a dinâmica da doença introduzidas por Berger e Jones [46], em 1985. A dinâmica da doença foi tratada no nível *cohort* (grupo de lesões) [47]. No modelo de trigo, cada folha foi considerada como um novo tecido foliar individual decorrente de crescimento simulado e assumiu-se ser um local de infecção potencial. O início da doença resultou de um inóculo inicial no ar produzido localmente ou externamente. Sob condições ambientais favoráveis à infecção ocorre a

formação de uma lesão invisível. No final do período de latência, a lesão se torna visível ou infecciosa e aumenta com o tempo. Na fase infecciosa, uma proporção dos esporos produzidos são distribuídos em três escalas de hierarquia espacial[48]. A taxa de infecção de um local em potencial é calculado de acordo com a proporção de autodeposição, allo-folha-deposição e allo-planta-deposição. Assume-se que a área doente é o resultado da totalização de lesões com área foliar e área foliar infectada, mas com sintomas não-visíveis. Efeitos da doença sobre a cultura são estimados por meio da área doente no modelo de doença, a qual é utilizada para atualizar a variável "*Leaf Area*"estimada pelo modelo de cultura. Ou seja, o modelo de doença devolve para o modelo da cultura a proporção de área foliar doente num dado dia. Esta variável afeta diretamente a fotossíntese da planta e, consequentemente, o crescimento e a produção de grãos.

O modelo genérico de doenças disponibiliza um arquivo de entrada (config.xml) para a parametrização específica de cada modelo de doença. Configurações do processo de simulação, quantidade de plantas a simular, datas de semeadura, parâmetros epidemiológicos, inóculo inicial, fórmulas e valores para cálculo de favorabilidade, entre outras, são variáveis disponíveis neste arquivo.

Doenças de plantas compartilham um requisito comum de água livre na superfície da planta para a germinação de esporos. No entanto, a informação sobre o Período de Molhamento Foliar (PMF) nem sempre está disponível em dados meteorológicos de séries temporais. Assim, pode ser derivado de outras variáveis climáticas, como umidade relativa do ar, temperatura do ponto de orvalho, entre outras [48]. Neste trabalho utilizou-se como PMF uma estimativa baseada no número de horas com a umidade relativa acima de 90% [49], de modo que o modelo pudesse ser validado; porém, com dados mais precisos e mais confiáveis acredita-se que o modelo poderia ser melhor calibrado. Os dados diários de horas de umidade relativa maior que 90, em Passo Fundo, foram obtidos junto ao observador meteorológico da Embrapa Trigo Passo Fundo. A variável nomeada RH90 foi adicionada ao arquivo de clima compartilhado por ambos os modelos.

Outra variável importantíssima para que os modelos reflitam o real progresso de epidemias é o inóculo inicial. A principal fonte de inóculo inicial de doenças, como a ferrugem, podem ser tanto plantas voluntárias quanto outros hospedeiros. Em geral, apesar da sua disponibilidade durante a maior parte do período de colheita, o tempo de detecção e padrões de aumento da ferrugem variam amplamente de ano para ano e entre regiões de produção [21]. Estimar a quantidade de inóculo inicial é um fator limitante importante, tanto para os modelos da ferrugem da folha de trigo, quanto para os modelos da mancha amarela, sendo este um parâmetro muito sensível. Uma alternativa, especialmente se o inóculo primário é originado externamente, é fornecer diferentes cenários de densidade ao inóculo inicial [50]. Esta variável foi a que recebeu os principais ajustes para as diferentes cultivares, tratamentos, épocas e doenças consideradas no experimento.

4.3.4 Acoplamento e calibração dos modelos de simulação

Os modelos utilizados nesta validação estão disponíveis na plataforma *Coupling* para a montagem de uma execução. Tanto o modelo representando a ferrugem da folha do trigo quanto o modelo

da mancha foliar, são parametrizações do modelo genérico de doenças, e estão disponíveis sob o nome de RUSTSIM e TANSPOT, respectivamente. Enquanto que o modelo *Cropsim-wheat* está disponível sob o nome de CROPSIM. As rotinas de acoplamento, necessárias na comunicação entre os modelos e a plataforma, já estão integradas a estes modelos.

Dados do experimento de campo foram introduzidos no AgroDB, por meio da ferramenta denominada *SmartSim*, criada por Bavaresco [40]. O *SmartSim* disponibiliza, através da Web, a consulta e a inserção de dados de experimentos no banco de dados AgroDB. O cadastro de um novo experimento segue o formato passo a passo. Ao todo são onze passos, nos quais os dados são organizados por relação, sendo eles: dados principais, datas do experimento, cultivar do experimento, datas de semeadura, localidade, anos de experimento, resultados desejados, parâmetros de controle, condições iniciais do solo, fertilizantes e finalizar.

Os experimentos cadastrados no banco de dados AgroDB tornam-se acessíveis na Web, por meio de um serviço denominado *AgroService* [40]. URIs acessadas pela plataforma *Coupling* devolvem os dados dos experimentos, permitindo assim a configuração e montagem do ambiente de execução dos modelos acoplados.

Para a realização da calibração foi adotado um método empírico. Alteravam-se os valores dos coeficientes das variáveis, nos modelos de doenças; iniciava-se a execução dos modelos acoplados, via plataforma; e analisavam-se os resultados. A análise foi realizada através da comparação dos valores observados nos experimentos de campo em relação aos valores simulados pelo acoplamento dos modelos *Cropsim-Wheat* e Modelo Genérico de Doenças.

Com relação ao modelo de cultura, foram alterados os valores dos coeficientes genéticos das cultivares de trigo (Tabela 1), armazenadas no arquivo .CUL. Nos parâmetros de controle, descritos no arquivo .WHX, a propriedade DISES foi alterada de 'N' (Não) para 'Y' (Sim), para que o modelo considerasse a ocorrência de doenças.

4.4 RESULTADOS E DISCUSSÃO

Na integração entre os modelos, informações relativas a proporção de área foliar doente são produzidas diariamente e por folha. O modelo *Cropsim-Wheat* cria várias folhas durante o processo de desenvolvimento da planta. Na simulação acoplada as doenças competem pela área disponível (área sadia) de cada uma. Como resultado de cada cálculo diário o modelo de doenças devolve a área infectada. Na rotina de acoplamento o modelo de cultura é atualizado, sendo que esta área utilizada pela doença não estará mais disponível na folha no próximo dia. As saídas produzidas pela plataforma foram comparadas com as saídas geradas pelo modelo *Cropsim-Wheat* para comprovar a integração. O arquivo LEAVESDI.OUT, gerado pelo modelo de cultura, apresenta os detalhes relativos as áreas das folhas e foi utilizado na comparação. Ambas as saídas apresentaram o mesmo resultado, caracterizando a perfeita integração dessas variáveis entre os modelos acoplados.

A seguir são mostrados os resultados comparativos entre os experimentos observados no campo e os respectivos experimentos simulados através da plataforma *Coupling*. O comparativo de

Tabela 1. Coeficientes genéticos de cultivares de trigo usadas com o modelo *Cropsim-Wheat*

Cultivar / Coeficiente [1]	**BR-23**	**Embrapa-16**	**BRS-179**
P1V	0	0	0
P1V0	0	0	0
P1D	15	15	25
P2D	40	70	50
P1	480	480	680
P2	240	180	110
P3	160	160	100
P4	150	150	150
P5	140	140	140
P6	35	35	35
P7	175	175	175
P8	500	500	500
GNUMW	25	25	25
GWTX	35	35	34
SWTS	1,5	1,5	1,5
PHINT	90	90	90

[1] Coeficientes: P1V: Dias de temperatura do ar ótima para completar a vernalização; P1V0: Fator de vernalização quando germinada (0-1); P1D: Porcentagem de redução na taxa de desenvolvimento na fase um num fotoperíodo de 10 horas; P2D: Porcentagem de redução na taxa de desenvolvimento na fase dois num fotoperíodo de 10 horas; P1: Duração do final da fase juvenil para a fase de espigueta terminal; P2: Duração da fase de espigueta terminal para pseudocaule; P3: Duração da fase de pseudocaule ereto terminando com o crescimento da folha; P4: Duração da fase do crescimento da folha até o pico do crescimento da planta; P5: Duração do pico do crescimento da planta até a antese; P6: Duração da fase de antese; P7: Atraso nos grãos; P8: Duração da fase de enchimento de grãos; GNUMW: Número de grãos por peso de massa na antese; GWTX: Número máximo de grãos em condições ótimas (mg); SWTS: padrão, sem estresse peso seco (incluindo grão) da parte aérea madura no final do ciclo; PHINT: Intervalo de filocron (aparecimento de uma folha e uma sucessiva)

cada uma das três cultivares analisadas são apresentados para os anos de 2003 e 2004. Nos gráficos, a linha contínua preta representa as observações feitas em campo, enquanto que a linha tracejada representa os valores de severidade obtidos pela simulação. Também são demonstrados e avaliados o erro médio absoluto (*Mean Absolute Error* - MAE); o erro médio padrão (*Mean Standard Error* - MSE) e o erro médio quadrático (*Root Mean Square Error* - RMSE). Estes três índices estatísticos apresentam seu melhor resultado próximo de zero e são medidos na mesma unidade das variáveis testadas.

Na Figura 15 estão representados os comparativos para a Mancha Amarela. Cada coluna representa uma cultivar, enquanto que as linhas representam os anos de 2003 (linha L1) e de 2004 (linha L2). Os valores máximos de severidade, caracterizados pela área afetada da planta em cm^2, medidos para o ano de 2003 (L1), foram: 35,185 cm^2 para a cultivar BRS 179; 46,250 cm^2 para a cultivar Embrapa 16 e 35,417 cm^2 para a cultivar BR 23, como pode-se verificar nas colunas C1, C2 e C3, respectivamente. Em uma análise superficial sobre os dados observados em 2003, percebe-se que a cultivar de trigo Embrapa 16 foi mais suscetível a Mancha Amarela em comparação com as demais.

Os dados simulados para o ano de 2003 apresentaram um resultado aceitável em comparação com os dados observados. Percebe-se claramente um padrão na curva de evolução da doença.

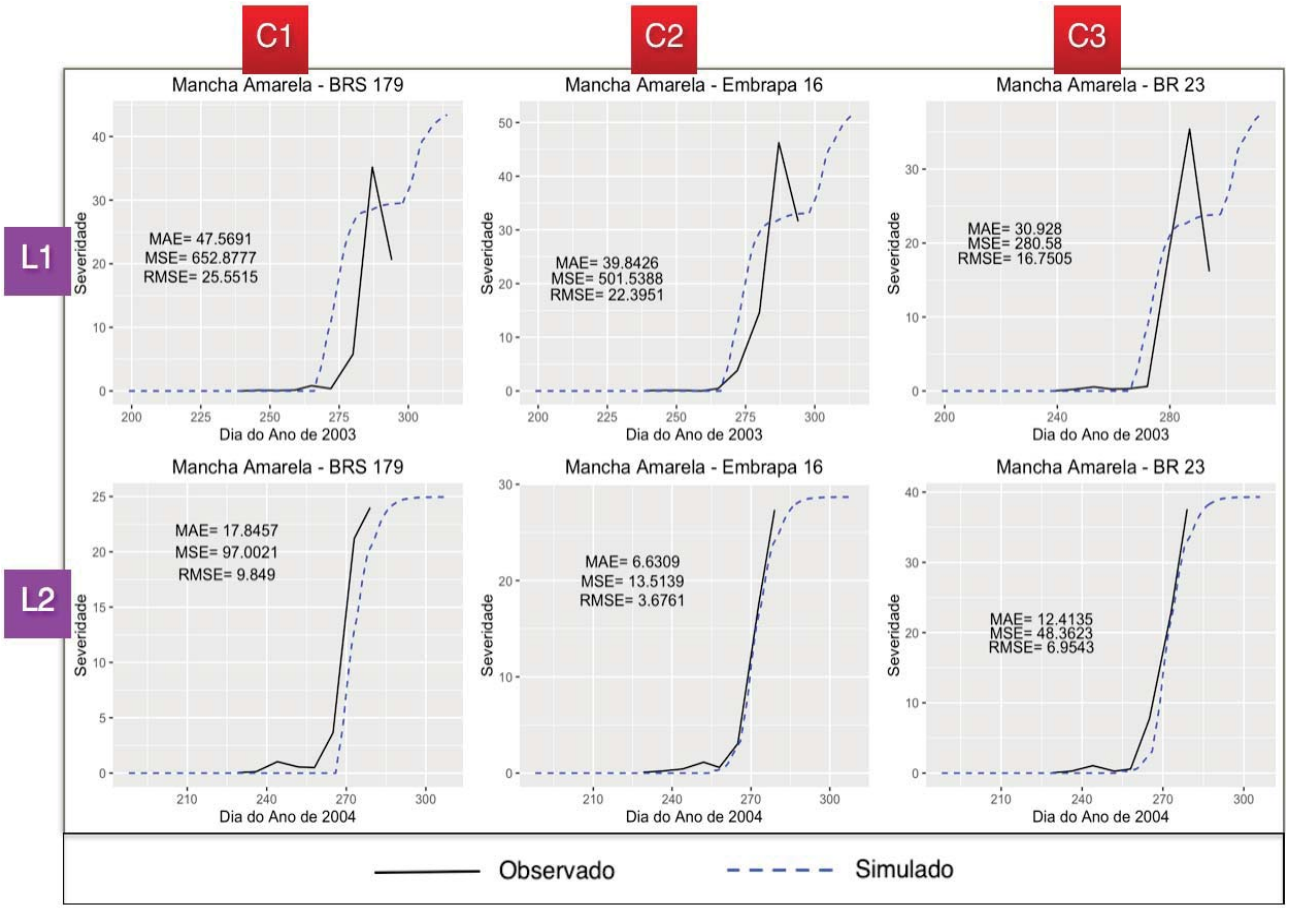

Figura 15. Representação gráfica dos resultados dos experimentos, simulados e observados, nos anos de 2003 (L1) e 2004 (L2), para as cultivares de trigo: BRS 179 (C1), Embrapa 16 (C2) e BR 23 (C3), com severidade por Mancha Amarela. A linha contínua na cor preta representa as observações feitas em campo, enquanto que a linha tracejada representa os valores de severidade obtidos pela simulação.

A área total comprometida da planta é quase a mesma que a medida em campo, e o pico de infecção se deu em dias muito próximos.

Devido a dificuldade para estimar os valores de algumas variáveis, utilizadas pelo modelo genérico de doenças, e, visando a obtenção de resultados compatíveis com os observados em campo, fez-se necessária uma calibração do modelo. As variáveis e valores alterados para cada cultivar e ano podem ser verificadas na Tabela 2. Pavan [10] define essas variáveis da seguinte forma:

- *Initial Inoculum*: Valor do inóculo inicial;
- *Acumulate Favorability*: Número de dias fisiológicos necessários para liberação dos primeiros esporos;
- *Host Factor*: Fator de resistência do hospedeiro; Informado na proporção 0-1.
- *Wetness Threshold*: Número mínimo de horas de molhamento necessário para criar novas lesões.

Tabela 2. Parametrização do Modelo da Mancha Amarela para as diferentes cultivares e anos do experimento

Mancha Amarela				
Ano	**Variável**	**Cultivar**		
		BR 179	**Embrapa 16**	**BR 23**
2003	Initial Inoculum	10	10	15
	Acumulate Favorability	18	18	18
	Host Factor	1	1	0.5
	Wetness Threshold	21	21	21
2004	Initial Inoculum	15	38	36
	Acumulate Favorability	25	25	25
	Host Factor	1	0.3	0.5
	Wetness Threshold	24	24	24

No ano de 2003 o PMF, obtido a partir de uma estimativa no número de horas com umidade relativa acima de 90%, acabou ficando abaixo do valor padrão definido para a variável *Wetness Threshold*. Por este motivo, não era possível determinar o processo de infecção diário, pois não havia favorabilidade para isso. Pavan [10] chama atenção para esta variável, alertando para a necessidade de se ter dados mais precisos e mais confiáveis para uma melhor calibração do modelo. Portanto, para que houvessem resultados para a simulação da Mancha Amarela no ano de 2003, o valor desta variável precisou ser diminuído (Tabela 2).

Para o ano de 2004 os dados observados e simulados, para a Mancha Amarela nas três cultivares, apresentaram praticamente a mesma área comprometida (Figura 15). Ao analisar os valores dos índices MAE, MSE e RMSE verifica-se que os resultados estão mais próximos de zero em comparação com os resultados obtidos para esses mesmos índices no ano de 2003. Os valores baixos obtidos para esses índices no ano de 2004 evidenciam o sucesso na execução dos modelos acoplados via plataforma *Coupling*.

Os comparativos para a Ferrugem da Folha do trigo são apresentados na Figura 16. Nas colunas C1, C2 e C3, são listados, respectivamente, os resultados obtidos para as cultivares BRS179, Embrapa 16 e BR23. As linhas L1 e L2 definem os anos dos experimentos, 2003 e 2004, respectivamente.

Percebe-se nos dados observados que a incidência da Ferrugem da Folha foi muito semelhante para o ano de 2003 nas cultivares BRS179 e Embrapa 16, enquanto que para o mesmo ano a cultivar BR 23, teve baixa incidência. Para o ano de 2004 as cultivares apresentam incidência bem diferentes umas das outras. O resultado dos dados simulados, reproduzidos pela linha tracejada, mostraram-se satisfatórios em relação aos dados observados. Tanto a curva de evolução da doença, quanto a área total final comprometida da planta, estão muito próximas ao que foi verificado no campo.

Da mesma forma que na doença da Mancha Amarela, as variáveis do modelo genérico de doenças tiveram de ser calibradas, utilizando-se para isso o método empírico. As variáveis e valores alterados para cada cultivar e ano podem ser verificadas na Tabela 3.

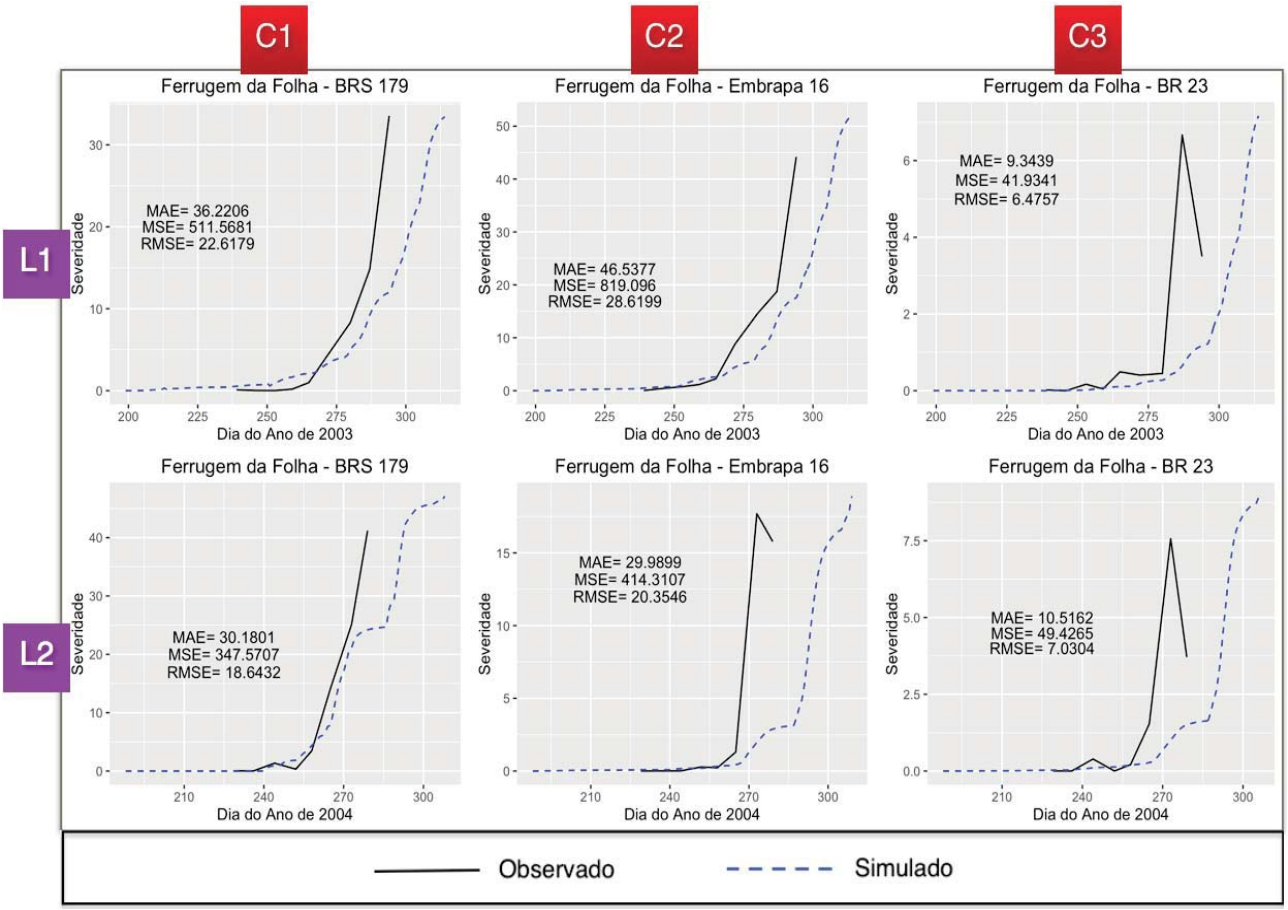

Figura 16. Representação gráfica dos resultados dos experimentos, simulados e observados, nos anos de 2003 (L1) e 2004 (L2), para as cultivares de trigo: BRS 179 (C1), Embrapa 16 (C2) e BR 23 (C3), com severidade para a Ferrugem da Folha do Trigo. A linha contínua na cor preta representa as observações feitas em campo, enquanto que a linha tracejada representa os valores de severidade obtidos pela simulação.

Tabela 3. Parametrização do Modelo da Ferrugem da Folha do Trigo para as diferentes cultivares e anos do experimento

Ferrugem da Folha do Trigo				
Ano	**Variável**	**Cultivar**		
		BR 179	**Embrapa 16**	**BR 23**
2003	Initial Inoculum	8	10	6
	Acumulate Favorability	5	5	5
	Host Factor	1	0.8	0.1
2004	Initial Inoculum	10	6	6.6
	Acumulate Favorability	55	5	14
	Host Factor	1	0.8	0.1

Os resultados obtidos por meio do acoplamentos dos modelos de doença, da Mancha Amarela e da Ferrugem da Folha do Trigo, ao modelo de cultura *Cropsim-Wheat*, permitiram avaliar a evolução das doenças, para diferentes cultivares, em diferentes anos, por meio da plataforma *Coupling*. A estrutura disponibilizada pela plataforma garantiu a integração dos modelos, desenvolvidos em diferentes linguagens de programação, bem como, simplificou os processos de configuração das

execuções. Em função do caráter adaptável dos modelos envolvidos na simulação, sustentado pela parametrização por meio de arquivos de entrada, foi possível a calibração de valores de parâmetros que variam amplamente de ano para ano e entre cultivares.

Cabe salientar que características específicas das doenças, utilizadas no modelo genérico para cálculos de favorabilidade e avanço das doenças sobre a área da planta, não foram consideradas nesta análise. Os arquivos de configuração para ambos os modelos foram obtidos de estudos, que validaram ou utilizaram o modelo genérico na simulação dessas doenças.

4.5 CONCLUSÃO

A plataforma de acoplamento desenvolvida é somente uma prova de conceito; a execução dos modelos acoplados em cenários experimentais não têm a intenção de tirar conclusões substantivas. No entanto, conseguiu-se comprovar que os modelos interagiram com a infra-estrutura de acoplamento corretamente, bem como, que os ponteiros se moveram entre os modelos acoplados adequadamente. Apresentou-se os resultados do sistema acoplado através de prova-de-conceito de dois cenários de exemplo. Ressalta-se que estes resultados são apenas para fins ilustrativos.

No entanto, foi possível comprovar que o modelo *CROPSIM-Wheat* ligado aos modelos de doenças produziram área infectada similar aos dados observados em ensaios de trigo em Passo Fundo, RS, Brasil. O comparativo comprovou que esses modelos estão se comunicando, por meio da plataforma *Coupling*, e que a planta está sendo afetada pelas doenças.

A rapidez e simplicidade no processo de configuração dos modelos na plataforma, sem dúvida, representam o diferencial para o usuário, que deseja simular os resultados de uma execução de modelos acoplados. Também, indica a possibilidade de utilização do conhecimento legado, proporcionado pela criação e melhoria de outros modelos.

5. CONSIDERAÇÕES FINAIS

O uso de *WebSockets* permitiu a integração de modelos de simulação via internet de forma rápida, segura e com sobrecarga mínima. A definição de um canal de comunicação *full-duplex*, sobre o qual mensagens podem ser enviadas em ambos os sentidos, oferece uma enorme redução no tráfego de rede e latência, em comparação com outras tecnologias aplicadas na implementação da comunicação em tempo real entre cliente e servidor. A comunicação é iniciada por meio de uma URL, e portanto, qualquer linguagem de programação que dê suporte ao HTTP poderá fazer uso desta tecnologia. Rotinas de acoplamento foram criadas para modelos desenvolvidos em Fortran, por meio de um conector C++, e Java, utilizando os recursos de comunicação em rede disponíveis em cada um dessas linguagens. *WebSockets* apresentou-se uma excelente tecnologia de comunicação em tempo real, sua arquitetura permitiu que diferentes modelos, em diferentes localizações, desenvolvidos em diferentes linguagens de programação, se conectassem e trocassem mensagens entre si rapidamente.

Com o desenvolvimento da plataforma *Coupling* foi possível realizar a integração do modelo de simulação *Cropsim-Wheat* a modelos que contabilizam estresses bióticos. Teoricamente, pela generalidade do projeto *Coupling* é possível realizar a integração com outros modelos de simulação, não alterando, ou adicionando poucas características estruturais a plataforma. No entanto, ainda faz-se necessário testar a integração com outros modelos de simulações de culturas.

O uso da plataforma *Coupling* permitiu a configuração de execuções de maneira simples. Utiliza-se de fontes de dados de experimentos e disponibiliza modelos pré configurados, desenvolvidos em diferentes linguagens de programação, na própria estrutura da plataforma. As saídas geradas pela execução acoplada ficam disponíveis na área restrita do usuário para um futuro acesso. Modelos conhecidos pelo usuário, e não incorporados a plataforma, podem ser acoplados com o uso de rotinas de acoplamento, que são fornecidas.

No estudo de caso uma prova de conceito comprovou a ligação entre um modelo de cultura e dois modelos de doenças, com linguagens de programação distintas. Os modelos acoplados interagiram com a infra-estrutura de acoplamento corretamente e os ponteiros moveram-se entre os modelos acoplados conforme o previsto. Foi necessária a calibração dos modelos envolvidos para que os resultados apresentassem valores mais próximos dos identificados em campo. A comparação foi em relação a severidade de duas doenças com incidência no Planalto Médio do Rio Grande do Sul. O acoplamento, no entanto, permite a avaliação de outras variáveis, como o comprometimento de uma doença no rendimento de grãos.

Como trabalhos futuros a exploração das saídas graficamente, bem como, a disponibilização de recursos de visualização e manipulação de dados em tela, seriam muito importantes para avaliar uma execução e para consolidar a plataforma como uma alternativa à execução de modelos acoplados. Apesar da possibilidade de acoplamento de modelos não conhecidos pela plataforma, a disponibilização de outros modelos na estrutura é outro recurso importante e necessário. Outra questão há se trabalhar, e não menos importante, diz respeito ao tempo de execução dos modelos acoplados.

Hoje uma execução por meio da plataforma, apesar do contexto em que está envolvida, de realizar o acoplamento via Web, necessita de cerca de 30 segundos para finalizar o processo completamente, enquanto que numa execução sem acoplamento este tempo não passa de 3 segundos. No entanto, esse tempo superior não compromete e nem diminui a plataforma em virtude das funcionalidades disponíveis e da arquitetura na qual esta inserida.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] FLOOD, J. The importance of plant health to food security. *Food Secur*, v. 2, p. 215–231, 2010.

[2] BEBBER, D. P.; RAMOTOWSKI, M. A. T.; GURR, S. J. Crop pests and pathogens move polewards in a warming world. *Nature Clim. Change*, Nature Publishing Group, v. 3, n. 11, p. 985–988, 11 2013. Disponível em: <http://dx.doi.org/10.1038/nclimate1990>. Acesso em: 18 Abr. 2017.

[3] FAO. *Food and Agriculture Organization of the United Nations*. 2016. Disponível em: <http://www.fao.org/>. Acesso em: 18 Abr. 2017.

[4] HOOGENBOOM, G. Contribution of agrometeorology to the simulation of crop production and its applications. *Agricultural and Forest Meteorology*, v. 103, n. 1-2, p. 137–157, 2000.

[5] GRAVES, A. et al. Crop simulation models as tools in computer laboratory and classroom-based education. *Journal of Natural Resources and Life Sciences Education*, v. 31, p. 48–54, 2002.

[6] FERNANDES, J. M. C. et al. Simulação de epidemias. *Revisão Anual de Patologia de Plantas*, RAP, Passo Fundo, Brasil, v. 2, n. 1, 1994.

[7] CHWIF, L.; MEDINA, A. C. *Modelagem e simulação de eventos discretos: teoria e aplicações*. 4. ed. Rio de Janeiro: Elsevier, 2015. 294 p.

[8] CHWIF, L. *Redução de Modelos de Simulação de Eventos Discretos na sua Concepção: uma Abordagem Causal*. Tese (Doutorado) — Escola Politécnica da Universidade de São Paulo, São Paulo, SP, Brasil, 1999.

[9] KELTON, D. W.; SADOWSKI, R. P.; SADOWSKI, D. *Simulation with Arena*. Boston: McGraw-Hill, 1998.

[10] PAVAN, W. *Técnicas de Engenharia de Software Aplicadas à Modelagem e Simulação de Doenças de Plantas*. Tese (Doutorado) — Universidade de Passo Fundo, Passo Fundo, RS, Brasil, 2007.

[11] JONES, J. W.; PORTER, C. H. Brief history of agricultural systems modeling. *AGSY*, Elsevier B.V., n. June, 2016. ISSN 0308-521X. Disponível em: <http://dx.doi.org/10.1016/j.agsy.2016.05.014>. Acesso em: 18 Abr. 2017.

[12] HOOGENBOOM, G. et al. *Decision Support System for Agrotechnology Transfer (DSSAT) Version 4.6*. 2015. Disponível em: <http://dssat.net>. Acesso em: 18 Abr. 2017.

[13] JONES, J. et al. *DSSAT Cropping System Model*. 2003. Disponível em: <http://dssat.net>. Acesso em: 18 Abr. 2017.

[14] DELPONTE, E. M.; FERNANDES, J. M. C.; PAVAN, W. A risk infection simulation model for fusarium head blight of wheat. *Fitopatologia Brasileira*, Brasilia, Brasil, v. 30, n. 6, p. 634–642, 2005.

[15] SILVA, F. C. da; BERGAMASCO, A. F.; VENDITE, L. L. *Modelos de simulação para análise e apoio à decisão em agrossistemas*. [S.l.: s.n.], 2002.

[16] HUNT, L.; PARARAJASINGHAM, S. Cropsim-wheat: A model describing the growth and development of wheat. *Canadian Journal of Plant Science*, v. 75, n. 3, p. 619–632, 1995.

[17] BATCHELOR, W. et al. Pest and disease damage module in dssat v.4.0 documentation and source code listing. *Agricultural and Biological Engineering Department*, University of Florida, Gainesville, Florida, US, n. 00-1205, 2000.

[18] OLIVEIRA, G. M. D. et al. Controle da ferrugem da folha do trigo (Puccinia triticina) em diferentes momentos de aplicação de fungicida Control of wheat leaf rust (Puccinia triticina) at different timings of fungicide application. p. 436–441, 2013.

[19] PONTE, E. M. D. et al. Giberela do Trigo – Aspectos Epidemiológicos e Modelos de Previsão. v. 29, n. 53, p. 587–605, 2004.

[20] PEDRINI, J. E. *Acoplando um Modelo de Doenças ao Modelo Cropgro-Soybean: Ferrugem Asiática da Soja*. 128 p. Dissertação (Mestrado) — Faculdade de Agronomia e Medicina Veterinária - Universidade de Passo Fundo, Passo Fundo, RS, Brasil, 2010.

[21] DELPONTE, E. M. et al. Models and applications for risk assessment and prediction of asian soybean rust epidemics. *Fitopatologia Brasileira*, Brasilia, Brasil, v. 31, p. 533–544, 2006.

[22] SOLANO, E.; MORRIS, R.; BOBASHEV, G. Coupling models by routing communication through a database. *RTI International*, n. September, p. 19, 2013. Disponível em: <http://www.rti.org/publications/rtipress.cfm?pubid=21470>. Acesso em: 15 Set. 2015.

[23] BULATEWICZ, T. *Support for model coupling: An interface-based approach*. Tese (Doutorado) — University of Oregon, Eugene, OR, 2006.

[24] KOO, J. *Modeling the impacts of climates variability on tomato disease management and production*. Tese (Doutorado) — University of Florida, Florida, US, 2002.

[25] RECH, G.; PAVAN, W.; HOLBIG, C. A. Modelo de simulador aplicado ao desenvolvimento de culturas. *Simpósio de Computação Aplicada*, Passo Fundo, RS, Brasil, 2010.

[26] PORTER, C. H.; BRAGA, R.; JONES, J. W. An approach for modular crop model development. *Agricultural and Biological Engineering Department*, Gainesville, Florida, US, n. 99-0701, 1999. Disponível em: <http://dssat.net/wp-content/uploads/2014/03/modular.pdf>. Acesso em: 26 Fev. 2017.

[27] PAVAN, W.; FERNANDES, J. Uso de orientação a objetos no desenvolvimento de modelos de simulação de doenças de plantas genéricos. *Revista Brasileira de Agroinformática*, São Paulo, SP, Brasil, v. 9, p. 12–27, 2009.

[28] TOEBE, J. *Um Modelo Baseado em Agentes para o Ciclo de Vida de Afídeos: aplicação na interação afídeo-planta-vírus*. Tese (Doutorado) — Universidade de Passo Fundo, Passo Fundo, RS, Brasil, 2014.

[29] BOOTE, K. et al. Coupling pests to crop growth simulators to predict yield reductions. *Phytopathology*, Gainesville, Florida, US, v. 73, p. 1581–1587, 1983.

[30] BATCHELOR, W.; JONES, J.; BOOTE, K. Extending the use of crop models to study pest damage. *Trans. Am. Soc. Agric. Eng*, p. 551–558, 1993.

[31] LIU, Q.; SUN, X. Research of Web Real-Time Communication Based on Web Socket. v. 2012, n. December, p. 797–801, 2012.

[32] BIN packing approximation algorithms: Combinatorial Analysis. 2013. Disponível em: <https://www.websocket.org>. Acesso em: 26 Fev. 2017.

[33] JUNIOR, A. R. S.; COSTA, F. M. Suporte de middleware auto adaptável para aplicações móveis e distribuídas. *SBPCNET*, 2015. Disponível em: <http://www.sbpcnet.org.br/livro/63ra/conpeex/pibic/trabalhos/ADALBERT.PDF>. Acesso em: 26 Fev. 2017.

[34] W3SCHOOLS. *JSON Data Types*. 2017. Disponível em: <http://www.w3schools.com/js/jsjsondatatypes.asp> Acesso em: 26 Fev. 2017.

[35] JSON.ORG. *Introdução ao JSON*. 2017. Disponível em: <http://www.json.org/json-pt.html>. Acesso em: 26 Fev. 2017.

[36] WHITE, J. W. et al. Integrated description of agricultural field experiments and production: The ICASA Version 2.0 data standards. *Computers and Electronics in Agriculture*, Elsevier B.V., v. 96, p. 1–12, 2013. ISSN 01681699. Disponível em: <http://dx.doi.org/10.1016/j.compag.2013.04.003>. Acesso em: 18 Abr. 2017.

[37] AGMIP. *AgMIP Data Interchange*. 2017. Disponível em: <https://data.agmip.org/>. Acesso em: 26 Fev. 2017.

[38] LAZZARETTI, A. *Integração de Banco de Dados e Modelos de Simulação de Culturas para Estimar o Impacto de Mudanças do Clima no Rendimento de Grãos e na Severidade da Giberela em Trigo*. Tese (Doutorado) — Universidade de Passo Fundo, Passo Fundo, RS, Brasil, 2013.

[39] VILLALOBOS, C. *Using JSON e Riak for AgMIP*. 2012. Disponível em: <http://research.agmip.org/ download/attachments/1212592/usingjsonandriak.pdf>. Acesso em: 26 Fev. 2017.

[40] BAVARESCO, J. *Um Sistema Integrado de Modelagem de Cultivos e Doenças: Brusone no Trigo*. Dissertação (Mestrado) — Universidade de Passo Fundo, Passo Fundo, RS, Brasil, 2015.

[41] DEVELOPMENT, A. S. *QuadUI*. 2014. Disponível em: <http://research.agmip.org/display/dev/QuadUI>. Acesso em: 26 Fev. 2017.

[42] BAIRD, D. *A short and sweet WebSocket client for C++*. 2016. Disponível em: <https://github.com/dhbaird/easywsclient>. Acesso em: 26 Fev. 2017.

[43] FUNCK, G.; FERNANDES, J. M.; PIEROBOM, C. Doenças foliares , área verde sadia e peso de grãos em diferentes cultivares de trigo Leaf diseases , healthy green area and kernel weight in different wheat cultivars. *Revista Brasileira de Ciências Agrárias*, Recife, PE, Brasil, v. 4, p. 3–10, 2009.

[44] FERNANDES, J. M. et al. Modelling fusarium head blight in wheat under climate change using linked process-based models. *International Symposium on Fusarium Head Blight*, Orlando, 2004.

[45] LAZZARETTI, A. T.; FERNANDES, J. M.; PAVAN, W. Calibração do cropsim-wheat para simulação do desenvolvimento e rendimento de grão de trigo no Sul do Brasil Material e Métodos. *Revista Brasileira de Ciências Agrárias*, Recife, PE, Brasil, v. 10, n. 3, p. 356–364, 2015.

[46] BERGER, R.; JONES, J. A general model for disease progress with functions for variable latency and lesion expansion on growing host plants. *Phytopathology*, Gainesville, Florida, US, p. 792–797, 1985.

[47] BERGER, R. et al. A simulation model to describe epidemics of rust of phaseolus beans i. development of the model and sensitivity analysis. *Phytopathology*, Gainesville, Florida, US, v. 85, n. 6, p. 715–721, 1995.

[48] RODRIGUES, R. D. Á. et al. Utilization of the cropgro-soybean model to estimate yield loss caused by Asian rust in cultivars. *Agrometeorology*, v. 71, n. 2, p. 308–317, 2012.

[49] GLEASON, M. L. et al. Development and validation of an empirical model to estimate the duration of dew periods. *The American Phytopathological Society*, v. 78, p. 1011–1016, 1994.

[50] RODRIGUES, R. D. Á. et al. Asian soybean rust: modeling the impact on soybean grain yield in the triângulo mineiro / Alto Paranaíba region, Minas Gerais, Brazil. v. 29, n. 2, p. 264–279, 2013.