Arquitetura de software para integração de
informações geográficas de sensores e voluntários
na gestão de risco de inundação

Raul Eduardo Simoni Castanhari

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Raul Eduardo Simoni Castanhari

Arquitetura de software para integração de informações geográficas de sensores e voluntários na gestão de risco de inundação

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. João Porto de Albuquerque Pereira

USP – São Carlos
Junho de 2016

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

C634a Castanhari, Raul Eduardo Simoni
 Arquitetura de software para integração de
informações geográficas de sensores e voluntários
na gestão de risco de inundação / Raul Eduardo
Simoni Castanhari; orientador João Porto de
Albuquerque Pereira. -- São Carlos -- SP, 2016.
 133 p.

 Dissertação (Mestrado - Programa de Pós-Graduação
em Ciências de Computação e Matemática Computacional)
-- Instituto de Ciências Matemáticas e de Computação,
Universidade de São Paulo, 2016.

 1. Arquitetura orientada a serviços. 2. Integração.
3. Dados geoespaciais. 4. informação geográfica de
voluntários. 5. gestão de inundação. I. Pereira, João
Porto de Albuquerque, orient. II. Título.

Raul Eduardo Simoni Castanhari

**Software architecture for integration of sensor and
volunteered geographic information in the flood risk
management**

Master dissertation submitted to the Instituto de
Ciências Matemáticas e de Computação – ICMC-USP,
in partial fulfillment of the requirements for the degree
of the Master Program in Computer Science and
Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and
Computational Mathematics

Advisor: Prof. Dr. João Porto de Albuquerque Pereira

**USP – São Carlos
June 2016**

Mãe, Pierina Adélia Simoni, pai, Jair do Carmo Castanhari, irmão, Francisco Rafael Simoni Castanhari, dedico esse trabalho a vocês como uma das formas de agradecimento e retribuição por toda ajuda que prestaram a mim. Orientador, João Porto de Albuquerque Pereira, dedico também esse trabalho a você como contribuição a um de seus projetos. Fazer esse trabalho só foi possível graças a vocês, e ele tem um pouco de cada um de vocês.

AGRADECIMENTOS

Obrigado, mãe, Pierina Adélia Simoni, por toda ajuda e apoio, por sempre estar comigo e sempre tentar me animar nos momentos difíceis. Obrigado, pai, Jair do Carmo Castanhari, por sempre ter apoiado minhas decisões e sempre ter acreditado em mim. Obrigado, irmão, Francisco Rafael Simoni Castanhari, por sempre ter me feito companhia e sempre ter me ajudado nos momentos de tristeza. Obrigado, tios, José Antônio Rodrigues e Flauzina Simoni Rodrigues, por toda ajuda prestada que facilitaram imensamente minha vida. Obrigado, primo, Thiago Simoni, por me mostrado pontos de vista libertadores e compreender meus lamentos. Obrigado, amigos, em especial Luiz Fernando Ferreira Gomes de Assis, Flavio Eduardo Aoki Horita, Livia Castro Degrossi, Roberto dos Santos Rocha, Daniel Soares Santos, Alinne Cristinne Corrêa dos Santos, Francisco Carlos Monteiro Souza, Melanie Eckle, Svend-Jonas Schelhorn, Luisa Kirschbaum e Benjamin Herfort, por toda ajuda, dedicação, paciência, carinho e acolhimento que tiveram por mim. Obrigado, Liliane Maria Gomes Torres, Neucy Ery Akagi Jordão e Gastão Luis Schmitt Jordão, por se importarem comigo e com meu trabalho, e por me proporcionarem paz, alegria, calma, tranquilidade e uma visão de comportamentos, culturas, costumes e ideias muito mais abrangentes do que as que eu tinha. Obrigado, orientador e amigo, João Porto de Albuquerque Pereira, por todos os ensinamentos, paciência e compreensão que me ajudaram a ser melhor. Obrigado, família, amigos e todos que estiveram comigo, pela ajuda, companhia e ânimo. Devo esse trabalho a todos vocês e sou imensamente grato por tudo.

Muito obrigado!

*Não busque um estado permanente de felicidade
pois ele não existe. O que existe são momentos
felizes, curtos mas diversos. Reconheça-os e
lembre-se deles para ter uma vida mais feliz.*

RESUMO

CASTANHARI, R. E. S.. **Arquitetura de software para integração de informações geográficas de sensores e voluntários na gestão de risco de inundação**. 2016. 133 f. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos – SP.

Desastres naturais são fenômenos que causam grandes danos às pessoas em áreas urbanas e rurais, motivando medidas de prevenção e reação. Se múltiplas fontes de informação são consideradas, tais medidas podem ser mais assertivas e efetivas. Porém, a integração de dados heterogêneos ainda impõe desafios devido às diferenças em suas estruturas. Para preencher essa lacuna, esta pesquisa apresenta uma arquitetura orientada a serviços, como parte de uma plataforma geoespacial para gestão de risco de desastres, a qual visa apoiar a integração de dados de sensores e de voluntários relacionados a inundações. Como resultado, a composição dos serviços descritos pelos componentes arquiteturais possibilita a integração entre dados de sensores e voluntários usando algoritmos diferentes de forma flexível e automática.

Palavras-chave: Arquitetura orientada a serviços, Integração, Dados geoespaciais, informação geográfica de voluntários, gestão de inundação.

ABSTRACT

CASTANHARI, R. E. S.. **Arquitetura de software para integração de informações geográficas de sensores e voluntários na gestão de risco de inundação**. 2016. [133](#) f. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos – SP.

Natural disasters are phenomena that can cause great damage to people in urban and rural areas, thus motivating preventive and reactive measures. If multiple sources of information are considered, such measures can be more assertive and effective. However, the integration of heterogeneous data still poses challenges due to the differences in their structures. To fill this gap, this paper presents a service-oriented architecture, as part of a geospatial platform for disaster risk management, which aims to support the integration of sensor data streams and Volunteered Geographic Information (VGI) related to floods. The architectural components are implemented as Web services, and their composition enables the integration between sensor data and VGI using several algorithms in a flexible and automated manner. As result, the composition of the services described by the architectural components allows the integration of sensor and volunteered data by using diverse algorithms in a flexible and automated way.

Key-words: Service oriented architecture, Integration, Geospatial data, Volunteered geographic data, flood management.

LISTA DE ILUSTRAÇÕES

Figura 1 – Contexto de pesquisa	27
Figura 2 – Arquitetura orientada a serviços (SOMMERVILLE, 2011).	31
Figura 3 – Exemplo de árvore de utilidade de atributos de qualidade (adaptada de Clements, Kazman e Klein (2002)).	37
Figura 4 – Ciclo de desastre (adaptada de Poser e Dransch (2010)).	42
Figura 5 – Processo de projeto de arquitetura em três etapas (adaptada de Gorton (2011)).	54
Figura 6 – Diagrama de casos de uso da AGORA-IFM	57
Figura 7 – Processo de integração definido para a AGORA-IFM.	58
Figura 8 – Diagrama de componentes da arquitetura da AGORA-IFM.	63
Figura 9 – Funcionamento lógico da arquitetura.	67
Figura 10 – Árvore de utilidade de atributos de qualidade da AGORA-IFM.	75
Figura 11 – Pontos utilizados na categorização de observações de sensores (Zaffani e Mendiondo (2012)).	85
Figura 12 – Cenário de referência.	87
Figura 13 – Resultado da integração.	90
Figura 14 – Resultado da integração dicotomizado.	91

LISTA DE QUADROS

Quadro 1 – Exemplo de ficha de análise de requisito (adaptado de Clements, Kazman e Klein (2002)).	39
Quadro 2 – Resumo das etapas do ATAM (adaptado de Clements, Kazman e Klein (2002)).	41
Quadro 3 – Lista de requisitos arquiteturais	55
Quadro 4 – Lista de decisões arquiteturais	74
Quadro 5 – Lista de cenários arquiteturais	78
Quadro 6 – Resumo dos resultados	81
Quadro 7 – Ficha de análise do requisito R2	101
Quadro 8 – Ficha de análise do requisito R3	102
Quadro 9 – Ficha de análise do requisito R4	102
Quadro 10 – Ficha de análise do requisito R6	103
Quadro 11 – Ficha de análise do requisito R7	103
Quadro 12 – Ficha de análise do requisito R8	104
Quadro 13 – Ficha de análise do requisito R10	104
Quadro 14 – Ficha de análise do requisito R11	105

LISTA DE CÓDIGOS-FONTE

Código-fonte 1 – Exemplo da operação de requisição <i>getObservation</i> de acordo com padrão O&M	44
Código-fonte 2 – Exemplo de dados geográficos de acordo com o padrão <i>SWE Common Data Model</i>	45

LISTA DE TABELAS

Tabela 1 – Dados simulados de sensores	87
Tabela 2 – Dados simulados categorizados	88

SUMÁRIO

1	INTRODUÇÃO	25
1.1	Motivação	25
1.2	Contexto de Pesquisa	26
1.3	Objetivos	28
1.4	Organização da Dissertação	28
2	FUNDAMENTAÇÃO TEÓRICA E TRABALHOS RELACIONADOS	29
2.1	Arquitetura de Sistemas de Software	29
2.1.1	<i>Arquitetura Orientada a Serviços</i>	31
2.1.2	<i>Avaliação de Arquiteturas de Software</i>	32
2.2	Gestão de Desastres Naturais	41
2.3	Informações Geográficas de Sensores	42
2.4	Informações Geográficas de Voluntários	46
2.5	Integração de Informações Geográficas	47
2.6	Trabalhos Relacionados	48
2.7	Considerações Finais	52
3	AGORA INFORMATION FUSION AND MANAGEMENT	53
3.1	Considerações Iniciais	53
3.2	Metodologia	53
3.3	Requisitos Arquiteturais	55
3.4	Projeto da Arquitetura	56
3.4.1	<i>Visão de Caso de Uso</i>	56
3.4.2	<i>Visão de Processo</i>	57
3.4.3	<i>Visão de Desenvolvimento</i>	62
3.4.4	<i>Visão Lógica</i>	66
3.5	Validação	69
3.6	Considerações Finais	69
4	AVALIAÇÃO ARQUITETURAL	71
4.1	Condução	71
4.2	Resultados	73
4.3	Discussão	77

5	PROVA DE CONCEITO	83
5.1	Implementação	83
5.2	Experimento	85
5.3	Resultados	89
5.4	Discussão	89
6	CONCLUSÃO	93
6.1	Trabalhos Futuros	94
REFERÊNCIAS		95
APÊNDICE A	FICHAS DE AVALIAÇÃO ARQUITETURAL	101
APÊNDICE B	A SOFTWARE ARCHITECTURE TO INTEGRATE SENSOR AND VOLUNTEERED GEOGRAPHIC IN- FORMATION FOR FLOOD RISK MANAGEMENT .	107
APÊNDICE C	A GAMIFICATION-BASED SOCIAL COLLABORA- TIVE ARCHITECTURE TO INCREASE RESILIENCE AGAINST NATURAL DISASTERS	121

INTRODUÇÃO

Neste capítulo são apresentados a motivação para a realização deste trabalho, o contexto em que ele está inserido, seus objetivos e a metodologia adotada para alcançar esses objetivos.

1.1 Motivação

Desastres naturais, como inundações, tempestades, incêndios e furacões, podem causar grandes danos às pessoas quando ocorrem em locais de atividade humana, provocando perdas de alimentos, recursos naturais, infraestrutura urbana e, mais importante, vidas humanas. Em 2014 desastres naturais geraram perdas financeiras próximas a US\$110 bilhões em todo o mundo e causaram 7700 fatalidades ([MUNICHRE, 2014](#)). Considerando o tipo de desastres naturais ocorridos em 2014, 73% foram causados por eventos extremos com origem hidrometeorológica, isto é, inundações, tempestades, avalanches e deslizamentos de terra causados por chuvas ([MUNICHRE, 2014](#)).

Esses desastres têm aumentado a preocupação em se estar preparado e minimizar as consequências dos próximos eventos, o que estimula a construção de comunidades resilientes, principalmente daquelas localizadas em áreas de risco, como morros ou próximas de rios. Nesse sentido, resiliência é a capacidade da comunidade de resistir a eventos extremos ([MEDIONDO, 2010](#)), e uma alternativa para promovê-la é por meio da aplicação de sistemas de monitoramento ambiental ([MANSOURIAN *et al.*, 2006](#)). Outra alternativa está relacionada à obtenção de informações precisas, contínuas e de forma oportuna, o que pode ajudar muito as agências envolvidas na gestão de risco de inundações ([HORITA *et al.*, 2015](#)). Porém, o monitoramento contínuo desse tipo de risco requer informações locais, como dados de chuva e observações do nível dos rios, para avaliar a possibilidade de potenciais inundações ([DEGROSSI *et al.*, 2014](#)). Esse monitoramento pode não compreender todas as áreas necessárias, principalmente em países como o Brasil devido à sua

vasta extensão territorial.

Nesse contexto, informação geográfica de voluntários, do inglês *Volunteered Geographic Information* (VGI), tem sido de grande valor prático para complementar bancos de dados geoespaciais existentes (GOODCHILD, 2007) devido ao grande número de potenciais voluntários que agem como “sensores” (POSER; DRANSCH, 2010). Há vários desafios de pesquisa em aberto para tornar VGI uma fonte de dados efetiva para gestão de risco de inundações. Um deles é como integrar VGI a bancos de dados existentes ao invés de considerá-la como informações paralelas (ALBUQUERQUE *et al.*, 2015). Em particular, pesquisas sobre como integrar VGI a dados de sensores ainda enfrentam desafios porque os dados de sensores são padronizados e os de voluntários raramente o são, dados de sensores são precisos e os de voluntários dependem de sua percepção, e os dados de voluntários são geralmente menos confiáveis que os de sensores. Além disso, VGI normalmente requer um processamento manual, por exemplo, a observação de um vídeo por uma pessoa para extrair alguma informação. Isso limita os contextos de uso de VGI porque esse tipo de tarefa tem objetivos muito específicos e torna inflexível o processo que a emprega. Ainda, atividades manuais consomem muito tempo, o que pode ser crítico no contexto de gestão de desastres onde agências de emergência precisam agir rapidamente.

Por outro lado, integrar esses dados é importante porque o uso de várias fontes de dados pode prover um cenário de desastre mais completo em relação ao uso de apenas uma fonte (HORITA *et al.*, 2015; GOODCHILD, 2007). Aumentar a concentração espacial e temporal de informações pode auxiliar a comunidade a se conscientizar dos acidentes potenciais e em andamento e contribuir para uma gestão de desastres naturais mais assertiva e eficiente. Nesse sentido, este trabalho propõe uma arquitetura de software para integrar dados de sensores e de voluntários no contexto de gestão do risco de inundações.

1.2 Contexto de Pesquisa

O projeto de pesquisa chamado AGORA – *A Geospatial Open Collaborative Approach for Building Resilience against Disasters and Extreme Events* – visa criar uma plataforma para juntar as perspectivas da ciência, população geral e governo para abordar desafios sociais (ALBUQUERQUE; ZIPF, 2012), com o foco inicial em inundações. Inundações são o desastre natural que causa maior dano no Brasil, com aproximadamente 48 milhões de pessoas afetadas entre 1980 e 2010 (UNISDR, 2014).

O projeto AGORA é composto por três partes: aquisição, responsável por obter e armazenar dados geoespaciais de sensores e voluntários; integração, cujo objetivo é integrar dados dos dois tipos anteriores; e aplicação, responsável por gerar indicadores para facilitar o processo de tomada de decisão e informar a comunidade sobre possíveis

problemas ambientais. Como destacado na Figura 1, essa pesquisa encontra-se na parte de integração do projeto AGORA.

Figura 1 – Contexto de pesquisa

A parte de aquisição de dados da AGORA compreende duas aplicações que não estão inclusas nesta pesquisa. Uma delas é a *Dynamic Sensor Management* (AGORA-DSM), uma aplicação que gerencia dinamicamente geosensores heterogêneos, sendo responsável por coletar, armazenar e disponibilizar os dados desses sensores (ASSIS *et al.*, 2016). Os dados de voluntários são gerenciados pela *Volunteered Observation Service* (AGORA-VOS), uma aplicação acessível via *website* ou aplicativos de celular e gerenciado por uma instância personalizada da plataforma de *crowdsourcing* Ushahidi¹ (DEGROSSI *et al.*, 2014). Dados de voluntários, em um contexto geral, pode ser textos, fotos, vídeos ou áudios não padronizados, apresentando assim diferentes estruturas de dados das quais é difícil extrair informação relevante, o que muitas vezes é feito manualmente. A interface dessa aplicação guia o voluntário em como relatar suas informações, estruturando-as, apesar de também armazenar outros dados não estruturados.

A parte de integração da AGORA está inclusa nesta pesquisa e compreende a aplicação *Information Fusion and Management* (AGORA-IFM), que utiliza as duas aplicações anteriores e produz resultados com dados de sensores e voluntários integrados. Esses dados são utilizados pela parte de aplicação da AGORA, que compreende a aplicação *Decision Support* (AGORA-DS) (HORITA *et al.*, 2015) e que também não está inclusa nesta pesquisa.

O trabalho desenvolvido por Horita *et al.* (2015) também utiliza a AGORA-DSM e a AGORA-VOS para juntar dados de sensores e de voluntários relacionados a inundações de modo a uma fonte complementar as informações da outra. Esses dados são apresentados em um único painel para auxiliar na tomada de decisão por parte de autoridades responsáveis e na conscientização da população quanto à ocorrência de alagamentos. Nesse trabalho, o qual não considera a AGORA-IFM, os dados de sensores e de voluntário são graficamente justapostos na AGORA-DS, porém, eles ainda não são integrados de forma

¹ <https://www.ushahidi.com/> Acessado em 31/01/2016.

que se possa processá-los igualmente para se obter novas informações. Por exemplo, a aplicação de um algoritmo geoespacial sobre os dados dessas duas fontes, sem precisar diferenciá-los, poderia integrá-los gerando novas informações relevantes à gestão de desastres naturais. Este projeto insere-se nesse contexto propondo uma arquitetura de software para a AGORA-IFM a fim de viabilizar a integração desses dados.

1.3 Objetivos

Esta pesquisa aborda as questões levantadas propondo uma arquitetura orientada a serviços com o objetivo geral de apoiar a integração de dados de sensores e de voluntários. Para tanto, essa arquitetura foi desenvolvida com os três seguintes objetivos específicos:

- definir um processo de integração automatizado;
- possibilitar a execução de algoritmos espaciais sobre dados de sensores e voluntários de modo que seja fácil trocar um algoritmo por outro; e
- estabelecer a arquitetura que suporte um processo de integração com essas características de tal forma a facilitar sua compreensão, comunicação e implementação.

O desenvolvimento desta arquitetura contou com a identificação dos requisitos arquiteturais envolvidos no contexto apresentado, a elaboração de componentes e suas conexões ilustrados em diagramas pertinentes e uma avaliação a fim de verificar se tal arquitetura é adequada ao seu propósito. Além disso, uma implementação dessa arquitetura foi feita como prova de conceito para se executar o processo de integração proposto.

1.4 Organização da Dissertação

Este capítulo apresentou a motivação para este trabalho, o contexto em que está inserido, os objetivos de pesquisa e a metodologia adotada. O restante desta dissertação está organizado da seguinte maneira: o [Capítulo 2](#) apresenta os conceitos envolvidos neste trabalho, principalmente sobre arquitetura de software e informações geográficas; o [Capítulo 3](#) apresenta a arquitetura proposta por esta pesquisa, a AGORA Information Fusion and Management; o [Capítulo 4](#) apresenta a avaliação da arquitetura proposta, em que foi empregada uma adaptação do *Architecture Trade-off Analysis Method*; o [Capítulo 5](#) apresenta uma prova de conceito da AGORA-IFM, envolvendo sua implementação e um experimento; por fim, o [Capítulo 6](#) apresenta as conclusões obtidas com esta pesquisa e propostas de trabalhos futuros.

FUNDAMENTAÇÃO TEÓRICA E TRABALHOS RELACIONADOS

Neste capítulo são apresentados os conceitos relacionados a essa pesquisa: arquitetura de sistemas de software, mais especificamente arquitetura orientada a serviços e avaliação de arquiteturas, gestão de desastres naturais, dados de sensores, dados de voluntários e integração de dados geográficos.

2.1 Arquitetura de Sistemas de Software

Com o aumento do volume de dados e da complexidade dos processamentos de sistemas de software, o desenvolvimento desse tipo de sistema sem uma estruturação planejada e/ou adequada ao mesmo pode produzir “gargalos”, componentes muito complexos ou que agrupam funcionalidades não coesas, podendo dificultar imensamente sua manutenção e evolução, além de torná-lo ineficiente. A elaboração de arquiteturas para sistemas de software insere-se nesse contexto provendo uma organização para esses sistemas com o objetivo de solucionar tais problemas e de esclarecer seu funcionamento.

Basicamente, uma arquitetura para sistemas de software estabelece a estrutura global do sistema, dividindo-o em componentes e definindo a conexão entre esses componentes (GARLAN, 2000; SEI, 2015). Cada arquitetura é moldada de uma forma particular para o sistema para o qual servirá de base, ou seja, a divisão em componentes e a forma como eles se comunicarão é voltada para as especificidades do sistema.

Para o esclarecimento do funcionamento dos sistemas, as arquiteturas podem ser apresentadas em diferentes visões. Muitas visões arquiteturais já foram propostas, dentre elas, podemos citar visão lógica, de processos, de implantação, de dados, e de casos de uso (KRUCHTEN, 1995). A visão lógica representa a organização conceitual do software em termos das camadas, subsistemas, pacotes, *frameworks*, classes e interfaces mais im-

portantes, resumindo também a funcionalidade dos principais elementos de software. A visão de processo representa os processos e linhas de execução da aplicação apresentando suas responsabilidades, colaborações e a alocação de elementos lógicos a elas. A visão de implantação representa a fase de implantação física de processos e componentes nos nós de processamento e a configuração física da rede entre os nós. A visão de dados mostra uma visão geral do fluxo de dados na aplicação, podendo fornecer o mecanismo de mapeamento dos objetos para um banco de dados, procedimentos e gatilhos armazenados do banco de dados, entre outros. Por fim, a descrição da arquitetura é organizada englobando essas quatro visões e, então, ilustrada utilizando-se alguns casos de uso ou cenários selecionados que constituem a quinta visão do modelo (GUESSI; NAKAGAWA; MALDONADO, 2010). Entretanto, não há um consenso sobre qual visão é mais adequada para determinada arquitetura. Contanto que ela seja expressiva, significativa e de fácil entendimento, qualquer visão pode ser utilizada.

No desenvolvimento de um sistema, pode-se fazer a análise de uma determinada arquitetura para se verificar sua qualidade. Já é sabido que a qualidade do sistema como um todo depende, e muito, da arquitetura empregada, e, dessa maneira, ela deve ser muito bem elaborada durante o desenvolvimento do sistema (BABAR; ZHU; JEFFERY, 2004; KAZMAN; KLEIN; CLEMENTS, 2000). Alguns métodos de avaliação arquitetural têm sido propostos (BABAR; ZHU; JEFFERY, 2004), como por exemplo o *Scenario based Architecture Analysis Method* (SAAM) (KAZMAN *et al.*, 1994) e o *Architecture Trade-off Analysis Method* (ATAM) (KAZMAN *et al.*, 1998), porém, ainda não se sabe ao certo quais critérios um método de avaliação de arquiteturas de software deve seguir e nem quais métodos são mais indicados para uma determinada arquitetura. Contudo, um “bom” método deve ser capaz de mostrar se certa arquitetura promove ou inibe as características do sistema que a utilizará como base para seu desenvolvimento, fazendo com que este tenha ou não maior eficiência, usabilidade, manutenibilidade e reusabilidade, por exemplo. Um software que siga um padrão arquitetural pode ser beneficiado ou não por ele, dependendo se o software segue as características nas quais o padrão foi baseado.

Considerando sistemas que atuam de maneira semelhante, as arquiteturas voltadas para eles também são semelhantes, o que nos permite aproveitar o projeto arquitetural de um deles para os demais, adaptando-se apenas o que houver de diferente entre eles e o sistema em questão. Desse modo, no desenvolvimento de um sistema com aspectos parecidos com outros sistemas já conhecidos, pode-se utilizar a arquitetura que é reconhecidamente eficiente para esses sistemas. Nesse sentido, vários padrões arquiteturais têm sido propostos para sistemas com diferentes objetivos (BUSCHMANN *et al.*, 1996), como padrões para sistemas distribuídos, adaptativos, interativos e outros. Um padrão bastante utilizado para tornar sistemas de software mais flexíveis e portáteis é o de arquitetura orientada a serviços, descrito detalhadamente na subseção a seguir.

2.1.1 Arquitetura Orientada a Serviços

Com a evolução dos sistemas de software e a demonstração de que eles são capazes de realizar, cada vez mais se exige que tais sistemas sejam tão adaptados quanto possível ao contexto o qual pretendem representar. Isso gerou o conceito de se desenvolver softwares que, mesmo após finalizados, permitem ser personalizados de acordo com as necessidades dos usuários, que podem estar relacionadas, por exemplo, aos recursos do sistema, à sua disponibilidade e usabilidade. Quando o sistema em questão pretende representar aspectos do mundo real com fidelidade, essa capacidade de se adaptar a mudanças torna-se indispensável. Esses fatos fizeram com que se buscassem tecnologias e abordagens alternativas que promovessem essas características nos sistemas computacionais (SOMMERVILLE, 2011).

A arquitetura orientada a serviços, mais conhecida como SOA (*Service Oriented Architecture*), apresenta-se como uma possível solução para essas questões. Ela é uma estruturação de sistemas em que as funcionalidades são elaboradas como serviços ou como uma composição de serviços. Nesse sentido, um serviço é um componente de software que possui seu próprio contexto funcional distinto e um conjunto de capacidades relacionadas a esse contexto, que, tipicamente, encapsulam um conceito de negócio (KRAFZIG; BANKE; SLAMA, 2004; ERL, 2009). Serviços representam um tipo de relação baseada em interações (atividades) entre, pelo menos, um provedor e um consumidor a fim de alcançar um determinado objetivo (ZHANG; ZHANG; CAI, 2007).

O funcionamento básico da arquitetura orientada a serviços é representado na Figura 2. Os provedores de serviço projetam e implementam os serviços e especificam sua interface. Eles também publicam informações sobre esses serviços em um registro acessível. Os solicitantes (clientes) que desejam utilizar um serviço encontram o contrato desse serviço e localizam seu provedor por meio do registro. Em seguida, eles podem comunicar-se com esse provedor por meio do *Simple Object Access Protocol* (SOAP), e vincular sua aplicação a esse serviço específico (SOMMERVILLE, 2011).

Figura 2 – Arquitetura orientada a serviços (SOMMERVILLE, 2011).

A linguagem de definição de serviço *web*, do inglês *Web Service Definition Language* (WSDL), é utilizada para descrever a interface de um serviço. Ela define como as operações (nomes, parâmetros e seus tipos) e associações de serviços devem ser realizadas (SOMMERVILLE, 2011). O padrão *Universal Description, Discovery and Integration* (UDDI) é utilizado para definir os componentes do contrato de um serviço para que uma aplicação possa descobri-lo. Esses componentes incluem informações sobre o provedor do serviço, os serviços fornecidos, o local da descrição de sua interface e informações sobre os relacionamentos de negócios.

Os sistemas estruturados segundo a arquitetura orientada a serviços são, principalmente, aqueles cujo acesso e funcionalidades são obtidos através da Internet. Os serviços disponibilizados e utilizados por esses sistemas permitem o envio e o recebimento de dados no formato XML, possibilitando a comunicação entre sistemas que utilizam tecnologias distintas, já que esse formato é independente de plataformas.

2.1.2 Avaliação de Arquiteturas de Software

Uma avaliação arquitetural é basicamente uma avaliação qualitativa, e seu principal objetivo é gerar informações capazes de responder se a arquitetura analisada é adequada ao sistema para o qual foi projetada. Nesse sentido, uma arquitetura é considerada adequada se atender a dois critérios: (CLEMENTS; KAZMAN; KLEIN, 2002)

- O sistema que resulta da arquitetura satisfaz seus requisitos de qualidade, por exemplo, se sua velocidade atende o desempenho esperado, se a segurança oferecida segue às restrições necessárias, se suas interfaces possibilitam a interoperabilidade planejada, etc.
- O sistema pode ser construído usando os recursos disponíveis, por exemplo, com o número de pessoas alocadas, no tempo planejado, com o custo previsto, etc.

Devido ao fato da avaliação arquitetural basear-se na adequação de um sistema aos seus requisitos não funcionais, ela produz como resultado um relatório, não com respostas determinísticas como “sim”, “não”, ou “70%”, mas sim informando em que ponto um atributo de interesse é afetado, positiva ou negativamente, por alguma decisão arquitetural, mostrando a quais objetivos ela é adequada e a quais não é. Por isso, adequabilidade só é relevante num contexto em que os objetivos de qualidade são definidos. Por exemplo, uma arquitetura projetada para promover desempenho pode não ser adequada para sistemas em que a segurança é o atributo de maior interesse, mas pode ser adequada a outros sistemas. Isso significa que, se nenhum objetivo de qualidade for definido para um sistema, então qualquer arquitetura é adequada a esse sistema.

A priorização e análise de requisitos de qualidade é uma importante atividade do *Architecture Tradeoff Analysis Method* (ATAM) (KAZMAN; KLEIN; CLEMENTS, 2000), método de avaliação baseado em cenários do qual uma adaptação foi utilizada para avaliar a arquitetura proposta neste projeto. A avaliação baseada em cenários é o tipo de avaliação mais utilizado pela indústria. Dentre os métodos desse tipo, o ATAM é o mais utilizado na avaliação de sistemas de sistemas, do inglês *Systems of Systems* (SoS) (SANTOS *et al.*, 2014). SoS são sistemas que agregam sistemas constituintes independentes e heterogêneos para desempenhar capacidades novas e emergentes. Nesse sentido, a plataforma AGORA enquadra-se como um sistema de sistemas e, dessa forma, o ATAM apresenta-se como um método de avaliação adequado no contexto deste projeto. Além disso, esse método também favorece o detalhamento de certos documentos arquiteturais, o que se mostrou bastante útil ao projeto.

O ATAM leva a avaliação a revelar não apenas se os objetivos de qualidade foram satisfeitos, mas também como esses objetivos interagem uns com os outros, ou seja, como eles se contrabalançam (*trade-off*). Ele guia seus participantes, as partes interessadas, a procurarem por conflitos na arquitetura e a resolverem esses conflitos. ATAM possui três influências: o *Software Architecture Analysis Method* (SAAM), os atributos de qualidade e as abordagens arquiteturais.

O método SAAM utiliza-se de cenários para analisar uma arquitetura. Isso porque cenários são fáceis de criar e entender, baratos por exigirem pouco esforço para lidar com eles, e efetivos. Devido a isso, eles são um ótimo meio pelo qual todas as partes interessadas podem expressar seus interesses arquiteturais. Nesse contexto, um cenário é uma curta declaração descrevendo uma interação com o sistema do ponto de vista de alguma parte interessada. Por exemplo, um desenvolvedor pode descrever um cenário de implementação do sistema a partir da arquitetura; um mantenedor pode descrever uma mudança no sistema como a adição de um novo componente; um usuário pode descrever o uso do sistema para realizar certa tarefa. Um cenário é constituído de três partes: estímulo, ambiente e resposta. O estímulo é aquilo que uma parte interessada faz para iniciar a interação com o sistema, por exemplo, um usuário pode invocar uma função, um testador pode executar um caso de teste, ou um mantenedor pode fazer uma atualização. O ambiente é o que está ocorrendo durante o estímulo, por exemplo, se o sistema está sobrecarregado, se um processador não está funcionando, ou mesmo se tudo está sob condições normais (caso em que geralmente é omitido). A resposta é a consequência do estímulo gerada no sistema, por exemplo, se a função é executada, se o caso de teste obtém sucesso, ou de quanto foi o esforço para se fazer uma atualização. Geralmente, a resposta revela os atributos de qualidade de interesse das partes interessadas, o que faz com que os cenários representem, assim, requisitos para a arquitetura. Alguns exemplos de cenários são:

- “Um usuário remoto solicita um relatório de um banco de dados pela Internet durante um período de pico e recebe o relatório em cinco segundos.”
- “Adicionar um novo tipo de mensagem ao sistema com menos de um homem-semana de trabalho.”
- “Aumentar o número de contas processadas por hora por um fator de dez mantendo o tempo de resposta do pior caso abaixo de dez segundos.”

A criação de cenários é feita em uma reunião com todas as partes interessadas, promovendo a sua comunicação e as decisões para se atender seus interesses, que, por vezes, podem ser conflitantes. Além disso, as discussões dessa reunião ajudam a entender a arquitetura identificando os componentes envolvidos na execução de cada cenário e verificando se a arquitetura é capaz de satisfazer os cenários, evidenciando quais componentes devem ser alterados caso um cenário não seja atendido. Durante as discussões, os participantes levantam cenários rapidamente e de forma não estruturada, fazendo com que nem todos os cenários sejam registrados com estímulo, ambiente e resposta. Por isso, normalmente apenas os cenários de maior prioridade são estruturados.

Os atributos de qualidade são as características a que se referem os requisitos não funcionais de um sistema de software, por exemplo, desempenho, segurança, usabilidade, disponibilidade, portabilidade, etc. Os atributos de qualidade variam muito de sistema para sistema e dependem do domínio em que estão inseridos. Normalmente, é difícil analisar se um sistema satisfaz os atributos de qualidade relevantes em seu domínio, e sua arquitetura pode desempenhar um papel importante nessa análise. A documentação de uma arquitetura de software facilita a identificação das abordagens arquiteturais e, consequentemente, o entendimento das características da arquitetura, favorecendo a análise de seus atributos de qualidade.

Abordagens arquiteturais são os estilos ou quaisquer decisões arquiteturais adotadas pelo arquiteto, por exemplo, divisão das funcionalidades do sistema em serviços. Elas são importantes porque representam os meios empregados na arquitetura para atender os requisitos não funcionais do sistema do qual a arquitetura serve de base. Observando-se as abordagens adotadas é possível verificar certas características, por exemplo, o modo como o sistema correspondente pode evoluir, enfrentar ameaças de segurança, ou integrar-se a outros sistemas. Essas características evidenciam se, e o quanto a arquitetura satisfaz os atributos de qualidade importantes em seu domínio. Por exemplo, ao se observar as decisões adotadas para facilitar o uso do sistema por diversas aplicações pode-se verificar se a interoperabilidade esperada para o sistema é atendida. O método ATAM utiliza-se das abordagens arquiteturais comparando-as, em uma primeira fase, com os requisitos de qualidade e seus atributos associados e, em uma segunda fase, com os cenários arquitetu-

rais para verificar se os requisitos não funcionais do sistema referente à arquitetura em avaliação foram atendidos.

Clements, Kazman e Klein (2002) consideram que a arquitetura de software é onde os interesses dos diferentes envolvidos com o sistema em questão se encontram e, portanto, é onde esses interesses devem ser analisados em conjunto. Analisar os interesses de todas as partes envolvidas é uma tarefa importante porque eles não estão todos explícitos em um documento de requisitos, e, além disso, muitas vezes são conflitantes, por exemplo, o número de funcionalidades que o cliente deseja pode entrar em conflito com o tempo disponível dos desenvolvedores. Dessa forma, esses interesses devem ser capturados dos próprios interessados para serem mediados e considerados adequadamente na arquitetura. Portanto, durante a avaliação arquitetural, deve-se assegurar que haverá participantes de forma que seus interesses englobem todos os interesses relevantes ao sistema em questão. Por exemplo, caso espere-se que o sistema suporte modificações para acompanhar a evolução de seu ambiente, deve estar presente alguém que possa dizer quais são os tipos de modificações esperadas; entretanto, caso usabilidade não seja um requisito relevante, então a falta de um especialista em interface de usuário não comprometerá a avaliação. Isso significa que a qualidade da avaliação arquitetural depende da representatividade de seus participantes.

O método ATAM sugere três grupos de participantes no sentido de se reunir a maioria dos pontos de vista relevantes em uma avaliação de arquitetura de software, a saber:

- Time de avaliação: responsável por realizar a avaliação como um todo. Os membros desse time incluem o líder, responsável por conduzir o progresso da avaliação, o escrevente, responsável por registrar pontos importantes, e os questionadores, responsáveis por reconhecer abordagens arquiteturais e levantar questões referentes a elas;
- Time de projeto: responsável por considerar o domínio de aplicação e os recursos disponíveis. Nesse time estão inclusos o gerente do projeto, responsável por considerar as regras de negócio e alocar recursos como tempo e custo, o arquiteto, responsável pela arquitetura em avaliação, e os desenvolvedores, responsáveis por implementar a arquitetura;
- Partes interessadas: responsáveis por verificar as funcionalidades do sistema. Incluem-se nesse time o cliente, responsável por levantar atributos de qualidade e assegurar o cumprimento dos objetivos do sistema, e os usuários finais, responsáveis por levantar cenários e verificar se as funcionalidades necessárias são atendidas;

De forma geral, as partes interessadas são todos os membros dos três grupos,

sendo que o nome de cada grupo serve apenas como uma referência do conjunto de suas atribuições. Além disso, cada grupo não necessariamente deve ter exatamente os membros citados, sendo essa caracterização apenas uma sugestão, observando-se que o importante é que os participantes representem de forma abrangente os interesses envolvidos. Ainda, cada responsabilidade não é restrita ao papel correspondente, por exemplo, o desempenho do papel de questionador, na prática, deve ser estimulado em todos os participantes.

A avaliação arquitetural definida pelo método ATAM é executada em duas fases, cada uma compreendendo uma reunião com um conjunto de participantes, totalizando três dias de duração. A primeira fase possui seis etapas, realizadas em um dia de trabalho, e conta com o time de avaliação e o time de projeto. A segunda fase possui as mesmas seis etapas da primeira, realizadas em um outro dia, e mais três, realizadas no dia seguinte, contando com o time de avaliação, de projeto e todas as partes interessadas. Na segunda fase, as seis primeiras etapas são recapituladas para os novos participantes, sendo, portanto, mais breves que na primeira fase. Aqui, considera-se um dia de trabalho os períodos da manhã e tarde com intervalos de descanso. As nove etapas definidas pelo método são apresentadas a seguir.

A etapa 1 consiste em o líder do time de avaliação apresentar o método que os participantes seguirão para avaliar a arquitetura, incluindo suas etapas e técnicas, como a identificação e análise de abordagens arquiteturais, atributos de qualidade e cenários. A etapa 2 consiste em o gerente de projeto ou o cliente apresentar o contexto do sistema referente à arquitetura, incluindo questões gerenciais e políticas, as regras de negócio e as principais funções e atributos de qualidade do sistema. A etapa 3 consiste em o arquiteto apresentar a arquitetura, incluindo restrições as serem obedecidas, como hardware e sistemas com os quais deve interagir, decisões e padrões arquiteturais usados para satisfazer os requisitos de qualidade, sistemas de terceiros usados internamente (*commercial off-the-shelf*, COTS) e exemplos de cenários arquiteturais. Nessa etapa o time de avaliação inicia a identificação das abordagens arquiteturais. A etapa 4 consiste em o time de avaliação identificar, mas ainda não analisar, as decisões arquiteturais, incluindo as que o arquiteto conscientemente utilizou e apresentou anteriormente e outras que possam ter sido capturadas durante a observação da documentação apresentada pelo arquiteto.

A etapa 5 consiste em todos os participantes dessa fase identificarem, refinarem e priorizarem os objetivos de qualidade mais importantes do sistema. Isso é feito elaborando-se uma árvore de utilidade de atributos de qualidade, que traduz os objetivos de negócio em atributos de qualidade concretos, forçando os participantes a definirem os requisitos precisamente. Uma árvore de atributos tem o nó “Utilidade” como raiz, ramificando-se em atributos de qualidade no segundo nível da árvore, como desempenho, disponibilidade, etc. Cada atributo então é refinado até que possam ser analisados como requisitos, eliminando declarações vagas como “A arquitetura deve ter alta disponibilidade”. A [Figura 3](#) mostra

um exemplo de árvore de utilidade.

Figura 3 – Exemplo de árvore de utilidade de atributos de qualidade (adaptada de Clements, Kazman e Klein (2002)).

Após a elaboração da árvore, os requisitos são priorizados, o que pode ser feito da seguinte maneira: a cada requisito atribui-se a prioridade “Alta” (A), “Média” (M) ou “Baixa” (B) sob duas dimensões, (i) a importância do requisito para o sucesso do sistema e (ii) o grau de dificuldade de se satisfazer o requisito. Sob a perspectiva da importância dos requisitos, a prioridade é atribuída por consenso entre os participantes, já em relação à dificuldade de alcançar um requisito, a prioridade é atribuída pelo arquiteto. Por exemplo, a priorização do requisito “(M,B) Minimizar latência de armazenamento no banco do cliente em 200ms” da Figura 3 significa que ele tem importância média no sucesso do sistema e que o arquiteto espera uma dificuldade baixa para alcançá-lo. Essa etapa de criação da árvore de utilidade é importante para evitar que os participantes tenham impressões diferentes sobre quais são os atributos de qualidade mais relevantes, tornando claro e comum a todos quais são os atributos que realmente têm grande influência para o sucesso do sistema.

A etapa 6 consiste em analisar as decisões arquiteturais, capturadas nas etapas 3 e 4, comparando-as com os requisitos de qualidade de maior prioridade, capturados na etapa 5, para verificar o quão bem a instanciação dessas decisões atendem os requisitos para os quais foram projetadas. Os requisitos a serem analisados são, primeiramente, aqueles priorizados como (A,A), ou seja, aqueles que são importantes para o sistema e difíceis de se alcançar. Em seguida, analisa-se os priorizados como (A,M) ou (M,A). Por fim, caso ainda haja tempo, analisa-se os priorizados como (M,M). Os requisitos priorizados com “B” em qualquer dimensão não precisam ser analisados, já que há pouca vantagem em

investir tempo discutindo casos de pouca relevância ao sistema ou de pouca dificuldade de obtenção.

A comparação das decisões com os requisitos visa principalmente encontrar e registrar pontos sensíveis, riscos, não riscos e pontos de contrabalanço. Pontos sensíveis são decisões arquiteturais que impactam a obtenção de um ou mais atributos de qualidade. São assim chamados porque a obtenção do atributo de qualidade é sensível à mudança da decisão correspondente, por exemplo, o nível de confidencialidade em uma rede privada (segurança) pode ser sensível ao número de bits usados em sua criptografia (decisão), ou seja, alterando-se o número de bits pode-se alterar o nível de confidencialidade resultante. Riscos são decisões arquiteturais potencialmente problemáticas. No caso do exemplo acima, usar um número muito baixo de bits na criptografia pode não atender a segurança esperada, tornando essa decisão um risco. Riscos também podem surgir devido a fatores externos à arquitetura, por exemplo, um orçamento insuficiente para o projeto, caso em que também devem ser registrados. Não riscos são boas decisões arquiteturais, que atendem os requisitos de qualidade para os quais foram projetadas. Observa-se que um não risco pode se tornar um risco quando as condições assumidas na tomada da decisão correspondente mudam, e, portanto, a decisão deve ser reanalisada. Por exemplo, no caso anterior, se a segurança não for um atributo prioritário (condição assumida) a decisão de usar um número baixo de bits na criptografia é um não risco; porém, caso a segurança torne-se prioritária, a decisão tomada pode passar a ser um risco. Pontos de contrabalanço (*trade-off*) são pontos sensíveis que afetam mais de um atributo de qualidade de maneira oposta. No caso do exemplo anterior, usar um número alto de bits na criptografia pode promover a segurança do sistema mas comprometer seu desempenho.

Nessa etapa, as vantagens e desvantagens conhecidas dos padrões arquiteturais adotados facilitam, respectivamente, a identificação de alguns não riscos e riscos, porém, outras eventuais decisões também devem ser analisadas. Em particular, a identificação de pontos sensíveis é importante porque eles revelam onde focar para se compreender os pontos por meio dos quais os requisitos de qualidade são satisfeitos, além de mostrar que é necessário cuidado em se alterar esses pontos, devendo-se, dessa forma, classificar todos os pontos sensíveis como risco ou não risco, dependendo se a decisão correspondente atinge ou não seus objetivos. Nesse sentido, os pontos de contrabalanço correspondem às decisões mais importantes na arquitetura devido ao seu caráter de contrabalanceamento de requisitos, exigindo maior atenção em sua avaliação. O registro dos riscos e não riscos deve ser feito cuidadosamente, incluindo uma justificativa para seu efeito positivo ou negativo, pois, além de revelar pontos importantes da arquitetura, podem ser consultados em avaliações futuras para facilitar a tomada de decisões arquiteturais. Por fim, também deve-se analisar e registrar situações em que ainda nenhuma decisão arquitetural foi tomada para se satisfazer algum requisito, caso em que a falta de decisão provavelmente será classificada como risco, já que o requisito correspondente dificilmente será atendido. O

Quadro 1 representa uma ficha de análise de requisito que pode ser usada para facilitar as análises desta etapa, na qual se observa o registro dos identificadores dos pontos sensíveis, pontos de contrabalanço, riscos e não riscos de cada decisão.

Quadro 1 – Exemplo de ficha de análise de requisito (adaptado de Clements, Kazman e Klein (2002)).

Análise da Decisão Arquitetural				
Id.: R1		Requisito: Texto.		
Atributo(s): Disponibilidade.				
Ambiente: Texto.				
Estímulo: Texto.				
Resposta: Texto.				
Decisão arquitetural	Sensibilidade	Contrabalanço	Risco	Não Risco
D1: Texto.	S1	T1		N1
...	...	...	...	...
Justificativa: Texto.				

A etapa 7 consiste em todos os participantes da avaliação identificarem cenários de aplicação da arquitetura e priorizá-los. Idealmente os cenários devem ser abrangentes para considerar todas as capacidades e limitações da arquitetura segundo o ponto de vista de cada parte interessada. Uma forma de facilitar a identificação de cenários diferentes é tentar encontrar cenários de três tipos, casos de uso, evolução e exploratórios. Cenários de casos de uso representam as formas como o sistema a que a arquitetura se refere será usado. Cenários de evolução representam as formas pelas quais a arquitetura proverá suporte a modificações de curto e médio prazo já previstas, por exemplo, adição de componentes, integração com outros sistemas ou mudanças de desempenho ou disponibilidade. Esse tipo de cenário ajuda a evidenciar as forças e fraquezas da arquitetura. Cenários exploratórios representam formas extremas de mudanças que exigem o máximo da arquitetura, por exemplo, mudança da infraestrutura ou dos objetivos do sistema, sobrecarga de processamento ou de conexões ou novos requisitos de grande relevância. Esses cenários são voltados à identificação de pontos sensíveis e de contrabalanço, os quais ajudam a avaliar os limites da arquitetura. Elicitar cenários com esses três tipos em mente ajuda na obtenção de um conjunto abrangente e representativo de cenários, mas não necessariamente cada cenário deve ser classificado em um desses tipos.

Após a identificação, os cenários devem ser priorizados. Isso pode ser feito por uma votação da seguinte maneira: cada participante dispõe de uma quantidade de pontos igual a trinta por cento do número de cenários, por exemplo, se houver dezenove cenários cada participante dispõe de seis pontos. Então, cada participante atribui, independentemente dos outros, os pontos aos cenários que julgarem mais importantes da forma que quiser. No exemplo anterior, pode-se atribuir seis pontos a um único cenário, um ponto em seis cenários diferentes ou qualquer distribuição entre essas duas formas. Em seguida, os

pontos de cada cenário são somados e ordena-se os cenários de acordo com suas pontuações totais. Por fim, seleciona-se uma quantidade de cenários dentre aqueles com maior total de pontos, por exemplo, os cinco cenários de maior pontuação. Os cenários selecionados serão analisados na etapa seguinte.

A etapa 8 é semelhante à 6, em que as decisões arquiteturais foram analisadas comparando-as com os requisitos de qualidade da árvore de utilidade. Nesta etapa, porém, os cenários são utilizados para complementar a árvore considerando o ponto de vista das partes interessadas envolvidas na etapa 7. O objetivo desta etapa é comparar e alinhar os requisitos de qualidade que o arquiteto e os demais participantes acreditam ser importantes ao sistema pretendido. Os cenários selecionados para análise são adicionados como nós folhas da árvore de utilidade de acordo com o atributo de qualidade a que mais se adequam, verificado por consenso dos participantes. Assim, a adição pode ocorrer de três formas: um cenário pode representar um requisito presente na árvore e, portanto, já analisado, pode representar um requisito novo de um atributo de qualidade já considerado, ou pode representar um requisito de um atributo novo. Para o primeiro caso, considera-se a análise já realizada; para os outros dois, cria-se nós e ramifica-se a árvore, caso necessário, para se chegar ao nível de detalhe adequado, prioriza-se os novos requisitos de acordo com sua importância e dificuldade de obtenção e repete-se o processo de análise da etapa 6, registrando-se pontos sensíveis, riscos, não riscos e pontos de contrabalanço dos novos requisitos de prioridade alta e média. Verifica-se aqui que pouco esforço será necessário se as visões do arquiteto e das demais partes interessadas estiverem alinhadas, pois a maioria dos cenários representarão requisitos já analisados.

A última etapa, 9, consiste em o líder do time de avaliação recapitular as etapas do método e sumarizar e apresentar aos participantes as informações coletadas durante a avaliação. Isso inclui o contexto de negócio, a arquitetura e, principalmente, os artefatos gerados pelo ATAM, ou seja, as abordagens arquiteturais, os atributos de qualidade, os requisitos (árvore de utilidade) e cenários priorizados, os riscos, não riscos, pontos sensíveis e de contrabalanço. Além disso, os riscos identificados podem ser agrupados em temas baseados em características comuns, por exemplo, riscos relacionados a inadequação de documentos ou a cópias de segurança insuficientes. Os temas de risco são então relacionados aos objetivos de negócio que eles afetam, o que facilita a tomada de decisão por parte de gerentes. Nesta fase, o time de avaliação pode também fazer recomendações, tanto no projeto da arquitetura quanto nos processos do ambiente que o envolvem, por exemplo, melhorar a comunicação entre desenvolvedores. Porém, essas recomendações não fazem parte de fato do ATAM, o qual concentra-se em identificar os pontos positivos e negativos de uma arquitetura de software. Um resumo do *Architecture Tradeoff Analysis Method* é apresentado no [Quadro 2](#). Ressalta-se que, apesar da numeração, as etapas não precisam estritamente seguir essa ordem, podendo ser adaptadas de acordo com a necessidade. Por exemplo, pode-se voltar a uma etapa anterior, pular momentaneamente uma etapa ou

iterar sobre algumas etapas se assim os participantes acharem necessário; o importante é ter em mente quais são as atividades envolvidas e os artefatos resultantes.

Quadro 2 – Resumo das etapas do ATAM (adaptado de Clements, Kazman e Klein (2002)).

ETAPA	1ª FASE		2ª FASE	
	Tempo	Participantes	Tempo	Participantes
1. Apresentar o ATAM	Dia 1	Time de avaliação + Time de projeto	Dia 2	Time de avaliação + Time de projeto + Partes interessadas
2. Apresentar as diretivas de negócio				
3. Apresentar a arquitetura				
4. Identificar decisões arquiteturais				
5. Gerar árvore de utilidade de atributos de qualidades				
6. Analisar decisões arquiteturais				
7. Identificar e priorizar cenários	N/A		Dia 3	
8. Analisar decisões arquiteturais				
9. Apresentar os resultados				

2.2 Gestão de Desastres Naturais

Em todo o mundo vem se observando, ao longo do tempo, um grande número de desastres naturais que afligem populações vulneráveis. Um desastre natural ocorre quando uma sociedade é atingida por um evento natural e seus danos são tão grandes que a ajuda externa é necessária (LONGUEVILLE *et al.*, 2010). Assim, medidas reativas para mitigar esses danos e evitar que eles transformem o evento em uma catástrofe devem ser adotadas (BAHARIN; SHIBGHATULLAH; OTHMAN, 2009; WANG; YUAN, 2010; POSER; DRANSCH, 2010). Com isso, é possível reduzir a perda de recursos e aprimorar o poder de resiliência da comunidade, ou seja, aperfeiçoar a capacidade da comunidade em resistir, mudar ou adaptar-se a fim de obter um novo nível de funcionamento ou estrutura na ocorrência de desastres e eventos extremos (MENDIONDO, 2010; NORRIS *et al.*, 2008).

A gestão de desastres insere-se nesse contexto e se faz necessária para evitar e reduzir o impacto dos desastres naturais (BAHARIN; SHIBGHATULLAH; OTHMAN, 2009). Ela é definida por um processo cíclico contínuo composto por um conjunto de atividades realizadas antes, durante e depois de um evento desse tipo, dividido em quatro fases principais: (1) mitigação, focada em redução e eliminação de riscos, (2) preparação, que trata do planejamento da resposta, (3) resposta, que objetiva reestabelecer e manter a segurança pública, e (4) recuperação, que é o processo de restaurar as condições de vida nas áreas afetadas (POSER; DRANSCH, 2010). A Figura 4 apresenta essas fases com exemplos de atividades associadas a cada uma delas.

As informações utilizadas pelas agências de emergência responsáveis por gerenciar esses eventos exercem um papel fundamental para garantir a redução dos impactos as-

sociados. Por isso é importante que elas sejam precisas, oportunas e completas sobre o estado atual das variáveis ambientais (HORITA *et al.*, 2013), bem como sobre os prognósticos científicos sobre as próximas mudanças e seus impactos. Na gestão de inundações em específico, destaca-se a importância da utilização de sensores locais, como pluviômetros e sensores de nível dos rios, além de dados geográficos e topográficos, para aquisição de informações fundamentais para alimentar modelos hidrológicos e a análise de risco e previsão de, por exemplo, nível de rios e tendências de precipitação.

Figura 4 – Ciclo de desastre (adaptada de Poser e Dransch (2010)).

Observa-se, entretanto, que, devido às preferências específicas de informação de cada uma dessas agências, elas tendem a utilizar seu próprio conjunto de estruturas, tanto de dados quanto de sistemas (BAHARIN; SHIBGHATULLAH; OTHMAN, 2009). Como resultado, muitas informações relativas aos desastres e mudanças climáticas estão dispersas ao longo de várias fontes distintas não relacionadas e sem um padrão de metadados para descrever seus recursos (TU; CHOU; CHU, 2009), dificultando seu acesso, utilização e compreensão.

2.3 Informações Geográficas de Sensores

Sensores são dispositivos capazes de coletar dados do ambiente e de se comunicar com outros sensores, formando uma rede, enviando e recebendo dados uns dos outros. O *Open Geospatial Consortium* (OGC) criou uma iniciativa chamada *Sensor Web Enablement* (SWE) para tornar redes de sensores acessíveis pela Internet e padronizar a

comunicação de aplicações com essas redes (BOTTS *et al.*, 2008). Assim, protocolos de padrões abertos e interfaces de programação de aplicação, do inglês *application programming interfaces* (APIs), foram desenvolvidos para facilitar a descoberta e acesso de redes de sensores, e a troca e processamento de observações distribuídas de sensores de forma interoperável (SIMONIS, 2008; ZHAO; FOERSTER; YUE, 2012).

Dentre as especificações estabelecidas pelo OGC, observa-se as seguintes:

- *Geography Markup Language* (GML): é um padrão XML (*eXtensible Markup Language*) para modelagem de dados geográficos e transferência de dados pela Internet. Com ele é possível referenciar pontos, linhas e polígonos, entretanto, esquemas de aplicações específicos podem ser utilizados para abstrair informações, podendo assim referenciar-se, por exemplo, estradas, pontes e prédios. Sistemas que utilizam o mesmo esquema podem trocar informações facilmente, mantendo os dados consistentes.
- *Web Processing Service* (WPS): fornece um meio de disponibilizar processamentos geoespaciais em forma de serviços web. O WPS define o modo como processos e suas entradas e saídas são descritas, como um cliente pode requisitar a execução de um processo, e como a saída de um processo deve ser manuseada. Neste contexto, “processo” pode ser qualquer algoritmo, cálculo ou modelo que opere sobre dados georreferenciados, sendo que esses dados podem estar tanto no próprio servidor quanto na Internet, e serem de qualquer tipo, inclusive outros serviços web do OGC. A ideia por trás do WPS é oferecer qualquer tipo de processamento, simples e complexos, para sistemas de informação geográficos (SIG) (Stollberg e Zipf, 2007).
- *Observations and Measurements* (O&M): especifica uma implementação GML de um modelo conceitual de observações e medidas de sensores, incluindo características de amostragem. Ele provê modelos para a troca de informações descrevendo ações de observação e seus resultados entre comunidades científicas e técnicas.
- *Sensor Observation Service* (SOS): define uma interface que permite consultar observações e metadados de sensores, remover e registrar novos sensores, inserir novas observações e outras funcionalidades de controle de observações. Um exemplo de operação definida nessa especificação é a *getCapabilities*, codificada segundo o padrão O&M, que fornece informações sobre o próprio serviço com seus metadados, por exemplo, quais outras operações estão disponíveis, quais sensores estão associados a esse serviço e quais propriedades esses sensores medem. Outras operações desse padrão, também codificadas em O&M, incluem *getObservation*, *insertObservation* e *deleteObservation*, que, respectivamente, recuperam, inserem e apagam observações do conjunto de dados gerenciado pelo servidor SOS. O [Código-fonte 1](#) apresenta um exemplo da operação de requisição *getObservation*.

Código-fonte 1: Exemplo da operação de requisição *getObservation* de acordo com padrão O&M

```

1 <!-- Fonte: adaptado de https://github.com/52North/SOS/blob/develop/
   webapp/src/main/webapp/static/examples/sos\_v20/requests\_xml/
   GetObservation.xml. Acessado em 14/01/2016 -->
2
3 <?xml version="1.0" encoding="UTF-8"?>
4 <sos:GetObservation service="SOS"
5   version="2.0.0"
6   xmlns:sos="http://www.opengis.net/sos/2.0"
7   xmlns:fes="http://www.opengis.net/fes/2.0"
8   xmlns:gml="http://www.opengis.net/gml/3.2"
9   xmlns:swe="http://www.opengis.net/swe/2.0"
10  xmlns:xlink="http://www.w3.org/1999/xlink"
11  xmlns:swes="http://www.opengis.net/swes/2.0"
12  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
13  xsi:schemaLocation="http://www.opengis.net/sos/2.0
14    http://schemas.opengis.net/sos/2.0/sos.xsd">
15
16  <swes:extension>
17    <swe:Boolean definition="MergeObservationsIntoDataArray">
18      <swe:value>true</swe:value>
19    </swe:Boolean>
20  </swes:extension>
21  <sos:offering>GAUGE_HEIGHT</sos:offering>
22  <sos:observedProperty>
23    urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel
24  </sos:observedProperty>
25  <sos:temporalFilter>
26    <fes:TEquals>
27      <fes:ValueReference>phenomenonTime</fes:ValueReference>
28      <gml:TimeInstant gml:id="ti_1">
29        <gml:timePosition>latest</gml:timePosition>
30      </gml:TimeInstant>
31    </fes:TEquals>
32  </sos:temporalFilter>
33  <sos:featureOfInterest>foi_1401</sos:featureOfInterest>
34  <sos:featureOfInterest>foi_3001</sos:featureOfInterest>
35  <sos:spatialFilter>
36    <fes:BBBOX>
37      <fes:ValueReference>
38        http://www.opengis.net/req/omxml/2.0/data/samplingGeometry
39      </fes:ValueReference>
40      <gml:Envelope srsName="http://www.opengis.net/def/crs/EPSCG
41        /0/4326">
42        <gml:lowerCorner>-22.077147 -47.951247</gml:lowerCorner>

```

```

42         <gml:upperCorner>-21.959933 -47.840868</gml:upperCorner>
43     </gml:Envelope>
44 </fes:BOX>
45 </sos:spatialFilter>
46 <sos:responseFormat>
47     http://www.opengis.net/om/2.0
48 </sos:responseFormat>
49 </sos:GetObservation>

```

- *SWE Common Data Model*: define modelos de dados de baixo nível que permitem aplicações e/ou servidores estruturarem, codificarem e transmitirem conjuntos de dados de sensores de forma auto descritiva e semântica. Alguns exemplos de modelos de dados são *boolean*, *text*, *category* (similar a enumeração), *count* (números discretos), *quantity* (números contínuos), *time*, *dataArray*, *matrix*, entre outros. O [Código-fonte 2](#) apresenta um conjunto de dados de acordo com essa especificação exemplificando o uso de alguns de seus modelos.

Código-fonte 2: Exemplo de dados geográficos de acordo com o padrão *SWE Common Data Model*

```

1  <!-- Fonte: http://schemas.opengis.net/sweCommon/2.0/examples/spec/
   array_weather.xml. Acessado em 14/01/2016 -->
2
3  <?xml version="1.0" encoding="UTF-8"?>
4  <swe:DataArray definition="http://sweet.jpl.nasa.gov/2.0/info.owl#
   TimeSeries"
5      xmlns:swe="http://www.opengis.net/swe/2.0"
6      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
7      xmlns:xlink="http://www.w3.org/1999/xlink"
8      xsi:schemaLocation="http://www.opengis.net/swe/2.0
9          http://schemas.opengis.net/sweCommon/2.0/swe.xsd">
10
11     <swe:description>
12         Array of synchronous weather measurements
13     </swe:description>
14     <swe:elementCount>
15         <swe:Count>
16             <swe:value>5</swe:value>
17         </swe:Count>
18     </swe:elementCount>
19     <swe:elementType name="weather_measurement">
20         <swe:DataRecord>
21             <swe:label>Weather Data Record</swe:label>
22             <swe:field name="time">
23                 <swe:Time definition="http://www.opengis.net/def/property/
OGC/0/SamplingTime">

```

```

24         <swe:label>Sampling Time</swe:label>
25         <swe:uom xlink:href="http://www.opengis.net/def/uom/ISO
-8601/0/Gregorian"/>
26     </swe:Time>
27 </swe:field>
28 <swe:field name="temperature">
29     <swe:Quantity definition="http://mmisw.org/ont/cf/parameter/
air_temperature">
30         <swe:label>Air Temperature</swe:label>
31         <swe:uom xlink:href="Cel"/>
32     </swe:Quantity>
33 </swe:field>
34 <swe:field name="pressure">
35     <swe:Quantity definition="http://mmisw.org/ont/cf/parameter/
air_pressure_at_sea_level">
36         <swe:label>Atmospheric Pressure</swe:label>
37         <swe:uom code="mbar"/>
38     </swe:Quantity>
39 </swe:field>
40 </swe:DataRecord>
41 </swe:elementType>
42 <swe:encoding>
43     <swe:TextEncoding blockSeparator="&#10;" tokenSeparator=","/>
44 </swe:encoding>
45 <swe:values>
46     2009-02-10T10:42:56Z,25.4,1020
47     2009-02-10T10:43:06Z,25.3,1021
48     2009-02-10T10:44:16Z,25.3,1020
49     2009-02-10T10:45:26Z,25.4,1022
50     2009-02-10T10:46:36Z,25.4,1022
51 </swe:values>
52 </swe:DataArray>

```

2.4 Informações Geográficas de Voluntários

Informações geográficas de voluntários podem ser definidas como uma coleção de dados espaciais digitais produzidos por indivíduos e instituições não formais, ou seja, cidadãos usando ferramentas apropriadas para colher e disseminar suas visões e conhecimento geográfico na *web* (GOODCHILD, 2007). VGI pode ser obtida por meio de diferentes atividades colaborativas: (i) compartilhamento de informações em mídias sociais, por exemplo Twitter, (ii) mapeamento colaborativo, por exemplo usando Wikimapia e OpenStreet-Map¹ e (iii) sensoriamento participativo, por exemplo por meio de observatórios cidadãos (HORITA *et al.*, 2015).

¹ <http://www.openstreetmap.org/> Acessado em 31/01/2016.

Nesse sentido, mensagens de texto, fotos, vídeos ou gravações de áudio sobre um determinado acidente podem ser considerados VGI, apresentando assim uma grande variação tanto de estrutura quanto de conteúdo. Isso pode dificultar seu uso em sistemas geográficos, principalmente de maneira automática. Dessa forma, o principal desafio relacionado a esse tipo de fonte de informação é como integrá-la a outros conjuntos de dados para gerar uma abordagem mais efetiva (POSER; DRANSCH, 2010).

VGI emergiu nos últimos anos como uma fonte de dados espaciais alternativa na Internet (OLIVEIRA *et al.*, 2014) e tem sido cada vez mais reconhecida por pesquisadores como um importante recurso para apoiar a gestão de desastres (CÂMARA *et al.*, 2014; POSER; DRANSCH, 2010). A utilização desse tipo de informação é uma abordagem que foi considerada eficaz pela pesquisa realizada por Erskine e Gregg (2012), Moreira, Degrossi e Albuquerque (2015). Em seu trabalho, indivíduos contribuíram com dados para um sistema centralizado que transmite mapas relevantes e precisos da área de um desastre para as agências de emergência responsáveis utilizando telefones celulares. Segundo os autores, os resultados da pesquisa mostraram a efetividade das informações geográficas voluntárias para desenvolver, em tempo real, um mapeamento eficaz e confiável da situação em questão, sendo, em muitos casos, mais eficaz que os sistemas tradicionais de resposta a desastres.

Poser e Dransch (2010) também realizaram um trabalho semelhante. Para avaliar a viabilidade da utilização de dados providos de voluntários para a modelagem empírica dos danos causados por uma inundação, eles utilizaram informações obtidas por meio de contatos telefônicos a vítimas da enchente que atingiu o município de Eilenburg na Alemanha em 2002. Os resultados apresentados tornaram evidente a eficácia dessa abordagem, apesar de revelar a necessidade de estudos mais detalhados em relação a validação dos resultados para outras inundações, de utilização de métodos automatizados para controle de qualidade e de integração com sensores remotos.

Goodchild e Glennon (2010) apontam também alguns fatos interessantes em relação a essa abordagem, como o de que, em uma crise, é mais difícil distinguir informações oficiais e voluntárias, e que a disseminação das informações voluntárias é mais rápida do que a das oficiais. Apesar dos problemas observados, tal abordagem apresenta-se como uma alternativa para fornecer informações ricas e atualizadas, auxiliando a atuação das agências de emergência e facilitando a organização da comunidade para a reação e recuperação de desastres (GOODCHILD, 2007; GILL; BUNKER, 2012).

2.5 Integração de Informações Geográficas

Apesar de redes de sensores contribuírem para a gestão de desastres naturais, no caso da gestão de inundações em particular pode haver áreas não medidas devido à pos-

sível grande extensão dos rios monitorados, gerando lacunas de informação (HORITA *et al.*, 2015). Nesse sentido, a integração de informações oficiais com dados de voluntários pode gerar benefícios como a produção de informações em escala local e o aumento da base de conhecimentos necessários à gestão de desastres (GIORDANO; PREZIOSI; ROMANO, 2013). Ainda, integrar dados de diversas fontes possibilita estimar a extensão de uma inundação quando dados de sensores são incompletos (SCHNEBELE *et al.*, 2014). Entretanto, as fontes de dados geográficos geralmente são distribuídas, heterogêneas e autônomas, tornando sua integração um desafio (GROSSO; BOUJU; MUSTIÈRE, 2009). Por exemplo, o processo de integração pode envolver tratar dados de precisões variadas e em diferentes estruturas, identificar informações equivalentes de conjuntos diferentes, uniformizar dados com estruturas semelhantes mas com conteúdos variados e relacionar conceitos semânticos de diferentes informações (BUTENUTH *et al.*, 2007; RAMOS; VAN-DECASTEELE; DEVILLERS, 2013).

Nesse contexto, a integração de dados geográficos compreende tornar conjuntos de dados compatíveis uns com outros para que, assim, eles possam ser razoavelmente apresentados em um mesmo mapa de modo que seus relacionamentos possam ser analisados (FLOWERDEW, 1991). Por exemplo, dados geográficos podem ser integrados em nível de esquemas de bancos de dados, como no método proposto por Ciferri (1995), entretanto, o foco deste projeto não é o método de integração, mas sim o suporte aos algoritmos de integração. No domínio geoespacial, atributos raramente são uniformes, e alguns tipos de interpolação normalmente são necessários para atribuir devidamente esses atributos às estimativas resultantes no conjunto de dados integrados. A seleção de um modelo de interpolação apropriado depende bastante dos tipos de dados, do grau de precisão desejado e da quantidade de esforço computacional proporcionado (FOTHERINGHAM; ROGERSON, 1993). Por exemplo, o processo de kriging (OLIVER; WEBSTER, 1990) tem sido amplamente utilizado no domínio de análises espaciais e experimentos computacionais.

2.6 Trabalhos Relacionados

Alguns trabalhos relacionados a esta pesquisa foram analisados para prover conhecimento sobre como a integração de dados de sensores e voluntários tem sido feita no contexto de gestão de desastres naturais, além de evidenciar os desafios e lacunas de pesquisa nessa área. Por exemplo, Kotsev *et al.* (2015) propuseram uma abordagem para permitir a interoperabilidade de sensores ambientais combinando armazenamento local e capacidades de serviços web. Por meio dessa abordagem, dados de observações ambientais podem facilmente ser combinados e reusados. Apesar de combinar dados de sensores, mesmo aqueles produzidos por dispositivos móveis de cidadãos comuns, essa abordagem não considera informações de voluntários, ou seja, dados geográficos voluntariamente disponibilizados, os quais apresentam estruturas diferentes em relação aos dados de sensores.

Desse modo, essa abordagem não possibilita a utilização de métodos espaço-temporais para a integração de dados de sensores e de voluntários, mas pode ser usada como fonte de dados de sensores.

Albuquerque *et al.* (2015) relacionaram dados de autoridades com de voluntários para aprimorar a identificação dos dados de voluntários que são relacionados a inundações. Os dados de autoridades foram derivados para se obter regiões afetadas por uma determinada inundação e a severidade com que essa inundação afetou essas regiões. Em seguida, mensagens georreferenciadas da plataforma Twitter, filtradas por palavras-chave sobre inundação, foram associadas à região afeta mais próxima. Com isso, análises estatísticas foram feitas para se identificar padrões espaciais entre as mensagens e as regiões afetadas a fim de verificar se o conteúdo de mensagens próximas a regiões severamente afetadas está mais relacionado às inundações em relação ao das mensagens afastadas. Nesse trabalho, o processo de identificação de mensagens combina dados de autoridades e de voluntários, entretanto, seu resultado não engloba informações de ambas as fontes. Além disso, esse processo é bastante específico e não torna os dados interoperáveis para que métodos de integração sejam facilmente aplicados sobre eles.

Schade *et al.* (2013) propuseram uma abordagem que compara conceitos da iniciativa SWE a conceitos de informações geográficas de voluntários. A partir disso, os autores desenvolvem um processo de detecção de eventos em que, inicialmente, um único serviço web monitora diversas fontes de informações de voluntários, como Flickr e Twitter. Em seguida, informações referentes ao evento de interesse são coletadas dessas fontes e integradas por métodos específicos para essas informações. Então, análises também específicas são realizadas sobre essas informações para se verificar a ocorrência de um evento. Desse modo, esse trabalho integra dados de voluntários de diversas fontes, porém, ainda não os integra a dados de sensores. Apesar da comparação de conceitos envolvendo a SWE, o trabalho não apresenta um exemplo detalhado de informação de voluntário codificada segundo um padrão do OGC para permitir sua integração com dados de sensores. Mesmo dados de voluntários codificados segundo um padrão do OGC podem apresentar dificuldades para serem integrados a dados de sensores, como evidenciado no processo de integração definido neste projeto e detalhado na subseção 3.4.2.

Bakillah *et al.* (2013) desenvolveram um modelo para obter interoperabilidade semântica entre dados de sensores e de voluntários. Os dados de sensores correspondem a observações de fenômenos naturais estruturadas segundo os padrões do OGC, e os de voluntários podem ser quaisquer dados de aplicações que se registrem ao sistema que implementa o modelo, por exemplo aplicações como OpenStreetMap e Flickr. Nesse trabalho, dados de sensores e de voluntários são mapeados a modelos semânticos e, então, uma camada é criada a partir disso para receber dados de ambas as fontes e fornecê-los a clientes por meio de consultas semânticas. Como os dados de sensores são mapeados a

modelos semânticos diferentes dos modelos usados pelos dados de voluntários, os dados em si não são interoperáveis, ou seja, a interoperabilidade é alcançada por meio do uso da camada semântica que faz a interface entre os dados e aplicações clientes. Desse modo, a integração é feita para se responder uma consulta de forma a uma fonte complementar a outra, porém, o modelo não permite a seleção e aplicação igualitária de um método de integração sobre todos os dados. Para se incorporar um método de integração à camada semântica, é preciso que o responsável pelo método saiba diferenciar como ele será aplicado aos dados de cada fonte. Assim, apesar de o modelo proposto também não estar detalhado como uma arquitetura de software, esse trabalho pode ser usado como uma camada de aquisição de dados para posterior processamento.

Em quatro trabalhos (SCHNEBELE; CERVONE, 2013; SCHNEBELE *et al.*, 2014; SCHNEBELE; CERVONE; WATERS, 2014; SCHNEBELE *et al.*, 2015) os autores propuseram uma metodologia para estimar danos de inundação integrando dados oficiais e não oficiais. Os dados oficiais incluíram mapas, modelo digital de elevação e dados meteorológicos e hidrológicos de agências governamentais e institutos de pesquisa. Dados não oficiais incluíram fotografias classificadas via crowdsourcing e vídeos manualmente classificados indicando a severidade do dano de inundações. Os dados dessas fontes foram transformados em camadas e então unidos em uma única camada de forma que uma complementasse a ausência de informação das demais. Nesse processo, foi usado um ou mais algoritmos de geração de estimativas baseados em pontos amostrais, como a estimativa de densidade kernel ou a interpolação kriging. Como o processamento feito sobre os dados de voluntários é manual, a replicação desse método para considerar novos dados torna-se custosa em relação ao tempo. Além disso, esses trabalhos não consideram a produção contínua de dados, por exemplo o envio de dados atualizados por sensores ou voluntário ao longo do tempo. Ainda, o processamento feito sobre os dados de ambas as fontes é específico para se aplicar certos algoritmos, dificultando a alteração desse algoritmo para se obter informações diferentes. Finalmente, o foco desses trabalhos é a metodologia desenvolvida para se realizar a integração dos dados, não sendo apresentada uma arquitetura de software que suporte tal metodologia.

Wan *et al.* (2014) propuseram um *framework* para coletar e gerenciar dados de autoridades e de crowdsourcing relacionados a inundações com suporte a análises estatísticas e visualização gráfica. Na implementação desse *framework*, os dados de autoridades foram extraídos principalmente de bancos de dados, relatórios e imagens de sensores de acesso público, o que envolveu atividades manuais. Os dados de crowdsourcing são obtidos por meio de um formulário online com campos de preenchimento para as mesmas informações que os dados de autoridades compreendem, como local, data e severidade de uma inundação. Nesse trabalho apenas se considera o evento de uma inundação em andamento ou passada, portanto não inclui dados sobre um rio fora do seu estado de transbordamento, por exemplo seu nível de água. Ainda, os dados de autoridades não são

coletados de forma contínua, não considerando a produção de observações de sensores ao longo do tempo, além de as atividades manuais poderem ser morosas, dificultando sua replicação. Esse trabalho também não oferece a possibilidade de se aplicar algoritmos variados para integrar os dados de ambas as fontes. Por fim, não são detalhadas as especificações do *framework* como em uma arquitetura de software.

Meissen e Fuchs-Kittowski (2014) propõem uma arquitetura de referência para sistemas de alerta antecipado que integra componentes de aplicações de crowdsourcing, como recrutamento e gerenciamento de participantes. Esse trabalho apresenta em alto nível de abstração os aspectos gerais dessa integração, indicando quais funcionalidades os componentes arquiteturais devem possuir. Por exemplo, os autores mencionam que o processamento dos dados de crowdsourcing deve ser adaptado ao contexto em que serão utilizados. Também são apresentados exemplos de uso de dados de crowdsourcing em que a integração em um sistema de alerta é feita manualmente. Nesse trabalho não são detalhadas as comunicações entre os componentes arquiteturais, o processamento realizado nas funcionalidades identificadas e os dados envolvidos, dificultando a implementação de um sistema baseado nessa arquitetura. A arquitetura para a AGORA-IFM proposta neste projeto especifica essas informações de modo que ela possa ser avaliada e implementada, evidenciando o processo de integração realizado ao nível dos dados de sensores e de voluntários. Além disso, este projeto inclui apenas dados de voluntários quanto a observações de fenômenos naturais providos de uma aplicação de crowdsourcing, não sendo ele em si uma aplicação de crowdsourcing e também não sendo necessários componentes como de recrutamento e gerenciamento de participantes. Por fim, o processo de integração definido na arquitetura para a AGORA-IFM é automatizado, ou seja, não necessita de atividades manuais, o que é uma das principais contribuições deste projeto de pesquisa.

Os trabalhos relacionados à integração de dados de sensores e de voluntários já desenvolvidos apresentam diversas características presentes na proposta deste projeto. Entretanto, nenhum trabalho compreende um processo de integração que seja automatizado para que possa ser executado periodicamente a fim de coletar dados produzidos continuamente e produzir informações atualizadas ao longo do tempo. Além disso, nenhum trabalho possibilita a utilização de diversos métodos de integração espacial sobre um mesmo conjunto de dados interoperáveis englobando dados de sensores e de voluntários. Por fim, os trabalhos da área também não apresentam uma arquitetura de software que detalhe as especificações dos componentes envolvidos e das comunicações desses componentes. Esta pesquisa de mestrado visa preencher essas lacunas com uma arquitetura de software que também possibilita a criação de um processo de integração que permite uma combinação variada de algumas atividades e a inclusão de novas atividades por meio de serviços web.

2.7 Considerações Finais

Neste capítulo foram apresentados os principais conceitos envolvidos com esta pesquisa. Primeiramente, foi apresentado o conceito sobre o qual este trabalho é fundamentado, o de arquitetura de sistemas de software e, mais detalhadamente, o padrão de arquitetura orientada a serviços e métodos de avaliação arquitetural. Em seguida, foram apresentados os conceitos relacionados ao contexto em que a arquitetura proposta está inserida, ou seja, a gestão de desastres naturais. Mais especificamente, foram apresentados como são as informações geográficas provindas de sensores e de voluntário, e como elas podem ser integradas, principal atividade envolvida nesta pesquisa. Mostrou-se ainda como trabalhos relacionados estão tratando a integração de dados geográficos e quais lacunas esses trabalhos não abordam, notadamente a falta de um processo de integração automatizado, que agiliza o processo de tomada de decisão, a possibilidade de utilização de diversos métodos de integração em um mesmo processo e a definição de uma arquitetura de software que ofereça suporte a essa integração. O capítulo seguinte apresenta a arquitetura de software desenvolvida neste trabalho proposta para abordar tais lacunas.

AGORA INFORMATION FUSION AND MANAGEMENT

Neste capítulo é apresentado o projeto da arquitetura proposta nesta pesquisa de mestrado.

3.1 Considerações Iniciais

A arquitetura orientada a serviços chamada AGORA *Information Fusion and Management* (AGORA-IFM) foi proposta para apoiar a integração de dados de sensores e voluntários por diferentes métodos espaço-temporais de forma automática e facilmente atualizável, tornando o processo de integração mais rápido e flexível. Seus resultados permitem aumentar a concentração espacial e temporal de informações relevantes à gestão de desastres naturais.

Essa arquitetura usa o AGORA-DSM e o AGORA-VOS para obter, respectivamente, dados de sensores e de voluntário. Além disso, ela utiliza um motor de funções geoestatísticas espaço-temporais para isolar o algoritmo de integração, tornando fácil a alteração desse algoritmo e permitindo a integração automática dos dois tipos de dados anteriores. Os resultados do processo de integração são armazenados em um banco de dados geográfico e exportados via *Web Map Service* (WMS). A criação da arquitetura baseou-se numa metodologia já definida, a qual é apresentada a seguir.

3.2 Metodologia

A metodologia empregada no desenvolvimento da arquitetura foi baseada no processo iterativo proposto por [Gorton \(2011\)](#). Esse processo é composto por três etapas: definir os requisitos arquiteturais, que consiste em criar uma declaração dos requisitos

que irão guiar o projeto arquitetural, desenvolver o projeto da arquitetura, que consiste em definir as estruturas e responsabilidades dos componentes arquiteturais, e validação, que consiste em testar a arquitetura, tipicamente comparando as decisões de projeto com os requisitos. Uma vez proposta e validada uma arquitetura, pode-se verificar que requisitos ou partes do projeto arquitetural precisam ser aprimorados. Isso implica modificações nos artefatos produzidos até então e, conseqüentemente, uma nova validação, o que se repete iterativamente até que o projeto arquitetural atenda suficientemente bem todos os requisitos (GORTON, 2011). A Figura 5 ilustra esse processo.

Figura 5 – Processo de projeto de arquitetura em três etapas (adaptada de Gorton (2011)).

A arquitetura proposta neste trabalho teve sua elaboração baseada nesse método iterativo. Em cada iteração, realizava-se as cinco atividades seguintes: análise de atributos de qualidade, definição dos componentes, definição de um fluxo de processamento, definição da comunicação dos componentes e validação. Inicialmente, a primeira atividade identificou atributos de qualidade de forma genérica, como modularidade e interoperabilidade. Com esses atributos em mente, elaborou-se um conjunto inicial de componentes em alto nível de abstração, o que incluiu uma análise de possíveis métodos de integração espacial. Depois disso, definiu-se um possível fluxo de processamento encadeando esses componentes para gerar um processo de integração. Em seguida, foi feita uma análise sobre como deveria ser a comunicação dos componentes para que as trocas de mensagens fossem adequadas, o que incluiu uma análise de dados de sensores, de voluntários e dos padrões da área, que se mostraram bastante diversificados. Por fim, realizou-se uma validação desse conjunto inicial de atributos, componentes, processo e interfaces por meio de reanálises a trabalhos relacionados e consultas a especialistas da área, principalmente o orientador deste projeto. Com base nas observações colhidas com a validação, essas etapas eram então retomadas em um novo ciclo de iteração, em uma ordem não necessariamente igual à anterior dependendo das necessidades identificadas, exigindo mais ou menos esforço em cada atividade de acordo com o grau de detalhamento alcançado. Assim, o desenvolvimento da arquitetura foi encerrado quando todas as características que

a compõem alcançaram uma qualidade satisfatória. A validação final foi feita por uma avaliação arquitetural, a qual atualizou os requisitos e apontou melhorias no projeto. Tais melhorias foram incorporadas após a finalização da avaliação. Nas seções a seguir são apresentados os requisitos usados no desenvolvimento da arquitetura (portanto antes de serem atualizados) e o projeto arquitetural final, já com as melhorias incorporadas. A avaliação arquitetural, incluindo os requisitos finais atualizados, são detalhados no [Capítulo 4](#).

3.3 Requisitos Arquiteturais

Ao final do desenvolvimento da arquitetura, obteve-se um conjunto de nove requisitos, apresentados no [Quadro 3](#). A maioria dos requisitos mais importantes estão relacionados à flexibilidade do processo de integração, como a definição de componentes autônomos, adição de novos componentes, encadeamento alternativo de componentes e troca do script de integração.

Quadro 3 – Lista de requisitos arquiteturais

Cada serviço deve ser independente (distribuível) em sua funcionalidade
A arquitetura deve permitir a elaboração de um processo diferente do preestabelecido por meio do uso individual dos serviços oferecidos.
A arquitetura deve permitir a troca do script de integração com pouco esforço (sem exigir conhecimento da implementação do processo).
O sistema deve permitir a substituição e adição de novos serviços (p. ex., processamento ou coleta de novas fontes de informação) com pouco esforço.
A arquitetura deve usar padrões reconhecidos para trocas de mensagens
A arquitetura deve ser passível de ser implementada em diferentes ambientes (p. ex., Linux, Windows, etc.).
O tempo de resposta deve ser baixo (tempo que um usuário pode esperar sem considerar que houve falha no sistema).
A arquitetura deve suportar a quantidade de dados de sensores e voluntários prevista para uma cidade de grande porte.
A arquitetura deve suportar a quantidade de conexões prevista para uma cidade de grande porte.

A lista completa de requisitos foi analisada e atualizada durante a avaliação arquitetural. Por meio desta, quatro novos requisitos foram identificados e todo o conjunto foi priorizado. Desse modo, a versão final da lista de requisitos foi gerada apenas após a avaliação da arquitetura e, por esse motivo, todos os treze requisitos resultantes estão descritos detalhadamente na subseção [Resultados](#) do capítulo a respeito da avaliação.

3.4 Projeto da Arquitetura

A arquitetura desenvolvida neste projeto é descrita por quatro visões: a primeira corresponde à visão de caso de uso, a segunda à visão de processo, a terceira à de desenvolvimento e a quarta corresponde à visão lógica. Cada uma dessas visões é detalhada a seguir. Outras informações que podem contribuir para o entendimento da arquitetura, como cenários de utilização do sistema alvo, são apresentadas no [Capítulo 4](#) sobre a avaliação arquitetural.

3.4.1 Visão de Caso de Uso

A arquitetura proposta compreende oito principais funcionalidades. Todas elas podem ser usadas independentemente por quaisquer aplicações, bastando estas formatarem seus dados da forma esperada pelas entradas de cada funcionalidade. Por outro lado, alguns dados de entrada são mais facilmente obtidos por meio do uso de outras funcionalidades da própria arquitetura, o que se deve ao fato de que todas as funcionalidades foram elaboradas de tal forma a facilitar seu encadeamento. Dessa forma, a principal funcionalidade definida pela arquitetura executa as demais de forma ordenada, gerando um fluxo de processamento responsável pela integração de dados. A possibilidade de se usar cada uma das funcionalidades independentemente foi incorporada também para permitir a elaboração de um fluxo de processamento diferente do predefinido.

As funcionalidades definidas pela arquitetura estão representadas como um diagrama de caso de uso na [Figura 6](#). O fluxo de processamento predefinido ocorre encadeando as funcionalidades na seguinte sequência: (i) agendar processo de integração, (ii) executar processo de integração, (iii) recuperar dados de sensores, (iv) recuperar dados de voluntários, (v) unificar dados, (vi) categorizar dados, (vii) executar algoritmo de integração e (viii) exportar dados. As funcionalidades (iii) e (iv) utilizam internamente a (v) para unificar seus próprios dados. As funcionalidades (iii), (iv), (vii) e (viii) utilizam aplicações externas para auxiliá-las a cumprir seus objetivos. A funcionalidade (vii) é definida como uma extensão do processo de integração para possibilitar um fluxo alternativo em que se possa pré-processar os dados como preparação para a integração. Isso possibilita que várias requisições de integração sejam realizadas em paralelo sobre o mesmo conjunto de dados, eliminando o tempo de pré-processamento que seria comum a todas as requisições e isolando a parte de integração, a qual é a mais lenta de todo o processo. Essa alternativa é discutida também no [Capítulo 4](#) sobre a avaliação arquitetural. Cada uma das funcionalidades da arquitetura é detalhada na subseção a seguir na sequência do processo completo de integração predefinido neste projeto, incluindo também o papel das aplicações externas.

Figura 6 – Diagrama de casos de uso da AGORA-IFM

3.4.2 Visão de Processo

O processo de integração definido nesta arquitetura para a AGORA-IFM é composto pelo encadeamento das funcionalidades anteriores, como representado na [Figura 7](#) segundo o *Business Process Model and Notation* (BPMN). Nessa figura, a arquitetura da AGORA-IFM corresponde a uma “piscina” (*pool*) em que cada componente corresponde a uma de suas “raias” (*lane*). Os componentes representam os serviços responsáveis pelas funcionalidades anteriormente apresentadas. Ainda, as aplicações externas com as quais os componentes interagem são representados por outras piscinas. Algumas informações, por exemplo algumas de nível de abstração próximo da implementação, como formatação de mensagens, são apresentadas nas subseções seguintes a esta.

O evento de início do processo, representado na [Figura 7](#) por um relógio, ocorre por meio do componente “Escalonador”. Esse componente dispara o fluxo de processamento periodicamente, o que é possível porque todas as atividades desse processo são automatizadas. O período de execução o qual o Escalonador usará pode ser definido pelo usuário. Assim, novos dados podem ser processados automaticamente ao longo do tempo,

permitindo a atualização dos resultados da integração. O processo então é organizado pelo componente “Orquestrador”, responsável por requisitar a funcionalidade dos demais componentes em uma sequência ordenada.

A primeira atividade solicitada pelo Orquestrador manipula os dados provindos de sensores. Essa atividade, executada pelo componente “Manipulador de dados de sensores”, interage com a aplicação externa *Sensor Observation Service* (SOS), o qual obtém seus dados de sensores da aplicação AGORA-DSM. A atividade consiste em recuperar do SOS os dados solicitados pelo Orquestrador, que correspondem àqueles inclusos em operações *getObservation*. Essa operação retorna dados de observações de um determinado fenômeno natural, como o oferecimento (*offering*), que é aquilo a que a observação de refere, por exemplo nível de água, o valor da medição de uma observação, a data e hora em que a observação foi feita, e a posição geográfica do fenômeno medido. A recuperação desses dados depende da forma como são solicitados, por exemplo, pode-se solicitar todos os dados referentes a um determinado oferecimento, ou os dados de uma certa região, ou de um determinado período de tempo, ou os dados medidos por um sensor específico, ou ainda uma combinação desses critérios. Um certo conjunto desses dados pode estar distribuído em diferentes sensores, e um sensor também pode conter diferentes conjuntos de dados. Isso pode exigir desta atividade a realização de diversas operações entre este componente e o SOS a fim de recuperar os dados solicitados. Além disso, o campo oferecimento pode conter qualquer valor oferecido pela AGORA-DSM, por exemplo, velocidade do vento, temperatura do ar, umidade, etc. Este componente permite a seleção de quaisquer oferecimentos, inclusive oferecimentos diferentes em um mesmo conjunto de observações, entretanto, como apenas os dados de voluntários referentes a nível de água são selecionados na atividade seguinte, esta também seleciona apenas observações de nível de água para que a posterior integração seja semanticamente válida. Essa atividade utiliza o serviço de unificar dados, definido por um outro componente da arquitetura, para auxiliá-la, unificando em um único documento os dados de sensores coletados a partir das diversas interações com o SOS. Por fim, esse componente também é responsável por formatar os dados resultantes em um padrão mais adequado à sua representação.

A segunda atividade é responsável por manipular informações geográficas de voluntários (VGIs) recuperadas do SOS, as quais foram coletadas pela AGORA-VOS. Executada pelo componente “Manipulador de VGI”, essa atividade também recupera os dados inclusos em operações *getObservation*. As VGIs obtidas pelo SOS, apesar de estarem no mesmo formato dos dados de sensores, apresentam diferenças significativas em relação a estes, pois esse formato é flexível e permite uma quantidade variável da maioria dos campos definidos em seu padrão. Além disso, a AGORA-VOS utiliza a plataforma Ushahidi para coletar os relatos de voluntários, e essa plataforma apresenta algumas limitações para adaptar as VGIs ao padrão dos sensores. Por exemplo, para se obter o oferecimento, o valor, a data e a posição de uma única observação são necessárias duas requisições *getOb-*

servation diferentes e complementares, devendo haver um controle para associar uma a outra e, ao mesmo tempo, para diferenciá-las de requisições de outras observações. Dessa forma, este componente, apesar de fornecer uma funcionalidade para VGIs equivalente à de sensores, apresenta uma lógica interna diferente do “Manipulador de dados de sensores” e, portanto, sua determinação como um componente exclusivo a esta atividade foi realizada para promover a modularidade e a modificabilidade da arquitetura. O oferecimento das observações de voluntários coletadas pela AGORA-VOS pode conter um conjunto predefinido de valores. Esse oferecimento é utilizado em uma posterior atividade de categorização para tornar os dados de sensores e de voluntários comensuráveis. Como a categorização utiliza uma estratégia de definir “faixas” de nível de água (detalho adiante), apenas as observações com o oferecimento referente a nível de água são selecionadas nesta atividade. Apesar de este componente permitir a seleção de observações com outros oferecimentos, estes possuem uma semântica diferente daquelas apresentadas pelas observações de sensores, mostrando-se inapropriados à categorização. Esta atividade também utiliza o serviço de unificação de dados para unificar as VGIs de diferentes observações em um único documento. Esse serviço recupera informações equivalentes às do serviço de manipulação de dados de sensores e, assim, também formata as informações resultantes no mesmo padrão que o componente anterior.

A terceira atividade, desempenhada pelo componente “Unificador de dados”, é utilizada pelo Orquestrador para unificar os dados de sensores e de voluntários, os quais haviam sido armazenados separadamente até então. Esta é a atividade que é solicitada pelos dois componentes anteriores para unificar os dados pelos quais são responsáveis, uma vez para unificar apenas dados de sensores entre si, e outra para unificar os dados de voluntários.

A quarta atividade é responsável por categorizar os dados de sensores e de voluntários, agora já unidos em um único documento. Essa atividade, desempenhada pelo componente “Categorizador de dados”, acrescenta um campo numérico às observações coletadas até o momento. Isso é necessário para que seja possível realizar uma operação de integração sobre os dados de sensores e de voluntários uniformemente. Apesar de os dados de ambas as fontes estarem no mesmo formato e sempre se referirem a níveis de água devido à seleção feita nas requisições, seus conteúdos são distintos. Para as observações de sensores, o campo valor contém um número em alguma unidade de comprimento, e o campo oferecimento indica que a medida se trata de nível de água, mas sem mencionar qual é o nível. De forma oposta, para as observações de voluntários o campo oferecimento indica diretamente um nível de água baixo, médio, alto ou transbordando, mas o campo valor contém um texto com o relato do voluntário. A extração de medida a partir do processamento desse texto não foi possível devido às grandes variações de seu conteúdo. Assim, para se realizar a categorização utiliza-se o campo valor das observações de sensores e o campo oferecimento das observações de voluntários. Para se categorizar as observações

de voluntários, atribui-se a elas um valor numérico de acordo com o nível de água relatado no oferecimento correspondente. Para as observações de sensores, a atribuição de um valor numérico é feita estimando-se o nível de água (baixo, médio, alto ou transbordando) representado pela medida do campo valor, podendo, para tanto, adotar-se diferentes estratégias. Uma estratégia possível, utilizando a altura da margem do rio e um nível usual de água, é apresentada no [Capítulo 5](#), Prova de Conceito.

Após a categorização, o Orquestrador pode executar ou não a integração de fato dos dados dependendo do fluxo de processamento selecionado pelo usuário. Como dito anteriormente, um fluxo alternativo de pré-processamento, que não executa um método de integração, disponibiliza dados já categorizados para serem usados diretamente pelo serviço de integração, possibilitando o uso de diferentes métodos em paralelo e economizando o tempo da categorização comum a todos. Caso esse fluxo alternativo seja selecionado, a atividade seguinte à de categorização é a de exportação de dados. O fluxo principal definido neste projeto inclui o serviço de integração, detalhado a seguir.

No fluxo de processamento principal, a quinta atividade corresponde à integração dos dados de sensores e de voluntários utilizando-se a categoria anteriormente estabelecida. O componente responsável por essa atividade, “Manipulador de dados de script”, interage com a aplicação externa “Motor de script” para auxiliá-lo em seu objetivo. O algoritmo de integração é executado por um motor externo à arquitetura para facilitar a sua troca, pois há muitos métodos geoespaciais que podem ser aplicados nesse contexto. A utilização de scripts, como algoritmos na linguagem R ou Python, foi estabelecida por este ser um meio muito utilizado por profissionais da área e por haver um grande número de métodos já implementados dessa forma. Essa atividade consiste em transformar os dados de sensores e de voluntários já categorizados para os tipos de dados esperados pelo motor de script selecionado e disponibilizá-los em um contexto acessível ao motor para que este aplique o script sobre esses dados. Após a execução do script, essa atividade também recupera do motor os dados resultantes e os transforma novamente nos tipos utilizados pelos demais serviços. Se o algoritmo de integração fosse implementado na mesma linguagem dos demais serviços, este componente poderia ser substituído por um serviço de execução de algoritmos geoespaciais; entretanto, a abordagem utilizada aqui, além de promover a modificabilidade da arquitetura em relação à troca do script, facilita que outros componentes sejam incorporados como “adaptadores” para novos motores.

A última atividade, realizada pelo componente “Exportador de dados”, é responsável por salvar as posições e categorias resultantes da integração, além de outras informações como data e oferecimento, em um banco de dados geográfico. A posição de cada ponto é registrada como um tipo geográfico para facilitar análises geoespaciais, muitas das quais já são oferecidas por sistemas gerenciadores de bancos de dados geográfico, por exemplo cálculo de intersecção de regiões. Esses registros podem, então, ser acessados por

outros serviços, por exemplo o *Web Map Service* que apresenta informações geográficas como mapas, para serem utilizados em outras aplicações. Detalhes da comunicação entre os diversos componentes da arquitetura são apresentados na subseção a seguir.

3.4.3 Visão de Desenvolvimento

A arquitetura proposta para a AGORA-IFM estabelece um componente para cada funcionalidade previamente identificada, como apresentado no diagrama de componentes da [Figura 8](#). O componente “Escalonador”, responsável pelo agendamento do processo de integração, disponibiliza uma interface “scheduleProcess” em que é fornecido como dado de entrada dois números inteiros, um representando o período no qual se deseja que o processo seja executado e outro representando o identificador do processo a ser agendado. Ela também oferece como dado de saída um texto (*string*) contendo o resultado do processo de integração, formado pelos dados, descritos adiante, dos demais componentes. Esse texto é formatado segundo o esquema *Sensor Web Enablement Common Data Model* (abreviado a partir daqui e nas Figuras 8 e 9 como SWEC) por ser um formato simples que possibilita a codificação de uma lista de dados sem necessitar estruturas específicas (como *Arrays* por exemplo). Além de implementar essa interface, o Escalonador requer uma outra para executar o processo o qual ele deve agendar. Essa interface, “executeProcess”, define como dado de entrada um número inteiro representando o identificador do processo a ser executado. Como dado de saída, ela define o mesmo texto de saída da interface anterior, também de acordo com o SWEC. O componente “Orquestrador” implementa essa interface de execução de processo.

A interface fornecida pelo Orquestrador, que é requerida pelo Escalonador, pode ser utilizada diretamente por uma aplicação para se realizar um processo de integração uma única vez. O parâmetro identificador de processo pode ser utilizado para se selecionar o fluxo principal de processamento ou um fluxo alternativo, ou ainda um outro fluxo que possa ser incorporado à arquitetura. O Orquestrador, responsável por organizar a execução de cada atividade em uma sequência ordenada, requer diversas interfaces para cumprir o processo de integração, as quais correspondem às funcionalidades dos demais serviços.

Uma interface do Orquestrador é implementada pelo componente “Manipulador de dados de sensores”. Essa interface, “handleSensorData”, define como dado de entrada um texto formatado segundo o padrão O&M e de acordo com a requisição (*getObservation*) do *Sensor Observation Service*. Isso é necessário para que este componente possa receber todas as informações de observações, como posição e valor; e esse formato foi definido por ser o mais utilizado nesse contexto. O dado de saída dessa interface é um texto, de acordo com o SWEC, contendo apenas as quatro informações mais relevantes ao processo de integração, que são o oferecimento, o valor, a data e a posição das observações solicitadas. Este componente, responsável por manipular os dados de observações de sensores,

Figura 8 – Diagrama de componentes da arquitetura da AGORA-IFM.

requer uma outra interface para o SOS para que este recupere os dados de um banco de dados geográfico. Essa interface, “getObservation”, requer, como dado de entrada, um texto com uma requisição *getObservation*, e, como dado de saída, um outro texto com uma mensagem *getObservationResponse*, ambos formatos de acordo com o padrão O&M. Essa mesma interface é requerida pelo componente “Manipulador de VGI”. Uma outra interface também é requerida tanto pelo Manipulador de dados de sensores quanto pelo de VGI, que é a “bindData”. Por meio dela, os componentes Manipulador de dados de sensores e de VGI unificam, respectivamente, observações de sensores com outras também de sensores e observações de voluntários com outras também de voluntários. Essa interface, implementada pelo componente “Unificador de dados”, requer três textos formatados segundo o SWEC contendo oferecimento, valor, data e posição das observações em questão, sendo dois desses textos os dados de entrada que se deseja unificar e o terceiro o dado de saída contendo todos os dados de entrada juntos. Essa mesma interface é requerida também pelo Orquestrado para unificar as observações de sensores e de voluntários, as quais são recuperadas separadamente pelos seus respectivos componentes.

O componente “Manipulador de VGI” é similar ao “Manipulador de dados de sensores”, requerendo as mesmas interfaces para recuperar e unificar observações e implementando uma interface equivalente à de manipular dados de sensores, a “handleVGI”. Essa interface requer os mesmos dados de entrada e saída da interface equivalente para sensores, entretanto, como essas duas interfaces são implementadas por componentes diferentes, assim determinados para promover a modularidade e a modificabilidade da arquitetura, é necessário que haja uma interface específica para a manipulação de VGIs. O Orquestrador utiliza essa interface para recuperar observações de voluntários e, posteriormente, as unifica com as observações de sensores.

Outra interface requerida pelo Orquestrador é a “categorizeData”, implementada pelo componente “Categorizador de dados”. Essa interface requer como dado de entrada um texto contendo observações formatadas segundo o SWEC com as quatro informações citadas anteriormente. Como dado de saída, essa interface retorna um texto com o mesmo conjunto de observações dos dados de entrada, também de acordo com o SWEC, mas agora contendo, além das quatro informações anteriores, uma quinta informação para cada observação correspondendo à categoria. Essa categoria, representando um nível de água baixo, normal, alto, ou transbordando, é utilizada para tornar o valor das observações de sensores e de voluntários comensuráveis.

Para executar o fluxo de processamento completo de integração, o Orquestrador requer uma interface para transformar os dados categorizados na forma esperada por um motor de scripts. Essa interface, “handleScriptData”, é implementada pelo componente “Manipulador de dados de script” e define como dados de entrada dois textos, um contendo o caminho do script de integração e outro, formatado segundo o SWEC, contendo

as cinco informações de cada observação obtidas até então. Como dado de saída, essa interface retorna um texto, também de acordo com o SWEC, com as cinco informações anteriores, agora de todos os pontos estimados pelo método de integração. O dado de saída dessa interface é equivalente ao dado de saída da interface “categorizeData”, do componente Categorizador de dados, sendo a única diferença a quantidade de pontos presentes no texto retornado. Caso esteja sendo executado o fluxo de processamento completo, o dado de saída das interfaces “scheduleProcess”, implementada pelo Escalonador, e “executeProcess”, implementada pelo Orquestrador, será o mesmo do dado de saída da “handleScriptData”; caso o fluxo alternativo de pré-processamento esteja em execução, o dado de saída daquelas interfaces do Escalonador e Orquestrador será igual ao dado de saída da interface implementada pelo Categorizador de dados.

Além de implementar uma interface, o Manipulador de dados de script requer a interface “executeScript” para um motor capaz de executar a integração especificada pelo script em uso. Essa interface define como dados de entrada o caminho para o script, em texto, e as cinco informações das observações, agora no formato específico exigido pelo motor utilizado. Como dado de saída, ela retorna as mesmas cinco informações anteriores de todos os pontos resultantes do processo de integração no formato específico do motor. Dessa forma, observa-se duas limitações: o Manipulador e o motor devem ser capazes de manipular uma mesma estrutura de dados e, além disso, o script fica restrito a utilizar, de cada observação, apenas os dados de oferecimento, valor, data, posição, categoria e seus eventuais derivados, apesar de que alguns métodos de integração podem ser executados utilizando apenas posição e categoria. Essas duas limitações restringem as operações que podem estar presentes no script. Assim, se o Manipulador fornece ao motor uma estrutura de dados adequada a um determinado script, a troca desse script por um outro que utilize a mesma estrutura corresponde simplesmente à substituição do script, sem alteração de qualquer parte do sistema. Entretanto, caso o novo script utilize uma outra estrutura de dados, duas alternativas são possíveis: adaptar o próprio script à estrutura já suportada; ou, então, adaptar o Manipulador ou adicionar um novo manipulador ao sistema para prover suporte a estrutura de dados adequada ao novo script.

A última interface requerida pelo Orquestrador para finalizar o processo de integração é a “exportData”. Essa interface, implementada pelo componente “Exportador de dados”, define como dado de entrada um texto formatado segundo o SWEC com as cinco informações de todos os pontos obtidos até então. Como dado de saída, é definido um valor booleano indicando se a exportação foi bem-sucedida ou não. Para efetivar a exportação, o Exportador depende da comunicação com um banco de dados geográfico, a qual é definida pela interface “commitData”. Nessa interface, os dados de entrada correspondem, para cada ponto, aos seguintes: o oferecimento e o valor, como texto, a posição, como o tipo geográfico POINT, a data e hora, como DATE, e a categoria, como número inteiro. O dado de saída também é um booleano indicando o sucesso do registro dos dados. Caso

o fluxo alternativo de pré-processamento tenha sido selecionado, em ambas as interfaces os dados de entrada compreenderão apenas as observações de sensores e de voluntários recuperadas pela AGORA-DSM e AGORA-VOS por meio do SOS. A comunicação entre todas as aplicações externas e componentes envolvidos na arquitetura é apresentada na subseção a seguir.

3.4.4 Visão Lógica

A arquitetura proposta para a AGORA-IFM relaciona-se com elementos externos a ela, compondo a plataforma AGORA. O diagrama da [Figura 9](#) ilustra a lógica de funcionamento dessa arquitetura inserida no contexto em que foi planejada para ser aplicada. Os elementos são organizados em três camadas, as quais se referem às partes apresentadas inicialmente na [Figura 1](#). As camadas correspondem às partes de aquisição, integração e aplicação da plataforma, estando a AGORA-IFM incluída na camada de integração. O funcionamento dos elementos externos à AGORA-IFM não foi definido pelo autor, além disso, as comunicações que não envolvem nenhum elemento incluído na AGORA-IFM, como aquelas entre AGORA-DSM e SOS, ou SOS e banco de dados, ou banco de dados e AGORA-DS, também não foram estabelecidas pelo autor e estão ilustradas na [Figura 9](#) apenas para representar a existência de tais comunicações.

O diagrama representa a comunicação entre elementos por uma seta, indicando uma mensagem de requisição ou de resposta, associada a um rótulo, indicando o meio de comunicação e os dados transferidos. Com exceção das comunicações entre o “Manipulador de dados de script” e o “Motor de script”, todas as demais são feitas por meio de *Hypertext Transfer Protocol* (HTTP), assim, essas comunicações podem ser instanciadas, por exemplo, utilizando-se *Simple Object Access Protocol* (SOAP). O meio de comunicação entre o Manipulador e o Motor não foi definido devido à variação de meios pelos quais um motor de scripts pode ser utilizado, por exemplo como uma biblioteca estática ou dinâmica, uma aplicação não conectada à internet (*off-line*) instalada no mesmo servidor do Manipulador, ou ainda um serviço web comum. Os dados indicados nos rótulos das setas são apresentados sem grande formalismo e de modo a facilitar o entendimento do que está sendo transferido. Os “tipos” indicados antes dos dados podem, por vezes, representar um padrão de formatação ao invés de um tipo de dado nativo. As informações estão discriminadas para explicitar o que é transferido em cada mensagem, porém, elas podem ser transferidas como um único argumento em alguma estrutura de dados. Por exemplo, a mensagem “HTTP(*string* caminho, SWEC *offering*, posição, valor, data, categoria)” é transferida via HTTP e possui apenas dois argumentos, o caminho, do tipo texto, e cinco informações de uma ou mais observações, todas registradas em um único argumento, também do tipo texto, o qual é formatado de acordo com o padrão *Sensor Web Enablement Common Data Model*. Os tipos e quantidades exatas dos argumentos de

cada mensagem são apresentados no diagrama de componentes da [Figura 8](#) e detalhados na subseção anterior, Visão de Desenvolvimento.

A camada inferior do diagrama [9](#), Camada de Aquisição, é responsável por obter, centralizar e disponibilizar informações de observações, tanto de sensores quanto de voluntários. Ela compreende quatro elementos: AGORA-DSM, AGORA-VOS, SOS, e um banco de dados. A aplicação AGORA *Dynamic Sensor Management* (DSM) é responsável por coletar observações de sensores, e a AGORA *Volunteered Observation Service* (VOS) por coletar observações de voluntários. Ambas as aplicações utilizam o *Sensor Observation Service* (SOS) para inserirem suas observações em um banco de dados, o qual centraliza observações ainda não processadas. O SOS também é utilizado, por exemplo por componentes da AGORA-IFM ou pela AGORA-DS, para recuperar as observações desse banco.

Na camada intermediária, Camada de Integração, encontra-se a aplicação AGORA *Information Fusion and Management* (IFM) e dois elementos utilizados por esta. A AGORA-IFM compreende oito serviços, cujas interações são detalhas nas subseções anteriores deste capítulo. Por outro lado, essa visão permite algumas verificações como, por exemplo, as mensagens trocadas entre o Manipulador de dados de sensores e o SOS são equivalentes às trocadas entre o Manipulador de VGI e o SOS, assim como as mensagens trocadas entre esses dois manipuladores e o Orquestrador. O mesmo ocorre com as mensagens trocadas entre esses manipuladores e o Unificador de dados, e entre este e o Orquestrador. Ainda, as respostas do Categorizador e do Manipulador de dados de script para o Orquestrador são equivalentes com exceção à quantidade de observações retornadas. Uma dessas respostas, dependendo do fluxo de processamento selecionado, será exatamente igual à requisição do Orquestrador para o Exportador de dados, às respostas do Orquestrador para a AGORA-DS e para o Escalonador, e à resposta do Escalonador para a AGORA-DS. Por fim, as requisições da AGORA-DS e do Escalonador para o Orquestrador também são iguais. Os dois elementos da camada intermediária que não pertencem à AGORA-IFM são a aplicação Motor de scripts e um banco de dados. O motor, responsável por executar scripts sobre os dados selecionados no processo de integração, possui comunicações específicas que dependem do motor instanciado. O banco de dados armazena os dados resultantes da integração, e pode ser acessado por aplicações externas como a AGORA-DS.

A camada superior, Camada de Aplicação, utiliza os dados das camadas anteriores para gerar mapas e indicadores para os tomadores de decisão envolvidos no contexto dessa plataforma. Ela compreende a AGORA *Decision Support* (DS), que acessa o SOS, o banco de dados processados e os serviços do Orquestrador e Escalonador. Apesar de não configurar um fluxo de processamento definido neste projeto, aplicações desta camada também podem acessar diretamente os serviços dos demais elementos da AGORA-IFM.

3.5 Validação

Como mencionado anteriormente, a validação final foi feita por meio de uma avaliação arquitetural, na qual foi utilizada uma adaptação do *Architecture Tradeoff Analysis Method* (ATAM). A condução e os resultados dessa avaliação são apresentados no capítulo seguinte.

3.6 Considerações Finais

Neste capítulo foi apresentada a AGORA *Information Fusion and Management*, AGORA-IFM, uma arquitetura de software para integrar informações de sensores e voluntários no contexto de inundações. Trabalhos relacionados a integração de dados geográficos não estabelecem um processo totalmente automatizado e que possibilite usuários escolherem o método de integração. Essas características correspondem às principais contribuições desta pesquisa, que também defini comunicações com aplicações externas para coletar informações, processá-las e apresentá-las.

A arquitetura proposta descreve componentes, interconectados de forma padronizada, para manipular, categorizar, integrar e disponibilizar informações relevantes à gestão de risco de inundações. Essa arquitetura foi avaliada por meio de um método de avaliação arquitetural, detalhado no próximo capítulo. Além disso, ela também foi implementada e seu comportamento foi verificado por meio de um experimento, apresentado no [Capítulo 5](#).

AVALIAÇÃO ARQUITETURAL

Neste capítulo é apresentada a avaliação da arquitetura proposta para a AGORA-IFM, incluindo a forma como foi conduzida, seus resultados e uma discussão sobre esses resultados.

4.1 Condução

O método de avaliação arquitetural *Architecture Tradeoff Analysis Method* (ATAM) foi adaptado para ser realizado de forma mais rápida em virtude da dificuldade de se reunir em um mesmo horário e local, por mais de um dia, as pessoas que pudessem desempenhar os papéis mais indicados no contexto da AGORA-IFM. Essa adaptação consistiu em criar, antes da avaliação, uma versão inicial dos seguintes documentos:

- Lista de abordagens arquiteturais utilizadas pelo arquiteto.
- Árvore de utilidade, contendo os atributos de qualidade e requisitos do sistema considerados mais importantes.
- Lista de cenários arquiteturais.

Esses documentos foram utilizados durante a avaliação e refinados pelos participantes ao longo das etapas do método, tendo itens removidos, acrescentados ou alterados de acordo com as discussões das reuniões. Além desses documentos, fichas de avaliação das abordagens arquiteturais foram desenvolvidas para agilizar o processo de análise e registro recomendadas pelo método. Todos esses documentos foram elaborados em conjunto pelo arquiteto e o líder e o escrevente do time de avaliação, e validados pelo gerente de projeto. Os três autores dos documentos conduziram duas avaliações piloto antes da avaliação propriamente dita, uma referente à primeira fase e outra referente à segunda,

para se familiarizarem com o método e corrigir eventuais problemas nos documentos ou na condução do método.

A avaliação da AGORA-IFM contou com os seguintes participantes: na primeira fase, participaram, no time de avaliação, o líder, o escrevente e três questionadores; e no time de projeto, o arquiteto/desenvolvedor, o gerente de projeto, e um integrante do grupo AGORA. Na segunda fase, participaram, no time de avaliação, o líder, o escrevente e três questionadores; no time de projeto, o arquiteto/desenvolvedor, o gerente de projeto, e dois integrantes do grupo AGORA; e como demais partes interessadas dois integrantes do Programa de Pós-Graduação em Engenharia Hidráulica e Saneamento (PPG-SHS) da Escola de Engenharia de São Carlos (EESC) da Universidade de São Paulo. Os participantes foram selecionados de acordo com o conhecimento e experiência relacionados aos papéis aos quais foram atribuídos. As discussões de ambas as fases foram gravadas em áudio, com o consentimento dos participantes, para ajudar na utilização dos comentários levantados.

A arquitetura descrita no Capítulo 3 apresenta alterações, em relação à arquitetura avaliada, efetuadas para tratar riscos identificados após a avaliação. As principais alterações estão relacionadas à determinação do uso do padrão SWEC e de um fluxo alternativo de pré-processamento. Dessa forma, a condução e os resultados apresentados a seguir referem-se à arquitetura sem tais alterações, as quais são destacadas na discussão dos pontos em que ocorreram. Observa-se ainda que uma nova avaliação deve ser realizada a fim de verificar se essas alterações são adequadas e suficientes para tratar os riscos a que se referem.

Com a utilização da documentação previamente elaborada, a primeira fase da avaliação ocorreu em apenas um período de um dia. Ela seguiu as seis primeiras etapas do ATAM na ordem sugerida: apresentação do método, do contexto do projeto, da arquitetura, identificação das abordagens arquiteturais, geração da árvore de utilidade e análise das abordagens. Os nós da árvore de utilidade foram ramificados em atributos de qualidade de acordo com suas características, no primeiro nível, e sub-características, no segundo nível, estabelecidas pela *International Organization for Standardization* e *International Electrotechnical Commission* (padrão ISO/IEC 25010). Com isso, obteve-se um conjunto de quatro atributos e oito sub-atributos de qualidade ramificados em doze requisitos priorizados, dos quais sete foram analisados, registrando-se seus pontos sensíveis, riscos, não riscos e de contrabalanço. Todas as nove etapas da segunda fase ocorreram em um período de um único dia ao invés de em dois dias como sugerido pelo método devido ao auxílio dos documentos predefinidos e dos relativos à primeira fase. As seis primeiras etapas foram recapituladas e então as três últimas foram conduzidas, seguindo a ordem apresentada anteriormente. Assim, atualizou-se e produziu-se os documentos finais, acrescentando-se aos anteriores um novo requisito em um atributo já identificado, analisando-se também

esse novo requisito; e uma lista de cenários priorizados, um para cada requisito, totalizando treze cenários, dos quais quatro foram analisados. Por fim, elaborou-se um resumo dos pontos fortes e fracos da arquitetura, contando com oito pontos sensíveis, três de contrabalanço, cinco riscos e três não riscos. Os resultados da avaliação são apresentados em detalhe na subseção a seguir.

4.2 Resultados

O primeiro artefato resultante da avaliação é a lista de decisões arquiteturais. Cinco decisões foram identificadas para satisfazer os requisitos de qualidade. A primeira delas, D1, foi seguir o padrão de arquiteturas orientadas a serviços para promover o baixo acoplamento, ou seja, facilitar a criação de componentes com funcionalidades independentes, projetadas como serviços. Isso é importante para que os serviços possam ser utilizados individualmente ou para a criação de um encadeamento diferente daquele predefinido, atividades comuns no contexto de sistemas de informação geográficos. Essa modularidade também facilita a implantação de novos serviços, promovendo a evolução da arquitetura. Além disso, essa decisão também foi tomada para facilitar a comunicação entre seus componentes internos e com aplicações externas, promovendo sua interoperabilidade. Isso é importante para que ela possa ser complementada com novas funcionalidades e se adaptar ao caráter complexo e dinâmico dos desastres naturais.

A segunda decisão, D2, foi a de disponibilizar publicamente os principais serviços, ou seja, os oito serviços apresentados como componentes no diagrama 8. Isso visa possibilitar a utilização individual desses serviços por aplicações externas, promovendo a reusabilidade da arquitetura.

A terceira decisão, D3, foi a de separar dos componentes arquiteturais o *script* de integração. Tornar o *script* independente dos serviços é importante porque há muitos métodos de integração e sua alteração pode ser frequente. Além disso, as pessoas envolvidas com a utilização desse tipo de método muitas vezes não têm conhecimento técnico em serviços web, portanto, além de promover a modificabilidade, essa separação facilita o uso da arquitetura por essas pessoas.

A quarta decisão, D4, foi a de que a arquitetura deve ser implantada em nuvem computacional. Hospedar seus serviços em nuvem promove o desempenho do sistema por possibilitar elasticidade na alocação de recursos computacionais, além de garantir alta disponibilidade e tolerância a falhas de hardware de acordo com o contrato de hospedagem estabelecido.

A última decisão, D5, foi a de encapsular a formatação das mensagens trocadas pelos componentes em um serviço. Essa decisão visa unificar o ponto em que as informações tratadas em cada serviço são estruturadas segundo algum padrão, por exemplo o

O&M, e, consequentemente, facilitar a alteração do padrão adotado, promovendo a interoperabilidade da arquitetura. O [Quadro 4](#) apresenta um resumo das decisões arquiteturais adotadas e os respectivos atributos de qualidade a que estão relacionadas. Um identificador da forma “R#” indica a quais requisitos a decisão correspondente está associada.

Quadro 4 – Lista de decisões arquiteturais

Id.	Decisão arquitetural	Atributo(s) de qualidade	Descrição
D1	Arquitetura orientada a serviços (R9, R1, R4)	Interoperabilidade, Modularidade, Modificabilidade	Os serviços foram divididos de forma que suas funcionalidades fossem independentes de acordo com as atividades identificadas no processo de integração.
D2	Disponibilização pública dos principais serviços (R2)	Reusabilidade	Os principais serviços foram disponibilizados publicamente para permitir a utilização individual em diferentes contextos.
D3	Separação do <i>script</i> de integração de dados dos demais serviços (R3)	Modificabilidade	O <i>script</i> de integração é executado a partir de um arquivo independente em relação ao código de implementação. Caso o formato das entradas e saídas do <i>script</i> sejam iguais às do serviço que o executa, a alteração do <i>script</i> exigirá pouco esforço.
D4	Hospedagem em nuvem (R5, R10, R12, R13)	Disponibilidade, Capacidade, Comportamento temporal	A arquitetura é implantada em nuvem, o que garante alta disponibilidade e tolerância a falhas de hardware.
D5	Encapsulamento da formatação das mensagens em um serviço (R8)	Interoperabilidade	O encapsulamento da formatação de mensagens em serviço facilita a implantação de novos padrões.

O segundo artefato da avaliação arquitetural foi a árvore de utilidade de atributos de qualidade, que resultou em treze requisitos. Nos casos em que um requisito está relacionado a mais de um atributo de qualidade, tal requisito foi adicionado ao nó referente ao atributo de que mais se aproxima, porém, essa classificação não é rígida e pode ser atualizada. Alguns requisitos se referem a características como “pouco esforço” ou “pouco tempo”. Isso foi especificado para os casos em que se chegou a um consenso, durante as discussões, de quanto seria seu valor para um determinado requisito; porém, em outros casos, essa característica permaneceu em alto nível por não ter havido concordância sobre seu valor devido à diversidade de situações que podem estar envolvidas. Mesmo assim a avaliação pôde prosseguir com resultados satisfatórios. Os requisitos analisados, ou seja, aqueles classificados com prioridades alta e média, são detalhados na subseção Discussão.

A Figura 10 mostra a árvore de utilidade que resultou do processo de avaliação. O nó inicial, “Utilidade”, foi suprimido para reduzir espaço.

Figura 10 – Árvore de utilidade de atributos de qualidade da AGORA-IFM.

Os primeiros quatro requisitos referem-se à característica manutenibilidade. O primeiro, R1, relacionado a modularidade, diz que cada um dos principais serviços deve ser distribuível e autocontido. Ele foi classificado como tendo alta importância para o sucesso do sistema devido às características buscadas com as decisões arquiteturais, e baixa dificuldade de obtenção por já ter sido alcançado. R2, relacionado a reusabilidade, diz que deve ser possível elaborar-se um processo diferente do processo de integração predefinido por meio do encadeamento dos serviços, por exemplo, para utilizar-se apenas dados de sensores. Foi classificado com alta importância e dificuldade média de obtenção por os serviços não poderem ser usados em qualquer ordem. R3 e R4 estão relacionados a modificabilidade. R3 diz que deve ser possível trocar o *script* de integração sem exigir conhecimento da implementação dos serviços. Teve alta importância e alta dificuldade de obtenção porque o serviço responsável por interagir com o *script* defini uma interface de entradas e saídas que é adequada a alguns mas não a todos os *scripts* possíveis. R4 diz que deve ser possível substituir e adicionar novos serviços com pouco esforço. Foi classificado com alta importância e alta dificuldade de obtenção porque, à época da avaliação, nem todas as mensagens trocadas pelos serviços haviam sido padronizadas.

Os três próximos requisitos foram identificados durante a primeira fase da avaliação, e estão relacionados a confiabilidade, sendo R5 e R6 relacionados a disponibilidade. R5 diz que o sistema deve ter alta disponibilidade de hardware, e foi classificado com baixa importância e baixa dificuldade de obtenção por a arquitetura ser implantada em nuvem computacional (decisão tomada para atender R10). R6 diz que o sistema deve ter alta disponibilidade de software, e foi classificado com alta importância e alta dificuldade pois nenhuma decisão foi tomada para se satisfazer esse requisito. Isso ocorreu porque ele foi identificado após a elaboração da arquitetura, durante o processo de avaliação. R7, relacionado a recuperabilidade, diz que, em caso de falha de software, o sistema deve realizar um processo alternativo com processamento mínimo para tentar evitar que ele deixe de responder. Foi classificado com alta importância e alta dificuldade por não ter havido decisão para tratar tal requisito. Isso também ocorreu porque ele ainda não havia sido identificado até o momento, porém, após a avaliação, acrescentou-se um fluxo alternativo para satisfazê-lo. Os dois requisitos seguintes estão relacionados à sub-característica interoperabilidade da característica compatibilidade. R8 diz que arquitetura deve usar padrões reconhecidos para trocas de mensagens. Foi classificada com alta importância e alta dificuldade porque, em certos casos, o padrão mais adequado é de difícil adoção devido à sua complexidade, como o *Web Processing Service* (WPS). R9 diz que arquitetura deve ser passível de ser implementada em diferentes ambientes (p. ex., Linux e Windows), e foi classificada com baixa importância e dificuldade devido à adoção de SOA.

Os quatro últimos requisitos estão relacionados a desempenho, sendo R10 e R11 relacionados a comportamento temporal. R10 diz que o tempo de resposta do processo de integração não deve exceder o tempo que um usuário pode esperar sem considerar que houve falha no sistema. Foi classificado como importância média e dificuldade de obtenção alta porque certos casos não exigem um tempo de resposta muito curto, mas certos algoritmos espaço-temporais são inerentemente demorados. R11 foi o único requisito acrescentado durante a segunda fase da avaliação devido à identificação do cenário C11, e diz que a arquitetura deve permitir o armazenamento de dados pré-processados (categorizados) para serem usados diretamente pelo serviço de integração. Isso foi levantado como uma proposta de se separar o processamento da integração de dados do restante do processo, além de permitir que vários métodos de integração sejam executados em paralelo sem a necessidade de se executar a parte inicial do processo que seria comum a todos os métodos. Esse requisito foi classificado como tendo alta importância para o sucesso do sistema e média dificuldade de obtenção, pois, apesar de ser um requisito identificado apenas durante a avaliação, pôde-se atualizar a arquitetura para comportá-lo. Os dois últimos requisitos estão relacionados a capacidade e dizem que a arquitetura deve suportar a quantidade de dados (de sensores e voluntários), no caso do R12, e a quantidade de conexões, no caso do R13, previstas para uma cidade de grande porte. Ambos foram classificados com média importância e baixa dificuldade de obtenção devido à possibilidade

de contratação de recursos de uma nuvem computacional.

O terceiro artefato da avaliação foi a lista priorizada de cenários arquiteturais, havendo um cenário para cada requisito. C11 foi o cenário identificado durante a avaliação e que originou o requisito R11. Todos os cenários representam situações ideais de funcionamento do sistema do qual a arquitetura é base, e, portanto, não necessariamente a arquitetura os suporta da forma como estão descritos. Os cenários são usados justamente para verificar os pontos que são e os que não são atendidos pela arquitetura. A seleção de cenários para análise foi estabelecida por consenso entre os participantes e considerou os cenários com três ou quatro votos, porém, como os requisitos correspondentes também haviam sido selecionados, tais cenários já haviam sido analisados. O [Quadro 5](#) apresenta o conjunto de todos os cenários com os votos que cada um recebeu. Os cenários analisados, juntamente com os demais requisitos prioritários, estão detalhados na subseção Discussão.

4.3 Discussão

Os oito requisitos analisados foram R2, R3, R4, R6, R7, R8, R10 e R11, e os quatro cenários foram C2, C4, C6 e C7. A análise de um requisito e seu cenário correspondente foi feita em conjunto, como sugerido pela ficha de análise do [Quadro 1](#). Todos os pontos da arquitetura aos quais esses requisitos e cenários se referem foram considerados pontos sensíveis por serem de alta prioridade e estarem relacionados a pelo menos um atributo de qualidade. As fichas de análise de todos os requisitos selecionados, incluindo todos os pontos sensíveis, de contrabalanço, riscos e não riscos identificados, encontram-se no [Apêndice A](#).

O requisito/cenário R/C2 refere-se ao uso individual dos serviços e elaboração de um encadeamento diferente, e está relacionado à decisão D2 de disponibilizar os principais serviços publicamente. Esse ponto sensível foi considerado um não risco porque viabiliza o cumprimento de seu objetivo e não envolve problemas por as fontes de dados relacionadas também serem públicas.

R/C3 refere-se à troca do script de integração e relaciona-se à decisão D3 de separar o script dos demais serviços. Para esse ponto identificou-se dois pontos de contrabalanço. O primeiro diz que essa decisão pode afetar o desempenho porque se o algoritmo de integração fosse interno à arquitetura, mesmo comprometendo a modificabilidade, seria mais rápido. O segundo diz que a interoperabilidade é prejudicada pois apenas uma interface de entradas e saídas foi definida, o que pode exigir um grande esforço para se usar scripts com interfaces diferentes. Entretanto, esse ponto foi considerado como um não risco pois a decisão tomada se mostrou a melhor alternativa para se alcançar o requisito em questão. Duas sugestões foram levantadas para abordar esse ponto, uma de aumentar o encapsulamento da interface e outra de utilizar algum padrão desse contexto, como o

Quadro 5 – Lista de cenários arquiteturais

Id.	Cenário	Votos
C1	Um mantenedor do sistema divide os serviços em diversos ambientes e os serviços continuam funcionando normalmente sem necessidade de alterar seus códigos.	2
C2	Um usuário utiliza os serviços individualmente para atender seus objetivos.	3
C3	Um usuário substitui/altera o <i>script</i> de integração modificando o arquivo correspondente com pouco esforço.	1
C4	O mantenedor do sistema substitui e adiciona novos serviços (p. ex. processamento ou coleta de novas fontes de informação, tratamento da qualidade da informação dos voluntários) com pouco esforço.	4
C5	Ocorrendo uma falha de hardware no servidor, o processo de integração é cancelado e a falha é tratada automaticamente pela infraestrutura de nuvem.	1
C6	Mesmo em situações extremas, o sistema deve ter alta disponibilidade.	3
C7	Mesmo em caso de uma falha de software, o processo de integração é executado e são fornecidas informações mínimas para a tomada de decisão.	3
C8	Quando o processo de integração é iniciado, as trocas de mensagens entre os serviços usam padrões reconhecidos.	1
C9	O mantenedor decide implantar os serviços em diferentes ambientes (Linux, Windows, etc.), e o resultado é fornecido igualmente em todos eles.	0
C10	Quando uma aplicação requisita a integração de dados, o processo de integração é executado e os resultados são fornecidos em pouco tempo.	1
C11	Quando um usuário requisita diversas integrações em paralelo sobre um mesmo conjunto de dados de entrada, usando um modelo diferente em cada integração, um conjunto de dados pré-processados (categorizados) é independentemente fornecido às requisições para ser usado diretamente pelo serviço de integração.	1
C12	Em caso de uma requisição de elevada carga de dados (volume), prevista para uma cidade de grande porte, o processo de integração é executado e os resultados são fornecidos.	0
C13	Em caso de uma elevada quantidade de requisições de dados (conexões), prevista para uma cidade de grande porte, o processo de integração é executado e os resultados são fornecidos.	0

WPS.

R/C4 refere-se à substituição e adição de novos serviços, e está relacionado à decisão D1 de adotar o padrão SOA. Esse ponto foi considerado um risco porque, até o momento da avaliação, nem todas as mensagens trocadas na arquitetura haviam sido padronizadas, o que podia dificultar a inclusão e alteração dos serviços. Isso ainda não havia sido feito devido à existência de diversos padrões estabelecidos, o que exigiu uma análise prolongada para se determinar o padrão mais adequado a cada mensagem. Porém,

após a avaliação, todas as mensagens da AGORA-IFM (com exceção às específicas de motores de scripts) foram padronizadas. Apesar de uma nova avaliação ser necessária, espera-se que essa atualização torne esse ponto um não risco.

R/C6 refere-se à alta disponibilidade de software. Nenhuma decisão foi tomada para tratar esse requisito porque ele só foi identificado durante o processo de avaliação, portanto, após a elaboração da arquitetura. Devido à falta de decisão, esse ponto foi considerado um risco, notando-se ainda que a centralização do fluxo de processamento em um único serviço orquestrador pode potencializar a ocorrência de falhas de software. Duas sugestões foram levantadas para abordar esse ponto, uma de se criar serviços redundantes para substituir serviços que se tornem inoperantes, e outra de descentralização do orquestrador, por exemplo utilizando fluxos de processamento alternativos ou uma abordagem de coreografia de serviços. Este foi o único ponto considerado risco para o qual não foi tomada uma decisão após a avaliação devido à complexidade da implantação das soluções sugeridas e à falta de tempo hábil para tanto.

R/C7 refere-se à utilização de um fluxo alternativo com processamento mínimo, em caso de falha de software, para tentar evitar que o sistema deixe de responder. Até o momento da avaliação não havia sido tomada uma decisão para tratar esse requisito porque ele foi identificado durante a avaliação e, por isso, esse ponto foi classificado como risco. A sugestão levantada para esse ponto foi a de se criar um fluxo de processamento que, por exemplo, fornecesse o resultado já armazenado de uma integração prévia, ou ainda que fornecesse os dados categorizados mas não integrados, o que já facilitaria de certa forma a tomada de decisão. Após a avaliação foi incorporada à arquitetura o fluxo de processamento do segundo exemplo, ou seja, desconsiderando a integração de dados, o que se espera poder tornar esse ponto um não risco.

R/C8 refere-se à padronização das mensagens trocadas pelos componentes da arquitetura, e está relacionado à decisão D5 de encapsular a formatação das mensagens em um serviço. Apesar de a decisão tomada facilitar a padronização, ela não é suficiente para satisfazer a interoperabilidade exigida na arquitetura. Nesse sentido, dever-se-ia determinar um padrão específico para cada mensagem, como identificado em R/C4, o que ainda não havia sido feito até o momento da avaliação. Por essa falta de padronização detalhada, esse ponto foi considerado um risco. Observou-se também que a arquitetura não prevê a evolução dos padrões em si, por exemplo quando um padrão ganha uma nova versão de si mesmo. Um ponto de contrabalanço identificado foi o de que essa decisão afeta a manutenibilidade do sistema, e uma sugestão levantada foi a de se adotar um padrão específico da área, por exemplo um determinado esquema XML. Após a avaliação, as mensagens envolvidas na arquitetura foram padronizadas e, com isso, espera-se que em uma nova avaliação esse ponto se torne um não risco.

R/C10 diz que o tempo de resposta do processo de integração deve ser baixo, e está

relacionado à decisão D4 de que a arquitetura deve ser implantada em nuvem. Esse ponto foi considerado um não risco pois pode-se contratar recursos de uma nuvem computacional que garantam o desempenho exigido pelo sistema.

R/C11 diz que a arquitetura deve permitir o armazenamento de dados pré-processados para serem usados diretamente pelo serviço de integração. Esse requisito foi identificado durante a avaliação e refere-se à possibilidade de se executar vários métodos de integração em paralelo utilizando um mesmo conjunto de dados pré-processados (categorizados), além de eliminar o tempo do início do processo que é comum a todos. Como esse requisito não havia sido identificado durante a elaboração da arquitetura, nenhuma decisão foi tomada para tratá-lo, tornando esse ponto um risco. Após a avaliação, o processo de pré-processamento sugerido foi incorporado à arquitetura, o mesmo adotado para tratar R/C7, o que pode tornar esse ponto um não risco.

Como resultado final da avaliação arquitetural, produziu-se o [Quadro 6](#) com a sumarização dos principais resultados, relacionando os requisitos analisados aos atributos de qualidade e decisões arquiteturais correspondentes, além de sua classificação como risco ou não risco. Como mencionado anteriormente, as principais decisões tomadas após a avaliação são a determinação do uso do padrão SWEC, para atender os requisitos R4 e R8, e de um fluxo alternativo de pré-processamento, para atender R7 e R11. A marcação “*” nos riscos correspondentes a esses requisitos indica que, por meio dessas novas decisões, acredita-se que o ponto correspondente se tornará um não risco em uma avaliação futura.

Quadro 6 – Resumo dos resultados

Requisito	Atributo(s) de qualidade	Decisão arquitetural	Risco	Não risco
R2. Elaboração de processos diferentes usando os serviços individualmente.	Reusabilidade	D2. Disponibilização pública dos principais serviços.		X
R3. Troca do <i>script</i> de integração.	Modificabilidade	D3. Separação do <i>script</i> de integração dos demais serviços.		X
R4. Substituição e adição de novos serviços.	Modificabilidade	D1. Adoção do padrão SOA.	X*	
R6. Alta disponibilidade de software.	Disponibilidade, Recuperabilidade	Sem decisão.	X	
R7. Processo alternativo mínimo.	Recuperabilidade	Sem decisão.	X*	
R8. Usar padrões reconhecidos.	Interoperabilidade	D5. Encapsulamento da formatação das mensagens em um serviço.	X*	
R10. O tempo de resposta deve ser baixo.	Comportamento temporal	D4. Hospedagem em nuvem.		X
R11. Integrações por vários modelos em paralelo.	Comportamento temporal	Sem decisão.	X*	

PROVA DE CONCEITO

Neste capítulo é apresentada uma prova de conceito da arquitetura proposta para a AGORA-IFM, em que o processo de integração definido por esta foi implementado e aplicado às observações de nível de água referentes a cinco sub-bacias urbanas da cidade de São Carlos, São Paulo, as quais apresentam pontos de alagamento nos períodos de chuva.

5.1 Implementação

Os componentes arquiteturais e um cliente foram desenvolvidos como serviços na linguagem de programação Java, e suas comunicações foram feitas via SOAP com auxílio do framework Axis¹. Os serviços, então, foram implantados na nuvem computacional AZURE². Nessa mesma nuvem, foram implantadas as aplicações AGORA-DSM e AGORA-VOS, desenvolvidas por membros do mesmo grupo de pesquisa do autor; além do servidor Sensor Observation Service implementado pela 52°North³.

O serviço de categorização foi implementado associando-se os números 1, 2, 3 e 4 respectivamente aos níveis de água “baixo”, “normal”, “alto” e “transbordando”. Para as observações de voluntários, isso é feito diretamente de acordo com o respectivo oferecimento. Para as observações de sensores, a categorização segue uma estratégia que utiliza dados disponíveis para quinze pontos das sub-bacias consideradas. Esses dados foram obtidos por hidrólogos e correspondem, para cada ponto, à profundidade do leito do córrego, incluindo a altura das margens de ambos os lados, e ao nível de água do córrego sem a ocorrência de chuva, chamado aqui de nível usual por simplificação. A partir desses dados, considera-se a altura de um rio como sendo a menor entre as duas margens de cada ponto,

¹ <https://axis.apache.org/> Acessado em 25/01/2016.

² <https://azure.microsoft.com/> Acessado em 25/01/2016.

³ <http://52north.org/> Acessado em 25/01/2016.

pois é ela que determina um transbordamento. Considera-se também que o nível usual está na categoria “normal”. Com isso, a definição dos níveis de água foi feita da seguinte maneira:

- O nível “baixo” varia do nível mais baixo possível (sempre zero) até o valor médio entre o nível mais baixo e o usual. Por exemplo, para um ponto em que a margem menor mede 2,213m e o nível usual é de 0,83m, essa categoria varia de 0m a $(0,83-0)/2 = 0,415\text{m}$.
- O nível “normal” varia do limite superior da categoria “baixo” até o valor médio entre o nível usual e a altura do rio. Para o exemplo anterior, varia de 0,415m a $0,83+(2,213-0,83)/2 = 1,5215\text{m}$.
- O nível “alto” varia do limite superior da categoria “normal” até a altura do rio. No exemplo, varia de 1,5215m a 2,213m.
- O nível “transbordando” inclui as medidas acima da altura do rio. No exemplo, acima de 2,213m.

De posse da faixa de medidas que cada nível compreende, para cada ponto das sub-bacias, a categorização das observações de sensores é feita atribuindo-se o número 1, 2, 3 ou 4 de acordo com a medida do campo valor da observação e do ponto a que esta se refere. Os quinze pontos utilizados são apresentados na [Figura 11](#).

O Motor de *script* selecionado foi o R 3.1.3 da GNU devido à grande quantidade de algoritmos geoespaciais disponíveis para ele. Para executar algoritmos em R a partir de código Java foi utilizado o pacote rJava⁴, que auxilia a transformação dos tipos de dados entre essas duas linguagens e disponibiliza esses dados ao motor. Os únicos pontos de contato entre os programas Java e R são as transformações de tipos dos dados de entrada e saída do *script*, tornando todo o resto independente.

O método estatístico utilizado para gerar estimativas de nível de água foi o kriging da biblioteca geoR⁵. Utilizou-se o kriging por ser uma interpolação espacial que incorpora o princípio de que, em locais próximos entre si, os valores de uma determinada característica também tendem a ser próximos, o que é válido para os níveis de água em uma região inundada. As estimativas dessa interpolação são combinações lineares ponderadas que correlacionam os pontos considerando seu arranjo espacial (STEIN, 2012). A precisão dessas estimativas será maior quanto maior o número de pontos na amostra a ser interpolada. Por outro lado, é importante ressaltar também que o foco deste projeto não é o método de integração, mas o suporte que a arquitetura oferece para integrar dados de sensores e voluntários automaticamente utilizando métodos variados.

⁴ <https://www.rforge.net/rJava/> Acessado em 28/01/2016.

⁵ <https://cran.r-project.org/web/packages/geoR/> Acessado em 28/01/2016.

Figura 11 – Pontos utilizados na categorização de observações de sensores (Zaffani e Mendiando (2012)).

Os dados resultantes do processo de integração são armazenados em um banco de dados PostgreSQL com a extensão PostGIS⁶, que adiciona suporte a objetos e consultas geográficos. A implementação da arquitetura foi usada na condução de um experimento para demonstrar a factibilidade desta e sua capacidade de apoiar um processo de integração automático. Mapas temáticos foram construídos a partir dos resultados da integração para apresentar visualmente o nível de água ao longo dos córregos considerados.

5.2 Experimento

O experimento contou inicialmente com um conjunto de três sensores reais instalados em dois pontos estratégicos selecionados por hidrólogos (HORITA *et al.*, 2014). Esses sensores enviavam suas observações para a AGORA-DSM, a qual inseria tais observações no SOS. Além disso, também foram utilizadas no experimento as observações de voluntá-

⁶ <http://postgis.net/> Acessado em 28/01/2016.

rios contidas no SOS previamente inseridas pela AGORA-VOS. Nos serviços responsáveis pela obtenção de dados da AGORA-IFM, foram selecionados dados de sensores gerados em períodos em que houve chuvas de tal forma que os valores de nível de água observados eram diferentes do nível usual dos córregos. Em relação aos dados de voluntários, foram selecionadas observações de forma que seus oferecimentos apresentassem níveis de água variados. A categorização dos dados de sensores considerou as faixas de níveis correspondentes aos pontos mais próximos de suas fontes. Considerando todo o conjunto de observações, os períodos de tempo a que estas se referiam eram diferentes devido à falta de dados de ambas as fontes para um mesmo período, entretanto, considera-se que tal fato não compromete este experimento, o qual visa demonstrar a factibilidade de um sistema baseado na AGORA-IFM.

O período de execução do processo de integração foi definido no serviço escalonador para um minuto, que corresponde a um tempo ligeiramente maior que o tempo de execução consumido para o conjunto de dados e método de integração selecionados. A cada execução, o fluxo de processamento produziu estimativas com estruturas de dados iguais para toda a área considerada, demonstrando a capacidade de integração automática dos dados de sensores e voluntários. Por outro lado, os dados disponíveis no SOS se mostraram ser pouco representativos devido à pouca quantidade e por concentrarem-se basicamente em dois únicos pontos com baixa variação de valores. Por esse motivo, o processo de integração foi novamente executado utilizando dados simulados.

Os dados de entrada para o cenário da nova execução foram simulados de forma que as observações ocupassem a região dos córregos analisados de maneira distribuída. Quinze observações de sensores, associadas a um identificador da forma “S#”, foram posicionadas nos pontos para os quais há informações de profundidade do leito e nível de água usual do córrego. Dessa forma, pôde-se utilizar todas as informações disponíveis e derivadas do cálculo da categorização. Nesse novo cenário, considerou-se uma chuva uniforme sobre toda a região de forma que o nível de água em todos os pontos fosse dois metros acima do nível usual, assim definido para que as categorias correspondentes fossem variadas. A [Tabela 1](#) apresenta, para cada ponto, a altura da margem menor, o nível de água usual e os limites superiores das faixas de medidas das categorias, todos medidos em metros. Por exemplo, para o ponto S1 da tabela, o nível usual é de 0,83m, o baixo varia de zero a 0,415m, o normal varia de 0,415m a 1,5215m, o alto varia de 1,5215m a 2,213m, que é a altura da margem, e o nível transbordando compreende valores acima de 2,213m.

As observações de voluntários, associadas a um identificador da forma “V#”, foram posicionadas em regiões de grande concentração urbana e para as quais não havia dados de sensores próximos. Foram assim estabelecidas para que não houvesse grandes áreas populosas sem informação alguma, o que diminui a precisão da interpolação espacial. Dessa forma, obteve-se, para esse novo cenário, a distribuição espacial dos pontos

Tabela 1 – Dados simulados de sensores

Id.	Nível usual (m)	Baixo (m)	Normal (m)	Alto (margem) (m)
S1	0,83	0,415	1,5215	2,213
S2	0,65	0,325	1,331	2,012
S3	0,69	0,345	2,3205	3,951
S4	0,50	0,25	1,6425	2,785
S5	0,29	0,145	1,301	2,312
S6	0,17	0,085	1,3595	2,549
S7	0,09	0,045	1,434	2,778
S8	0,45	0,225	2,4825	4,515
S9	0,65	0,325	1,973	3,296
S10	0,35	0,175	2,4705	4,591
S11	0,25	0,125	1,716	3,182
S12	0,38	0,19	1,7665	3,153
S13	0,16	0,08	2,072	3,984
S14	0,44	0,22	2,7855	5,131
S15	0,12	0,06	1,743	3,366

simulados apresentada na Figura 12.

Figura 12 – Cenário de referência.

O conjunto resultante de dados de entrada é apresentado na Tabela 2. Os valores de latitude e longitude estão em graus; as categorias correspondem a baixo (B), normal (N), alto (A) ou transbordando (T). Os níveis de água utilizados para se calcular a categoria das observações de sensores correspondem aos respectivos níveis usuais acrescidos de 2,0 devido à simulação de chuva. Sem a integração, essas observações pontuais seriam as

únicas informações disponíveis a tomadores de decisão.

Tabela 2 – Dados simulados categorizados

Id.	Latitude (°)	Longitude (°)	Categoria
S1	-21,998761	-47,840553	T
S2	-21,986058	-47,882542	T
S3	-22,000533	-47,899636	A
S4	-21,985558	-47,905014	A
S5	-22,000450	-47,899767	A
S6	-22,008725	-47,877544	A
S7	-22,002072	-47,899314	A
S8	-22,009089	-47,904997	N
S9	-21,999308	-47,928861	A
S10	-22,015114	-47,915197	N
S11	-22,031128	-47,868392	A
S12	-22,021161	-47,884492	A
S13	-22,018306	-47,89235	A
S14	-22,019097	-47,913208	N
S15	-22,031947	-47,849322	A
V1	-22,0172476	-47,9126188	T
V2	-22,0162654	-47,9059133	A
V3	-22,0188564	-47,8975153	A
V4	-22,0223177	-47,8865719	A
V5	-22,0295483	-47,8713906	A
V6	-22,0050653	-47,9257375	A
V7	-22,0125081	-47,9073188	N
V8	-22,0056049	-47,9028556	N
V9	-22,0042421	-47,8931352	A
V10	-22,0043441	-47,8841713	A
V11	-21,9946004	-47,9031426	A
V12	-21,9985497	-47,8970030	T
V12	-21,9930534	-47,8936878	T
V14	-21,9879400	-47,8749204	T
V15	-21,9947148	-47,8562737	T

O método kriging utilizado para interpolar esses dados foi o kriging ordinário com o modelo Gaussiano de variograma. O variograma é uma função incorporada no cálculo das estimativas que calcula, para uma mesma amostra, a variação de uma variável em relação às demais. Esse modelo teve seus parâmetros calibrados pelo método mínimos quadrados ponderados. Esses métodos e modelo foram selecionados porque, além de serem bastante utilizados em diversos contextos, apresentam comportamentos similares à distribuição de pontos da amostra. A interpolação foi feita usando uma resolução de aproximadamente um metro quadrado, ou seja, o valor de nível de água de cada ponto resultante corresponde ao nível de água em uma área de um metro quadrado ao redor do ponto.

5.3 Resultados

A interpolação das observações gerou 99.999 estimativas para toda a área coberta pelos córregos. A partir dos resultados produzidos no cenário com observações simuladas, dois mapas temáticos foram gerados manualmente por meio de um *script* em R independente do processo de integração para facilitar a compreensão das estimativas. Uma imagem compreendendo todos os córregos apresenta a largura destes muito estreita devido à grande extensão total ocupada, o que torna difícil visualizar as manchas produzidas pelas estimativas. Por esse motivo, os mapas gerados compreendem apenas um trecho do rio Monjolinho, de modo que na extremidade sul está a foz do córrego do Gregório e na extremidade norte está a foz do córrego Santa Maria do Leme. Para esse trecho, a interpolação produziu 45.019 estimativas. O mapa da [Figura 13](#) mostra os níveis de água ao longo desse trecho de acordo com às categorias a que estão associados.

O mapa da [Figura 14](#) evidencia as áreas com maior probabilidade de inundação. Ele foi produzido a partir da dicotomização dos dados gerados pela interpolação da seguinte maneira: os valores na categoria “transbordando”, isto é, aqueles que ultrapassaram a altura do córrego, foram associados ao número 1, e os demais foram associados ao número 0. Observa-se que a geração dos mapas das Figuras [13](#) e [14](#) não faz parte da AGORA-IFM, e foi feita apenas para apresentar os resultados da integração de forma gráfica.

5.4 Discussão

Durante a elaboração do experimento, foram alterados os parâmetros da integração ou mesmo o método todo simplesmente editando o *script* correspondente, o que não influenciou o restante do processo. Isso evidencia a modificabilidade proporcionada pela arquitetura em relação à alteração do método de integração. Além disso, o processo também foi executado usando apenas os serviços referentes a dados de sensores e apenas os referentes a de voluntários, demonstrando que o processo em si é adaptável, já que ele pode ser definido selecionando-se os serviços que o cliente desejar.

Na execução do experimento utilizando os dados providos do SOS, a integração foi realizada diversas vezes devido ao agendamento do escalonador, o que foi possível graças a todas as atividades do processo serem automatizadas. Dessa forma, também é evidenciado que a arquitetura possibilita que a integração se repita diversas vezes, o que pode ser muito útil quando há uma produção contínua de dados de modo que cada repetição atualize os resultados anteriores, aumentando a concentração temporal de informações. Essa possibilidade pode ser inviável em processos que incluam atividades manuais. Observa-se ainda que o tempo de cada execução foi relativamente curto devido ao algoritmo de integração ser simples, e que alguns modelos desenvolvidos por hidrólogos podem levar várias horas para serem executados.

Figura 13 – Resultado da integração.

O experimento que contou com um conjunto distribuído de dados simulados mostrou que a estimativa dos níveis de água usando dados de sensores e de voluntários aumenta a concentração de informações onde não há sensores mas há circulação de pessoas. Como mencionado anteriormente, antes da integração, apenas níveis de água pontuais correspondentes às observações estariam disponíveis aos tomadores de decisão. Considerando que cada estimativa fornece informação para uma região de um metro quadrado, a integração estimou o nível de água para uma área de aproximadamente 100km^2 correspondente à cobertura dos córregos, e uma área de 45.019m^2 correspondente ao trecho de rio apresentado nos mapas. As informações mostradas na Figura 13 podem aumentar o conhecimento da comunidade em relação à situação de um rio e melhorar a tomada de decisão por autoridades responsáveis. A Figura 14 pode evidenciar os pontos de um rio

Figura 14 – Resultado da integração dicotomizado.

mais suscetíveis a transbordamento, beneficiando, por exemplo, políticas públicas quanto à prevenção de inundações, especialmente em relação à infraestrutura urbana e gestão de áreas verdes. Assim, ambos os mapas podem ser usados para a própria população se informar e para a prevenção de perdas humanas e materiais, direcionar investimentos financeiros e evitar problemas sociais causados por inundações, como tráfego intenso e queda de energia.

CONCLUSÃO

Esta pesquisa apresentou uma arquitetura de apoio à integração de dados de sensores e de voluntários com o objetivo de aumentar a cobertura espaço-temporal de informações sobre inundações que são relevantes para promover a gestão de risco de desastres naturais. A arquitetura orientada a serviços proposta, AGORA-IFM, defini um processo de integração em seis macro atividades, manipular dados de sensores, manipular dados de voluntários, unificar esses dados em um único documento, categorizá-los de acordo com seus níveis de água, integrá-los e exportá-los.

Uma avaliação arquitetural também foi conduzida para se verificar sua adequação ao sistema para o qual foi projetada. Essa avaliação mostrou que a arquitetura contempla a maioria dos requisitos previstos antecipadamente, apesar de apresentar limitações quanto aos requisitos elicitados após sua elaboração.

Essa arquitetura foi instanciada para se realizar uma prova de conceito, em que se verificou que sua implementação é possível e pode gerar os resultados esperados. Entretanto, para que esses resultados sejam precisos, é necessário haver uma considerável quantidade de dados de sensores e de voluntários. Além disso, o método de integração, por exemplo a interpolação Kriging usada na prova de conceito, deve ser adaptado para que seus parâmetros representem adequadamente o rio sob estudo.

Por fim, a AGORA-IFM apresenta três principais contribuições: primeiramente, ela define um processo totalmente automatizado, produzindo resultados mais rápido em relação aos processos que envolvem atividades manuais e permitindo seu uso em vários contextos. Segundo, ela permite a integração de dados heterogêneos de forma flexível, possibilitando o uso de diferentes métodos de integração espaço-temporais facilmente permutáveis. Finalmente, ela define um processo de integração adaptável, permitindo a seleção dos componentes necessários e a adição de novos componentes.

Uma versão da arquitetura proposta nesta pesquisa ainda não incluindo as modi-

ficações sugeridas pela avaliação arquitetural foi aceita na *International Conference on Information Systems for Crisis Response and Management* (ISCRAM) de 2016. O artigo correspondente encontra-se no [Apêndice B](#).

6.1 Trabalhos Futuros

Como trabalhos futuros, o processo de integração pode ser estendido para torná-lo mais robusto, por exemplo, adicionando um fator de qualidade para diferenciar a confiabilidade dos dados de sensores em relação aos de voluntários. Além disso, é possível usar níveis de água previstos para produzir previsões de inundações. Ainda, uma otimização pode ser feita comparando os dados de duas execuções consecutivas para analisar a necessidade de atualizar o primeiro resultado, evitando uso desnecessário de recursos quando possível.

REFERÊNCIAS

ALBUQUERQUE, J. P.; HERFORT, B.; BRENNING, A.; ZIPF, A. A geographic approach for combining social media and authoritative data towards identifying useful information for disaster management. **International Journal of Geographical Information Science**, v. 29, n. 4, p. 667–689, 2015. Disponível em: <<http://dx.doi.org/10.1080/13658816.2014.996567>>. Citado 2 vezes nas páginas 26 e 49.

ALBUQUERQUE, J. P.; ZIPF, A. Collaborative information systems for disaster management: Building resilience against disasters by combining participatory environmental monitoring and vulnerability communication. In: **Proc. of the Alumni Seminar Natural Hazards**. Teresópolis, RJ: [s.n.], 2012. Citado na página 26.

ASSIS, L. F. F. G. d.; BEHNCK, L. P.; DOERING, D.; FREITAS, E. P.; PEREIRA, C. E.; HORITA, F. E. A.; UEYAMA, J.; ALBUQUERQUE, J. P. Dynamic sensor management: Extending sensor web for near real-time mobile sensor integration in dynamic scenarios. In: **30th IEEE International Conference on Advanced Information Networking and Applications (AINA)**. [S.l.: s.n.], 2016. Citado na página 27.

BABAR, M. A.; ZHU, L.; JEFFERY, R. A framework for classifying and comparing software architecture evaluation methods. In: IEEE. **Software Engineering Conference, 2004. Proceedings. 2004 Australian**. [S.l.], 2004. p. 309–318. Citado na página 30.

BAHARIN, S. S. K.; SHIBGHATULLAH, A. S.; OTHMAN, Z. Disaster management in malaysia: An application framework of integrated routing application for emergency response management system. In: IEEE. **Soft Computing and Pattern Recognition, 2009. SOCPAR'09. International Conference of**. [S.l.], 2009. p. 716–719. Citado 2 vezes nas páginas 41 e 42.

BAKILLAH, M.; LIANG, S. H.; ZIPF, A.; ARSANJANI, J. J. Semantic interoperability of sensor data with volunteered geographic information: a unified model. **ISPRS International Journal of Geo-Information**, Multidisciplinary Digital Publishing Institute, v. 2, n. 3, p. 766–796, 2013. Citado na página 49.

BOTTS, M.; PERCIVALL, G.; REED, C.; DAVIDSON, J. Ogc® sensor web enablement: Overview and high level architecture. In: **GeoSensor networks**. [S.l.]: Springer, 2008. p. 175–190. Citado na página 43.

BUSCHMANN, F.; MEUNIER, R.; ROHNERT, H.; SOMMERLAD, P.; STAL, M. **Pattern-Oriented Software Architecture: A System of Patterns**. [S.l.]: New York: John wiley & sons, 1996. v. 1. Citado na página 30.

BUTENUTH, M.; GÖSSELN, G. v.; TIEDGE, M.; HEIPKE, C.; LIPECK, U.; SESTER, M. Integration of heterogeneous geospatial data in a federated database. **ISPRS Journal of Photogrammetry and Remote Sensing**, Elsevier, v. 62, n. 5, p. 328–346, 2007. Citado na página 48.

- CÂMARA, J. H.; ALMEIDA, T. T.; CARVALHO, D. R.; FERREIRA, T. B.; BALARDINO, A. F.; OLIVEIRA, G. V.; FONSECA, F. J.; RAMOS, R. S.; SOUZA, W. D.; LISBOA-FILHO, J. A comparative analysis of development environments for voluntary geographical information web systems. In: **XV Brazilian Symposium on Geoinformatics – Geoinfo 2014**. [S.l.: s.n.], 2014. Citado na página 47.
- CIFERRI, C. D. d. A. **Integração de Sistemas de Banco de Dados Heterogêneos em Aplicações de Planejamento Urbano**. Dissertação (Mestrado) — Master's thesis, Universidade Estadual de Campinas, Campinas, SP, BR, 1995. Citado na página 48.
- CLEMENTS, P.; KAZMAN, R.; KLEIN, M. **Evaluating Software Architectures: Methods and Case Studies**. [S.l.]: Addison-Wesley, 2002. Citado 7 vezes nas páginas 15, 17, 32, 35, 37, 39 e 41.
- DEGROSSI, L. C.; ALBUQUERQUE, J. P. de; FAVA, M. C.; MENDIONDO, E. M. Flood citizen observatory: a crowdsourcing-based approach for flood risk management in brazil. In: **The 26th International Conference on Software Engineering and Knowledge Engineering, Vancouver, Canada**. [S.l.: s.n.], 2014. p. 570–575. Citado 2 vezes nas páginas 25 e 27.
- ERL, T. Soa: Princípios de design de serviços. **Pearson Prentice Hall, São Paulo**, v. 200, p. 18–21, 2009. Citado na página 31.
- ERSKINE, M.; GREGG, D. Utilizing volunteered geographic information to develop a real-time disaster mapping tool: A prototype and research framework. In: **CONF-IRM 2012 Proceedings**. [S.l.: s.n.], 2012. Citado na página 47.
- FLOWERDEW, R. Spatial data integration. **Geographical Information Systems**, v. 1, p. 375–387, 1991. Citado na página 48.
- FOTHERINGHAM, A. S.; ROGERSON, P. A. GIS and spatial analytical problems. **International Journal of Geographical Information Science**, Taylor & Francis, v. 7, n. 1, p. 3–19, 1993. Citado na página 48.
- GARLAN, D. Software architecture: A roadmap. In: **Proceedings of the Conference on The Future of Software Engineering**. New York, NY, USA: ACM, 2000. (ICSE '00), p. 91–101. ISBN 1-58113-253-0. Disponível em: <<http://doi.acm.org/10.1145/336512.336537>>. Citado na página 29.
- GILL, A. Q.; BUNKER, D. Crowd sourcing challenges assessment index for disaster management. In: **AMCIS**. [S.l.]: Association for Information Systems, 2012. ISBN 978-0-615-66346-3. Citado na página 47.
- GIORDANO, R.; PREZIOSI, E.; ROMANO, E. Integration of local and scientific knowledge to support drought impact monitoring: some hints from an italian case study. **Natural hazards**, Springer, v. 69, n. 1, p. 523–544, 2013. Citado na página 48.
- GOODCHILD, M. F. Citizens as sensors: the world of volunteered geography. **GeoJournal**, Springer, v. 69, n. 4, p. 211–221, 2007. Citado 3 vezes nas páginas 26, 46 e 47.
- GOODCHILD, M. F.; GLENNON, J. A. Crowdsourcing geographic information for disaster response: a research frontier. **Int. Journal of Digital Earth**, Taylor & Francis, v. 3, n. 3, p. 231–241, 2010. Citado na página 47.

- GORTON, I. **Essential Software Architecture**. 2. ed. [S.l.]: Springer Science & Business Media, 2011. Citado 3 vezes nas páginas 15, 53 e 54.
- GROSSO, E.; BOUJU, A.; MUSTIÈRE, S. Data integration geoservice: A first proposed approach using historical geographic data. In: CARSWELL, J.; FOTHERINGHAM, A.; MCARDLE, G. (Ed.). **Proc. of the 9th Web and Wireless Geographical Information Systems – W2GIS**. Maynooth, Ireland: Springer Berlin Heidelberg, 2009. (Lecture Notes in Computer Science), p. 103–119. ISBN 978-3-642-10600-2. Citado na página 48.
- GUESSI, M.; NAKAGAWA, E. Y.; MALDONADO, J. C. **Estabelecimento de Mecanismos para a Representação da Visão em Tempo de Execução de Arquiteturas de Referência Orientadas a Aspecto**. São Carlos, São Paulo, Brasil: [s.n.], 2010. 46 – 48 p. In: IV Workshop de Iniciação Científica e Tecnológica (WICT 2010). Citado na página 30.
- HORITA, F. E. A.; ALBUQUERQUE, J. P.; DEGROSSI, L. C.; MENDIONDO, E. M.; UYAMA, J. Development of a spatial decision support system for flood risk management in brazil that combines volunteered geographic information with wireless sensor networks. **Computers & Geosciences**, v. 80, p. 84 – 94, 2015. ISSN 0098-3004. Citado 5 vezes nas páginas 25, 26, 27, 46 e 48.
- HORITA, F. E. A.; DEGROSSI, L. C.; ASSIS, L. F. G. de; ZIPF, A.; ALBUQUERQUE, J. P. de. The use of volunteered geographic information (vgi) and crowdsourcing in disaster management: a systematic literature review. **AMCIS 2013 Proceedings**, 2013. Citado na página 42.
- HORITA, F. E. A.; FAVA, M. C.; MENDIONDO, E. M.; ROTAVA, J.; SOUZA, V. C.; UYAMA, J.; ALBUQUERQUE, J. P. AGORA-GeoDash: A geosensor dashboard for real-time flood risk monitoring. In: **Proceedings of the 11th International ISCRAM Conference**. Pennsylvania, USA: [s.n.], 2014. p. 304–313. Citado na página 85.
- KAZMAN, R.; BASS, L.; WEBB, M.; ABOWD, G. Saam: A method for analyzing the properties of software architectures. In: IEEE COMPUTER SOCIETY PRESS. **Proceedings of the 16th international conference on Software engineering**. [S.l.], 1994. p. 81–90. Citado na página 30.
- KAZMAN, R.; KLEIN, M.; BARBACCI, M.; LONGSTAFF, T.; LIPSON, H.; CARRIERE, J. The architecture tradeoff analysis method. In: IEEE. **Engineering of Complex Computer Systems, 1998. ICECCS'98. Proceedings. Fourth IEEE International Conference on**. [S.l.], 1998. p. 68–78. Citado na página 30.
- KAZMAN, R.; KLEIN, M.; CLEMENTS, P. **ATAM: Method for Architecture Evaluation**. Pittsburgh, PA, 2000. Disponível em: <<http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=5177>>. Citado 2 vezes nas páginas 30 e 33.
- KOTSEV, A.; PANTISANO, F.; SCHADE, S.; JIRKA, S. Architecture of a service-enabled sensing platform for the environment. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 15, n. 2, p. 4470–4495, 2015. Citado na página 48.
- KRAFZIG, D.; BANKE, K.; SLAMA, D. **Enterprise SOA: Service-Oriented Architecture Best Practices (The Coad Series)**. Upper Saddle River, NJ, USA: Prentice Hall, 2004. ISBN 0131465759. Citado na página 31.

- KRUCHTEN, P. B. The 4+ 1 view model of architecture. **Software, IEEE**, IEEE, v. 12, n. 6, p. 42–50, 1995. Citado na página 29.
- LONGUEVILLE, B.; LURASCHI, G.; SMITS, P.; PEEDELL, S.; GROEVE, T. Citizens as sensors for natural hazards: A vgi integration workflow. **Geomatica**, v. 64, p. 41–59, 2010. Citado na página 41.
- MANSOURIAN, A.; RAJABIFARD, A.; ZOEJ, M. V.; WILLIAMSON, I. Using SDI and web-based system to facilitate disaster management. **Computers & Geosciences**, Elsevier, v. 32, n. 3, p. 303–315, 2006. Citado na página 25.
- MEDIONDO, M. Integrated urban water management: Humid tropics. In: _____. [S.l.]: CRC Press, 2010. cap. Reducing vulnerability to water-related disasters in urban areas of the humid tropics, p. 109–127. Citado na página 25.
- MEISSEN, U.; FUCHS-KITTOWSKI, F. Towards a reference architecture of crowdsourcing integration in early warning systems. In: **Proceedings of the 11th International ISCRAM Conference**. [S.l.: s.n.], 2014. Citado na página 51.
- MENDIONDO, E. Reducing vulnerability ti water-related disasters in urban areas of the humid tropics. **Integrated Urban Water Management Humid Tropics, Paris, France**, p. 109–127, 2010. Citado na página 41.
- MOREIRA, R. d. B.; DEGROSSI, L. C.; ALBUQUERQUE, J. P. de. An experimental evaluation of a crowdsourcing-based approach for flood risk management. In: **Proceedings of the 12th Workshop on Experimental Software Engineering**. [S.l.: s.n.], 2015. Citado na página 47.
- MUNICHRE, Münchener Rückversicherungs-Gesellschaft. **Annual Statistics 2014**. 2014. Recuperado de <http://www.munichre.com/en/reinsurance/business/non-life/natcatservice/annual-statistics/>. Acessado em 2014-12-02. Citado na página 25.
- NORRIS, F. H.; STEVENS, S. P.; PFEFFERBAUM, B.; WYCHE, K. F.; PFEFFERBAUM, R. L. Community resilience as a metaphor, theory, set of capacities, and strategy for disaster readiness. **American journal of community psychology**, Springer, v. 41, n. 1-2, p. 127–150, 2008. Citado na página 41.
- OLIVEIRA, M. G. de; BAPTISTA, C. de S.; CAMPELO, C. E.; FILHO, J. A. M. A.; FALCÃO, A. G. R. Automated production of volunteered geographic information from social media. In: **XV Brazilian Symposium on Geoinformatics – Geoinfo 2014**. [S.l.: s.n.], 2014. Citado na página 47.
- OLIVER, M. A.; WEBSTER, R. Kriging: a method of interpolation for geographical information systems. **International Journal of Geographical Information System**, Taylor & Francis, v. 4, n. 3, p. 313–332, 1990. Citado na página 48.
- POSER, K.; DRANSCH, D. Volunteered geographic information for disaster management with application to rapid flood damage estimation. **Geomatica**, v. 64, n. 1, p. 89–98, 2010. Citado 5 vezes nas páginas 15, 26, 41, 42 e 47.

RAMOS, J. M.; VANDECASTEELE, A.; DEVILLERS, R. Semantic integration of authoritative and volunteered geographic information (vgi) using ontologies. In: **Proceeding of Association of Geographic Information Laboratories for Europe (AGILE) Conference 2013**. [S.l.: s.n.], 2013. Citado na página 48.

SANTOS, D. S.; OLIVEIRA, B.; GUESSI, M.; OQUENDO, F.; DELAMARO, M.; NAKAGAWA, E. Y. Towards the evaluation of system of systems software architecture. **VIII Workshop em Desenvolvimento Distribuido de Software, Ecossistemas de Software e Sistema de Sistemas (WDES)**, Maceio, Brasil, p. 53–57, 2014. Citado na página 33.

SCHADE, S.; DÍAZ, L.; OSTERMANN, F.; SPINSANTI, L.; LURASCHI, G.; COX, S.; NUÑEZ, M.; LONGUEVILLE, B. D. Citizen-based sensing of crisis events: sensor web enablement for volunteered geographic information. **Applied Geomatics**, Springer, v. 5, n. 1, p. 3–18, 2013. Citado na página 49.

SCHNEBELE, E.; CERVONE, G. Improving remote sensing flood assessment using volunteered geographical data. Copernicus Publications, 2013. Citado na página 50.

SCHNEBELE, E.; CERVONE, G.; KUMAR, S.; WATERS, N. Real time estimation of the calgary floods using limited remote sensing data. **Water**, Multidisciplinary Digital Publishing Institute, v. 6, n. 2, p. 381–398, 2014. Citado 2 vezes nas páginas 48 e 50.

SCHNEBELE, E.; CERVONE, G.; WATERS, N. Road assessment after flood events using non-authoritative data. **Natural Hazards and Earth System Science**, Copernicus GmbH, v. 14, n. 4, p. 1007–1015, 2014. Citado na página 50.

SCHNEBELE, E.; OXENDINE, C.; CERVONE, G.; FERREIRA, C.; WATERS, N. Using non-authoritative sources during emergencies in urban areas. In: HELBICH, M.; ARSANJANI, J. J.; LEITNER, M. (Ed.). **Computational Approaches for Urban Environments**. [S.l.]: Springer International Publishing, 2015, (Geotechnologies and the Environment, v. 13). p. 337–361. ISBN 978-3-319-11468-2. Citado na página 50.

SEI, Software Engineering Institute. **Community Software Architecture Definitions**. 2015. Recuperado de <http://www.sei.cmu.edu/architecture/start/glossary/community.cfm>. Acessado em 2015-11-06. Citado na página 29.

SIMONIS, I. **OGC Sensor Web Enablement Architecture**. 2008. 72 p. Citado na página 43.

SOMMERVILLE, I. Software engineering. In: _____. 9th. ed. New York: Addison-Wesley, 2011. cap. Service Oriented Architecture. Citado 3 vezes nas páginas 15, 31 e 32.

STEIN, M. L. **Interpolation of spatial data: some theory for kriging**. [S.l.]: Springer Science & Business Media, 2012. Citado na página 84.

TU, J.-Y.; CHOU, C.; CHU, P.-S. The abrupt shift of typhoon activity in the vicinity of taiwan and its association with western north pacific-east asian climate change. **Journal of Climate**, v. 22, n. 13, p. 3617–3628, 2009. Citado na página 42.

UNISDR, United Nations Office for Disaster Risk Reduction. **Brazil – Disaster Statistics**. 2014. Recuperado de <http://www.preventionweb.net/english/countries/statistics/?cid=24>. Acessado em 2014-12-02. Citado na página 26.

WAN, Z.; HONG, Y.; KHAN, S.; GOURLEY, J.; FLAMIG, Z.; KIRSCHBAUM, D.; TANG, G. A cloud-based global flood disaster community cyber-infrastructure: Development and demonstration. **Environmental Modelling & Software**, Elsevier, v. 58, p. 86–94, 2014. Citado na página 50.

WANG, F.; YUAN, H. Challenges of the sensor web for disaster management. **International Journal of Digital Earth**, Taylor & Francis, v. 3, n. 3, p. 260–279, 2010. Citado na página 41.

ZAFFANI, A. G.; MENDIONDO, E. M. **Poluição difusa da drenagem urbana ecohidrológica: diagnóstico atual e cenários de longo prazo em bacias urbanas de São Carlos, SP**. Dissertação (Mestrado) — Universidade de São Paulo, 2012. Citado 2 vezes nas páginas 15 e 85.

ZHANG, L.-J.; ZHANG, J.; CAI, H. **Services computing**. [S.l.]: Springer, 2007. Citado na página 31.

ZHAO, P.; FOERSTER, T.; YUE, P. The geoprocessing web. **Computers & Geosciences**, v. 47, p. 3 – 12, 2012. ISSN 0098-3004. Citado na página 43.

FICHAS DE AVALIAÇÃO ARQUITETURAL

Neste apêndice são apresentadas todas as fichas utilizadas na avaliação dos requisitos/cenários selecionados para análise. Cada ficha inclui a descrição de um requisito e do correspondente cenário dividido em ambiente, estímulo e resposta, os atributos de qualidade relacionados, pontos sensíveis, de contrabalanço, riscos, não riscos e comentários levantados durante as discussões. Essas fichas foram empregadas na aplicação do *Architecture Trade-off Analysis Method*, descrito no [Capítulo 5](#).

Quadro 7 – Ficha de análise do requisito R2

Análise de Requisito				
Id.: R2	Requisito: A arquitetura deve permitir a elaboração de um workflow diferente do preestabelecido por meio do uso individual dos serviços oferecidos.			
Atributo(s): Reusabilidade.				
Ambiente: Execução normal.				
Estímulo: Utilização dos serviços individualmente (não é usado o serviço de integração pré-estabelecido).				
Resposta: Os serviços são fornecidos normalmente.				
Decisão arquitetural	Sensibilidade	Contrabalanço	Risco	Não Risco
D2: Disponibilização pública dos principais serviços.	S2.1			N2.1
Sensibilidade				
S2.1: Os principais serviços são disponibilizados publicamente permitindo o reúso em diferentes contextosl.				
Risco				
N2.1: Uma boa decisão arquitetural uma vez que todas as fontes de dados utilizadas também são publicas e o usuário pode utilizar os serviços de forma independente estabelecendo seu próprio workflow.				

Quadro 8 – Ficha de análise do requisito R3

Análise de Requisito

Id.: R3	Requisito: A arquitetura deve permitir a troca do script de integração com pouco esforço.			
Atributo(s): Modificabilidade.				
Ambiente: Rotina de manutenção.				
Estímulo: Um stakeholder substitui/altera o script de integração modificando o arquivo correspondente, sem alterar o formato de suas entradas e saídas.				
Resposta: A arquitetura deve permitir a troca do script de integração com pouco esforço.				
Decisão arquitetural	Sensibilidade	Contrabalanço	Risco	Não Risco
D3: Separação do script de integração dos demais serviços.	S3.1	T3.1, T3.2		N3.1

Sensibilidade

S3.1: A separação do script permite a substituição do mesmo sem a alteração do serviço.

Contrabalanço

T3.1: Performance: Um algoritmo de integração interno provavelmente seria mais eficiente.

T3.2: Interoperabilidade: Uma vez que somente é aceito scripts escritos na linguagem R.

Risco

N3.1: Boa decisão uma vez que se mostrou a melhor alternativa para se alcançar o requisito em questão. Porem, um grande esforço deverá ser alocado quando necessária a alteração dos padrões de entrada/saída de um novo script. Sugestão: Encapsular mais o R e usar os padrões da OGC.

Quadro 9 – Ficha de análise do requisito R4

Análise de Requisito				
Id.: R4	Requisito: O sistema deve permitir a substituição e adição de novos serviços (e.g. processamento ou coleta de novas fontes de informação) com pouco esforço (Recurso investido para alterar o Workflow).			
Atributo(s): Modificabilidade.				
Ambiente: Rotina de manutenção.				
Estímulo: Adicionar novos serviços à arquitetura.				
Resposta: Os novos serviços são adicionados à arquitetura com o esforço previsto.				
Decisão arquitetural	Sensibilidade	Contrabalanço	Risco	Não Risco
Sem decisão.	S4.1		R4.1	
Sensibilidade				
S4.1: A falta de decisão pode tornar esse cenário infactível.				
Risco				
R.4.1: É um risco, pois a falta de padronização na troca de mensagens dificulta a substituição e adição de novos serviços.				

Quadro 10 – Ficha de análise do requisito R6

Análise de Requisito				
Id.: R6		Requisito: O sistema deve ter alta disponibilidade de software.		
Atributo(s): Disponibilidade, recuperabilidade.				
Ambiente: O processo de integração está em execução.				
Estímulo: Uma falha de software no servidor.				
Resposta:O processo de integração é cancelado e reiniciado por um serviço redundante.				
Decisão arquitetural	Sensibilidade	Contrabalanço	Risco	Não Risco
Sem decisão.	S6.1		R6.1	

Sensibilidade

S6.1: A falta de decisão pode tornar esse cenário infactível.

Risco

R6.1: É um risco pois a utilização de único serviço orquestrador potencializa a ocorrência de falhas de software. Uma alternativa de decisão possível seria usar configurações alternativas, i.e., diferentes workflows. Outra possível decisão seria a descentralização do orquestrador usando, por exemplo, uma abordagem de coreografia.

Quadro 11 – Ficha de análise do requisito R7

Análise de Requisito				
Id.: R7	Requisito: Em caso de falha de software o sistema deve enviar os dados brutos. (deixar um caso mínimo).			
Atributo(s): Recuperabilidade.				
Ambiente: Execução normal.				
Estímulo: Ocorre uma falha de software.				
Resposta: O processo de integração é executado e são fornecidas informações mínimas para a tomada de decisão.				
Decisão arquitetural	Sensibilidade	Contrabalanço	Risco	Não Risco
Sem decisão.	S7.1		R7.1	

Sensibilidade

S7.1: A falta de decisão pode tornar esse cenário infactível.

Risco

R7.1: Em caso de falha de software o sistema fica inoperante sem fornecer tipo de informação mínima para a tomada de decisão. Uma alternativa seria a flexibilização do workflow para permitir o fornecimento de dados brutos ou os últimos resultados do processo de integração para a tomada de decisão.

Quadro 12 – Ficha de análise do requisito R8

Análise de Requisito

Id.: R8	Requisito: A arquitetura deve usar padrão(es) reconhecidos para troca de mensagens.			
Atributo(s): Interoperabilidade.				
Ambiente: Execução normal.				
Estímulo: O processo de integração é iniciado.				
Resposta: Troca de mensagens através de padrões reconhecidos.				
Decisão arquitetural	Sensibilidade	Contrabalanço	Risco	Não Risco
D5: Formatação das mensagens encapsuladas em um serviço.	S8.1	T8.1	R8.1	

Sensibilidade
S8.1: O encapsulamento empregado influencia a obtenção de interoperabilidade entre os serviços.

Contrabalanço
T8.1: Afeta a manutenibilidade.

Risco
R8.1: É um risco pois não há um modelo de dados explícito para permitir a evolução arquitetural. Além disso, a arquitetura não prevê a evolução do padrão em si, impactando diretamente na manutenibilidade do sistema. A melhor alternativa arquitetural seria adotar padrões estabelecidos da área. Sugestão: Faltou definir schema XML.

Quadro 13 – Ficha de análise do requisito R10

Análise de Requisito				
Id.: R10	Requisito: O tempo de resposta deve ser baixo.			
Atributo(s): Comportamento temporal.				
Ambiente: Execução normal.				
Estímulo: Uma aplicação requisita a integração de dados.				
Resposta: O processo de integração é executado e o resultado fornecido em pouco tempo.				
Decisão arquitetural	Sensibilidade	Contrabalanço	Risco	Não Risco
D4: Hospedagem em nuvem.	S10.1			N10.1
Sensibilidade S10.1: A hospedagem em nuvem fornece garantias de desempenho.				
Risco N10.1: Não risco pois a hospedagem em nuvem garante o desempenho do sistema a depender do contrato de serviço firmado.				

Quadro 14 – Ficha de análise do requisito R11

Análise de Requisito					
Id.: R11		Requisito: A arquitetura deve permitir o armazenamento de dados pré-processados para serem usados diretamente pelo serviço de integração.			
Atributo(s): Comportamento temporal.					
Ambiente: Execução normal.					
Estímulo: Um usuário requisita diversas integrações em paralelo sobre um mesmo conjunto de dados de entrada, usando um modelo diferente em cada integração.					
Resposta: Um conjunto de dados pré-processados (categorizados) é independentemente fornecido às requisições para ser usado diretamente pelo serviço de integração.					
Decisão arquitetural		Sensibilidade	Contrabalanço	Risco	Não Risco
Sem decisão.		S11.1		R11.1	

Sensibilidade

S11.1: A falta de decisão pode tornar esse cenário infactível.

Risco

R11.1: Para se realizar a integração usando mais de um modelo, mesmo que para um único conjunto de dados de entrada, é necessário executar todo o processo, consumindo desnecessariamente o tempo de pré-processamento que é comum para todos os modelos. Uma alternativa seria fornecer um processo de pré-processamento independente do processo já definido, cujo resultado pudesse ser usado diretamente pelo serviço de integração (isolando a integração do restante do processo, já que esta etapa é a mais lenta).

**A SOFTWARE ARCHITECTURE TO
INTEGRATE SENSOR AND VOLUNTEERED
GEOGRAPHIC INFORMATION FOR FLOOD
RISK MANAGEMENT**

A Software Architecture to Integrate Sensor and Volunteered Geographic Information for Flood Risk Management

Raul E. Simoni Castanhari

ICMC, University of São Paulo
raul@icmc.usp.br

Roberto dos Santos Rocha

ICMC, University of São Paulo
ERCIS, University of Münster
rsrocha@usp.br

Sidgley Camargo de Andrade

ICMC, University of São Paulo
UFTPR, Federal University of Technology
sidgleyandrade@usp.br

João Porto de Albuquerque

Centre for Interdisciplinary Methodologies,
University of Warwick
J.Porto @warwick.ac.uk

ABSTRACT

Natural disasters are phenomena that can cause great damage to people in urban and rural areas, and thus require preventive and reactive measures. If they involve multiple sources of information, these measures can be more useful and effective. However, the integration of heterogeneous data still poses a challenges due to the differences in their structures and semantic contents. To overcome this difficulty, this paper outlines a service-oriented architecture, as part of a geospatial platform for disaster risk management, which aims to support the integration of sensor data and Volunteered Geographic Information (VGI) related to floods. The architectural components are implemented as Web services, and their composition enables sensor data to be integrated with the VGI by using several algorithms in a flexible and automated manner. The qualitative evaluation showed the architecture is suitable for its purposes. In addition, the architecture was implemented as a proof of concept and the results of its integration process were used to generate thematic maps, which can be used to heighten community awareness and support decision makers in managing flood risks.

Keywords

Service-Oriented Architecture, Integration, Geospatial Data, Volunteered Geographic Information, Flood Management.

INTRODUCTION

Natural disasters, such as, floods, storms, fires, and hurricanes can cause serious damage to people if they occur in places of human activity, since they lead to loss of food, natural resources, urban infrastructure, and, most

*Long Paper – Geospatial Data and Geographical Information Science
Proceedings of the ISCRAM 2016 Conference – Rio de Janeiro, Brazil, May 2016
Tapia, Antunes, Bañuls, Moore and Porto, eds.*

importantly, human lives. In 2014, natural disasters incurred financial losses close to US\$110 billion worldwide and caused 7,700 fatalities (Munich, 2014). With regard to the type of natural disasters that occurred in 2014, 73% were caused by extreme events of a hydrometeorological origin, *i.e.*, floods, flash floods, thunderstorms, avalanches, as well as landslides caused by rainfall (Munich, 2014).

Furthermore, there has been increasing concern to be prepared for oncoming events and to mitigate their consequences. This has encouraged the establishment of resilient communities, especially those located in risk areas, such as landscapes or near rivers. Resilience can be defined as the capacity of communities to withstand extreme events (Mediondo, 2010), and one alternative way of achieving this, is by using environmental monitoring systems (Mansourian, Rajabifard, Zoj and Williamson, 2006). Another alternative is to make use of timely, accurate and continuous information. This could greatly assist the emergency agencies involved in flood risk management, as discussed by Horita, Albuquerque, Degrossi, Mendiondo and Ueyama (2015). However, the continuous monitoring of flood risks requires local information such as observational data on rainfall and river levels to evaluate the risk of potential flooding. This can be a considerable challenge in regions where there is a vast territorial extension.

In this context, Volunteered Geographic Information (VGI) has been of great practical value as a means of complementing the existing geospatial datasets (Goodchild, 2007) owing to the large number of volunteers who act as potential ‘sensors’ (Poser and Dransch, 2010). This raises several challenges to open research to make VGI an effective data source for flood risk management. One of them is how to integrate VGI in existing data sets rather than considering it as a parallel source of information (Albuquerque, Herfort, Brenning and Zipf, 2015). In particular, research about how to integrate VGI with sensor data is still recent, for the following reasons: (i) the sensor data is structured and standardized, while the volunteered information is almost entirely unstructured; (ii) data from sensors are precise, while those from volunteers depends on their perceptions; and (iii) the volunteered data are generally less reliable than that obtained from sensors. However, integrating these data is important, because the use of several data sources can provide a pool of disaster information that is more complete and updated than just relying on one source (Goodchild, 2007; Horita *et al.*, 2015).

Moreover, the nature of VGI usually requires a degree of manual processing (*e.g.*, observation of a video by a human to extract information). This limits the use of VGI contexts since this kind of task has very specific goals and makes the process inflexible. In addition, manual activities are very time consuming, which can be critical in a disaster management situation, where emergency agencies have to act rapidly.

This research tackles these issues by proposing a service-oriented architecture to support the integration of sensor and volunteered data, and assist it by: (i) automating the integration, speeding up the process and making it applicable to different situations; (ii) making the process flexible, by allowing the employment of different spatial-temporal methods; and (iii) making the process adaptable, so that it can be modified through Web service chaining.

The remainder of this paper is structured as follows: the background section introduces the basic concepts that will be used throughout the paper; the Information Fusion and Management section describes the proposed architecture; the Architecture Evaluation section explains how the architecture was qualitatively evaluated; the Proof of concept section discusses the results obtained from its instantiation and how it is used to produce thematic maps; the Related Works section makes a comparison with previous related works; and the Conclusion summarizes the findings of the research.

BACKGROUND

In this section, there is an outline of the basic concepts related to geospatial Web services and VGI.

Geospatial Web Services

The Service-Oriented Architecture (SOA) is a very promising architecture for the practical implementation of the next generation of geographical information systems (GIS) (Lu, 2005). It has provided a technology evolution that moves from standalone GIS applications towards a more loosely coupled and distributed model based on Geospatial Web Services (GWS) (Zhao, Foerster and Yue, 2012). GWS can be viewed as a modular Web application that provides services regarding geospatial data (Rezég, Laskri and Servigne, 2010).

Before it can be processed, the heterogeneity of geospatial data must be overcome. The Open Geospatial

Consortium (OGC)¹ is a step toward enabling the interoperability of geospatial data. It develops protocols and open specifications that allow access to distributed and diverse data through interfaces (Zhao *et al.*, 2012). Examples of these specifications are the Sensor Observations Service (SOS), a service for requesting, filtering, and retrieving observations and sensor system information, the Web Map Service (WMS), a service used to transfer maps, and the Observations and Measurements (O&M), an XML-based standard which contains fields like position, timestamp and measured values (Walenciak and Zipf, 2010; Zhao *et al.*, 2012).

Furthermore, geospatial domains require massive computational power to collect, translate, analyze, and share distributed heterogeneous geospatial data. In this context, cloud computing can be applied to solve and overcome the challenges in GIS applications, because it maximizes the efficiency and data safety during the collaboration, while reducing the time and system development costs (Bhat, Shah and Ahmad, 2011).

Volunteered Geographic Information

Volunteered Geographic Information (VGI) is spatial related information that is volunteered produced and made available by common citizens using appropriate tools, like interactive mobile devices and Web portals (Goodchild, 2007). VGI has emerged in the last few years as an alternative spatial data source on the Web (de Oliveira, de Souza Baptista, Campelo, Acioli Filho and Falcão, 2014) and it has been increasingly recognized by researchers as an important resource to support disaster management (Câmara, Almeida, Carvalho, Ferreira, Balardino, Oliveira, Fonseca, Ramos, Souza and Lisboa-Filho, 2014; Poser and Dransch, 2010).

VGI can be obtained through different collaborative activities, such as (i) information sharing through social media (*e.g.* Twitter); (ii) collaborative mapping (*e.g.*, OpenStreetMap), and (iii) participatory sensing (*e.g.*, citizen observatories) (Horita *et al.*, 2015). In this way, VGI can vary greatly both in terms of structure and content, which can hamper its use in geographic systems, especially in an automatic way. Thus, the main challenge related to this kind of information source is how to integrate it with other data sets so that a more effective approach can be adopted (Poser and Dransch, 2010).

Finally, as anyone can produce VGI, this information cannot correspond to the reality, - that is, its reliability should be considered when it is being used. However, some results obtained from researches have shown that the use of VGI assists in establishing a greater awareness of a particular scenario (Erskine and Gregg, 2012; Moreira, Degrossi, and de Albuquerque, 2015; Poser and Dransch, 2010). Goodchild and Glennon (2010) also concluded that it is more difficult to distinguish between official and volunteered information during a crisis, and that volunteered information tends to spread faster than that from official sources.

Integration of VGI and other Information Sources

Geographic data integration is the process of making different data sets compatible with each other, so they can be clearly displayed on the same map so that their relationships can be analyzed (Flowerdew, 1991). In this respect, some research projects are being carried out to obtain integrated heterogeneous geographic data. For example, Schnebele, Oxendine, Cervone, Ferreira and Waters (2015) proposed an application to fill the gaps in remote sensing data during disasters by making use of data collected from non-authoritative sources, like VGI, or collected for purposes other than disaster assessment, such as that from traffic cameras. However, this study does not allow users to produce a set with all the integrated data. In a previous work, Schnebele, Cervone and Waters (2014) devised a methodology for estimating flood damage by integrating authoritative and non-authoritative data. They integrated the information of photographs classified by crowdsourcing and videos that were manually classified to determine the severity of the flood damage, but their methodology does not allow automatic integrations.

Kotsev, Pantisano, Schade and Jirka (2015) proposed an approach to enable the interoperability of environmental sensors that combine on-board storage with Web service capabilities. Through their proposed approach, observation data can easily be combined and reused. However, despite combining sensor data (even that produced by a citizen's mobile devices), their approach fails to take account of data provided by citizen perceptions (like filling out a form).

Meissen and Fuchs-Kittowski (2014) proposed a reference architecture of crowdsourcing integration in early

¹ <http://www.opengeospatial.org/>

warning systems, which integrates the components of monitoring systems with those of crowdsourcing systems components. In this way, their architecture has a capacity for participant management to improve the system for issuing and spreading warnings, but it cannot be implemented, since it requires further specifications before being developed in real contexts.

AGORA INFORMATION FUSION AND MANAGEMENT (AGORA-IFM) ARCHITECTURE

In this section, we examine some previous works, and provide an overview of the architecture, together with an explanation of how it works.

Preliminary Work

Floods are the natural disaster that cause most damage in Brazil, with approximately 48 million people affected between 1980 and 2010 (UNISDR, 2014). In this context, a Brazilian research project was proposed aiming to create a platform to bring together the perspectives of science, the general public, and the government to tackle social challenges, with an initial focus on floods (Albuquerque and Zipf, 2012).

The project comprises three parts (Figure 1): (i) acquisition, responsible for obtaining and storing geosensor data and VGI; (ii) integration, where the goal is to integrate data from the two previous types; and (iii) application, responsible for producing indicators to facilitate the decision-making process and heighten community awareness. As highlighted in Figure 1, this study takes place at the integration part of the project.

Figure 1. Project Overview

Overview

The service-oriented architecture called AGORA Information Fusion and Management (AGORA-IFM) was proposed to support the integration of sensor data and VGI by different spatial-temporal methods in an easily updatable and automatic way; the purpose of this was to speed up the integration process and make it more flexible. Its results allow a greater concentration of the spatial and temporal information that is required for the management of natural disasters.

This architecture uses the Dynamic Sensor Management (AGORA-DSM), a system that dynamically manages heterogeneous geosensors, to obtain sensor data (Assis *et al.*, 2016). These data are coded in O&M. This architecture obtains volunteered data by means of the Volunteered Observation Service (AGORA-VOS), a system accessible via Web site or smartphone application and managed by a customized instance of the Ushahidi² crowdsourcing platform (Degrossi *et al.*, 2014). In general, VGI can comprise unstructured texts, photos, videos and audios, which represent different data structures. It is hard to extract relevant information from these data and this is usually done manually. The interface of this system guides the volunteers on how to convey their information, and structures it as an O&M, although it also stores other unstructured data. AGORA-DSM and AGORA-VOS embed their data in a SOS server so that they can be made available. They correspond

²<http://www.ushahidi.com/>

to the acquisition part of the project (Figure 1) and are not included in this study. AGORA-IFM integrates sensor data and VGI through these systems, together with an engine of spatial-temporal geostatistical functions that isolates the integration algorithm. This makes it easily changeable and allows an automatic integration of the two previous data types. Finally, the results are stored in a geographic database and exported via WMS.

Workflow

Six main activities are proposed in the architecture: handling sensor data, handling VGI, bind data, categorizing data, executing R script and exporting data. The activities are arranged by an orchestrator that is responsible for executing them in the right order. Figure 2 shows the workflow based on the Business Process Model and Notation (BPMN). This workflow is triggered periodically by a scheduler and, in each execution, it processes the data related to a time defined by user, for example data arrived since its previous execution. The scheduler activation period is programmable - for example, the workflow can be triggered every ten minutes. We have defined a component to execute the activities for each lane, where, in this case, the components were implemented as Web services. The orchestrator and the scheduler themselves are also components, as illustrated in the architecture displayed in Figure 3 as a Unified Modeling Language component diagram.

Figure 2. Workflow for AGORA-IFM architecture

Figure 3. Components of AGORA-IFM architecture

The two first activities, ‘Handle sensor data’ and ‘Handle VGI’, are responsible for handling data that was previously collected by the acquisition part and stored separately in a SOS server. ‘Handle sensor data’ uses AGORA-DSM to achieve its goal, what includes making a data selection. AGORA-DSM can store several kinds of sensor data, like water level, wind speed or air temperature, as well as storing historical data. Hence, this activity supports the selection of the data property and period - for example, by selecting the water level from the previous five minutes. ‘Handle VGI’ uses AGORA-VOS to obtain volunteered data, from another selection. AGORA-VOS can store some predefined kinds of data, which are chosen by volunteers while the report is being sent. For example, it might be a free text, the state of the street around a river (whether it is passable or not), or a water level based on the volunteer’s perception (normal, high, etc.). It also stores historical reports. This activity supports the selection of the data type and period, but currently only selects data stored with the type of water level to allow an automatic categorization of the activities that follow. These two first activities obtain data in O&M but, despite this, there are some variations in these data. For this reason, the activities also transform data from both sources into a single format, which allows them to be categorized and parsed faster. The services that implement these activities correspond to the following components: ‘Sensor data handler’ and ‘VGI handler’.

The third activity, ‘Bind data’, just bind sensor and volunteered data in a single source. The service that implements it, which corresponds to the ‘Data binder’ component, is also used internally by the first activity to bind data from all sensors, and by the second activity to bind all data obtained from the volunteers.

The fourth activity, ‘Categorize data’, involves categorizing all the data, which is based on the water level values and the metadata of the VGI. The categorization consists of assigning one of the categories (‘low’, ‘normal’, ‘high’ or ‘overflowing’) to each observation, so that the situation of the river can be easily understood. The thresholds that determine the category for a given water level value depend on the river and vary at each river point; for this reason a previous analysis of all the rivers in question is required. This research is based on the assumption that the thresholds have been previously established. ‘Data categorizer’ is the component related to the services that carries out this categorization.

The fifth activity, ‘Handle R data’, performs the integration of sensor data and VGI in an easily updatable and automatic way. It consists of a service, which corresponds to the ‘R data handler’, that interacts with an engine of algorithms written in the R language. The parameters of the service interface include the path to the program script that has to be executed and the categorized data that have to be integrated. This activity is carried out as follow: first, the component transforms the data passed from the orchestrator to specific data types, then, it sends

these data to an external R engine. Next, the engine reads the script file that is specified by the path argument and executes it line by line. Lastly, the engine returns the resulting data to the component, which transforms the data back to common data types and returns them to the orchestrator. As this process isolates the integration algorithm, the effort to adapt it, for example by changing its parameters or even changing the integration method for another statistical model, only involves updating the script, in the cases that the input and output data of the algorithms are of the same type. It makes the process quite generic and easy to update. Adapting the algorithm in the cases that the inputs and outputs are of different types is also possible through a transformation (see the Implementation section).

Finally, the last activity, 'Export data', makes the results available for external applications. This is performed by arranging the integrated data in the way expected by a geographic database (generally at geometric points with the values of their associated water level), and then connecting to it and storing each resulting point with its geographic position format. In this way, an external server accesses these data and exports them via WMS. The service that carries out such activity corresponds to the 'Data exporter' component.

All the integration process (as represented in Figure 2 by the clock as an initial event) is initialized by a component that schedules it so that it can be executed periodically, which is possible since none of the activities require manual activities. In this way, new incoming data can be processed to update the results, which increases the temporal data coverage and provides a situational view of the considered area that is updated along the time. The 'Scheduler' component is responsible for this behavior.

EVALUATION OF THE ARCHITECTURE

An adaptation of the Architecture Tradeoff Analysis Method (ATAM) was employed to evaluate the AGORA-IFM.

ATAM

This method assesses the architecture by comparing the decisions made by the architect with regard to the architectural requirements and scenarios to determine if they conform to the relevant quality attributes. It involves people with knowledge in software engineering and domain application (project team), software architecture (evaluation team), and other stakeholders including the end users and customers. The method defines two phases, that involve a meeting with selected teams and a set of steps. The first phase has six steps and involves the evaluation and the project team. The second phase has the previous six steps from phase one and further three steps, involving all the stakeholders. Along the application of the method, we identify, refine, and analyze architectural decisions, priority requirements and scenarios by verifying risks (potentially problematic decisions) and non-risks (good decisions). See Clements et al. (2002) for details.

AGORA-IFM Architecture Evaluation and Results

We employed an adapted ATAM because some artifacts used in it had been designed previously rather than design them during the evaluation. The purpose of this was to save time, and then we just refined these artifacts during the meetings. Table 1 shows the architectural decisions made by the architect.

Id.	Architectural decision	Quality attributes	Description
D1	Service-oriented architecture.	Modularity Interoperability	The features of the integration process were arranged into independent services.
D2	Public availability of the main services.	Reusability	The main services were made publicly available to allow their individual use in different contexts.
D3	Separation of data integration script and other services.	Modifiability	The integration script is executed from an independent file regarding to the implementation code. This means that, if the script inputs and outputs are equal to the ones required by the service that executes it, changing the script will require little effort.
D4	Cloud hosting.	Availability/ Time Behavior	The architecture is deployed in cloud to ensure high hardware availability and tolerance to hardware faults.
D5	Message formatting	Interoperability	The message formatting encapsulation in a service

	encapsulated in service.		facilitates a change of the formatting standard.
--	--------------------------	--	--

Table 1. List of Architectural Decisions

As a result, we were able to determine the points where the architecture satisfies its requirements and where it has limitations. Table 2 summarizes these points. Requirement R8 was not met because, even facilitating the formatting standard change, it was found that it would have to determine what specific standard would be used in each message exchanging. This has not been carried out yet because there are many standards in this domain and different kinds of messages that have to be considered, and this is one of our current tasks. For the same reason, requirement R4 was not met because, although the components are distributed as web services, the format of the data exchanged by the new services needs to be defined. However, new services that conform to the formats of the services already defined can be seamless added or replaced. Requirements R6, R7 and R13 were not met due to the lack of decisions. This was because these requirements were identified during the evaluation, after the AGORA-IFM had been developed. R7 involves reducing the integration costs in cases where software faults to the system remain active; R13 consists of providing previously categorized and stored data to clients be able to directly do several integration requests in parallel. We are also currently considering them. The other requirements were successfully met, including the most important, which is to change the integration script. R2 involves allowing the workflow to be customized, like using just sensor data as the source; and allowing services to be used individually like just obtaining volunteered data. The integration process was demonstrated by the implementation of AGORA-IFM, which is outlined in detail in the next section.

Requirement	Quality Attribute	Architectural Decision	Risk	Nonrisk
R2. Customized use of the workflow and individual use of the services.	Reusability	D2. Public availability of the main services.		X
R3. Changing the integration script.	Modifiability	D3. Separation of the data integration script and other services.		X
R4. Replacing and addition of new services to the workflow.	Modifiability	No decision.	X	
R6. High software availability.	Availability, Recoverability	No decision.	X	
R7. Providing a lightweight workflow.	Recoverability	No decision.	X	
R8. Using recognized standards.	Interoperability	D5. Message formatting encapsulated in service.	X	
R10. The integration response time should be low.	Time Behavior	D4. Cloud hosting.		X
R13. Integration by several models in parallel.	Time Behavior	No decision.	X	

Table 2. Summary of the Evaluation Result of AGORA-IFM

PROOF OF CONCEPT

A proof of concept is demonstrated through the implementation of AGORA-IFM and an experiment carried out in Monjolinho River in the center of the city of São Carlos, São Paulo, Brazil, where flooding often affects the city's infrastructure and citizens.

Implementation

The described process was implemented in accordance with this architecture to determine its feasibility and results. Architectural components and a client were developed as cloud services in the Azure³ platform using the Java programming language, and their communication is operated via SOAP with the support of the Axis⁴ framework. All the loaded data is stored as text to maintain interoperability between different systems, which

³<https://azure.microsoft.com/>

⁴<http://axis.apache.org/>

contain information like the observed value, position, date and time.

The categorization service calculates the category so that the numbers 1, 2, 3 and 4 correspond to the low, normal, high and overflowing categories respectively, and stores them in the previous text data set. This is necessary to enable processing over VGI that reports a water level such as 'normal water level', which cannot be integrated with numerical values. The integration process uses these numbers to make predictions in regions where there are no observations but which have sensor or volunteered observations close to them.

The rJava⁵ package was used to execute R spatial-temporal statistics from Java. This is responsible for the interface between both languages and sending the script commands to the R processor, in this case, GNU's R 3.1.2. The only points of contact between the Java and R programs are the input and output data types transformations, which make all the rest independent.

The statistical method to obtain the water level estimates was instantiated with a spatial interpolation called Kriging through the geoR⁶ library. Using other methods with inputs and outputs of types different from Kriging is also possible, but this requires the transformation of these types. This can be carried out either inside the script, by transforming the R types generated by the already implemented services in the new R types, or creating new web services, by transforming the Java code in the new R types. It is also possible to use methods from different R libraries apart from the geoR, since it just requires the corresponding libraries be installed on the server that is hosting the R engine. It should be pointed out that the contribution of this research is not the integration algorithm itself, but the support of it, *i.e.*, the possibility that the architecture provides to use different integration methods to integrate sensor and volunteered data automatically.

The data resulting from the integration process is stored in a PostgreSQL database with the PostGIS⁷ extension, which adds support for geographic objects and allows SQL location queries. These data consist of points with water level categories for the entire region among the collected observations, and have geographic metadata to allow spatial-temporal manipulations and visualization.

We have employed the architecture implementation to conduct an experiment with the goal of testing its capacity to support automatic integration processes and to assess the flexibility that it provides. On the basis of the data resulting from the experiment, we have built thematic maps to show water level along the whole river and the points where there is a probability of flooding.

Experiment to Make Estimates of Flooding

The experiment uses a set of sensors installed at strategic points of the Monjolinho River analyzed by hydrologists (Horita *et al.*, 2014) and volunteer reports located at points where there is a great movement of people. Figure 4 shows the scenario with the spatial distribution of data samples (S_i and V_i) and places with frequent floods (A and B). The samples were based on measurements from real sensors (deployed on the banks of the River Monjolinho), volunteers reports, and simulated VGI (due to an insufficient number of volunteers and reports). We collected water level measurements from the last measure of each sensor from AGORA-DSM and data with the water level type from the past one hour from AGORA-VOS. Table 3 shows the input data used in the integration process. In determining the thresholds used in the categorization, we made use of the measurements calculated by hydrologists about the depth and water normal level at one point with frequent floods (Point A in Figure 4) and applied them to the whole river.

Id.	Value	Coordinates (lat, long)	Id.	Value	Coordinates (lat, long)	Id.	Value	Coordinates (lat, long)
S1	127.2cm	-22.00051, -47.89978	V5	N	-22.00941, -47.90503	V15	O	-22.01712, -47.91180
S2	124.7cm	-22.00241, -47.90037	V6	O	-22.00800, -47.90356	V16	L	-22.01510, -47.91026
S3	109.6cm	-22.00687, -47.90531	V7	L	-22.00580, -47.90386	V17	O	-22.00858, -47.90400
S4	98.5cm	-22.01010, -47.90625	V8	H	-22.00463, -47.90146	V18	L	-22.01341, -47.90827
S5	109.7cm	-22.01557, -47.91104	V9	L	-22.01290, -47.90737	V19	O	-22.00678, -47.90456
S6	115.4cm	-22.01775, -47.91279	V10	H	-22.00380, -47.90145	V20	N	-22.00082, -47.89980
V1	O	-22.01672, -47.91224	V11	O	-22.00358, -47.90029	V21	O	-22.00383, -47.90036

⁵ <https://www.rforge.net/rJava/>

⁶ <https://cran.r-project.org/web/packages/geoR/>

⁷ <http://postgis.net/>

V2	L	-22.01443, -47.90861	V12	H	-22.00120, -47.90042	V22	L	-22.01982, -47.91340
V3	L	-22.00536, -47.90256	V13	N	-22.01883, -47.91306	V23	L	-22.02061, -47.91350
V4	L	-22.01139, -47.90637	V14	O	-22.01816, -47.91340	V24	L	-22.00919, -47.90490

Table 3. Input Data for the Integration Process. The values in centimeters are the water levels measured by sensors. ‘L’, ‘N’, ‘H’, and ‘O’ are the water levels reported by volunteers, standing respectively for ‘Low’, ‘Normal’, ‘High’, and ‘Overflowing’. ‘lat’ and ‘long’ in coordinates stands for ‘latitude’ and ‘longitude’.

Figure 4. Reference scenario - Monjolinho River, São Carlos, Brazil

We used Kriging interpolation as the statistical method to estimate the water level along the whole river (Figure 5) and areas likely to flood (Figure 6). The interpolation was performed with a resolution of approximately one square meter. The Gaussian model was chosen by the Weighted Least Squares (WLS) estimation method as the model fit that best represents the gathered data. Both the estimation method and model fit were selected after making a preliminary review and set as parameters of Kriging.

Figure 5. Result of the integration process using Kriging to estimate water levels

Figure 6. Result of the integration process using Kriging to predict flood areas

The interpolated data have been dichotomized to predict the flood areas. The values that were in the overflowing category, *i.e.*, that exceeded the river height, were associated with the number 1, and the rest were associated with 0. This stage was conducted manually, although it is possible to customize the R script so that it can be done automatically.

Thematic maps were designed to show the results yielded by AGORA-IFM (Figure 5 and Figure 6). However, it should be stressed that this stage is not the focal point of this research since it is in the application part of the global project (Figure 1). Thus, the maps were designed manually in R from the results of AGORA-IFM.

Discussion of the Results

During the experiment we were able to execute the entire process several times because it is fully automatic, and lasted less than ten minutes for each execution. Every time we changed the integration parameters or even the whole method. This was done by simply editing the file corresponding to the script, which did not influence the rest of the process. This demonstrates the flexibility and the benefits of the automation provided by the architecture. In addition, we executed the process by using just the services about the sensor data and that about VGI, which demonstrates that the process itself is adaptable, since it can be defined by selecting the services we need.

The estimation of the water level using both sensor data and VGI provides benefits by determining water level values for areas that do not have sensor data but where there is a movement of people. It can be seen that the prediction of flood areas matches the flood areas of the reference scenario (Figure 4), and it is possible to visualize the situation of each region of the river (Figure 5). This can heighten community awareness and improve the decision-making by responsible authorities. With regard to decision-making, Figure 6 shows the areas more likely to be flooded, which can support public policies in undertaking a flood prevention more accurately, especially through urban development plans and by spending money on control measures against flooding (*e.g.*, through infrastructure schemes and vegetation). In view of this, both thematic maps can be used to prevent human and material losses, economic disruption and social problems (*e.g.*, heavy traffic and power outage) caused by floods.

This result is a simulation and cannot be generalized because there are some limitations involved. First, the thresholds must be calculated at several points of the Monjolinho River. Second, VGI must be collected through AGORA-VOS without simulation, which can be done when there are enough volunteers reports. In this case, as

the number of observations increase, the result tends to be improved. In addition, by putting into effect a continuous production of data, such as sensors and volunteers who are continuously sending data, the information concentration can be improved also over time. These limitations, however, do not disqualify the results of this proof of concept.

CONCLUSION

This paper has outlined an architecture to support the integration of sensor data and VGI with the aim of increasing the spatial-temporal coverage of information on floods that is required to improve disaster risk management.

The architectural evaluation showed that most of found risks occurred because the associated requirements were not identified before the architectural development. We now are working to include these requirements. In addition, we were able to verify that the main requirements were met and the architecture is suitable for its purposes, which was analyzed by its implementation.

AGORA-IFM is different from other systems because it can be implemented in real contexts to heighten the awareness of decision makers and communities. It also involves volunteers who submit reports, enable clients to choose the integration method that is most suitable for their needs, and perform data integration automatically and online. Thus, three main research contributions of this architecture can be summarized as follows: first, it executes its workflow automatically, and produces results faster than using manual activities, which allows it to be used in several situations. Second, it flexibly integrates heterogeneous data, by enabling the use of different spatial-temporal methods in an easily updatable way. Finally, it makes the integration process adaptable, by selection of the required components.

In future studies, we intend to consider the new requirements and extend the integration process to make it more robust. For example, this can be achieved by adding a quality factor to differentiate the reliability of sensor data in comparison with the volunteered data. In addition, it is possible to use predicted water levels to generate forecasted estimates. Furthermore, an optimization can be carried out by comparing the data of two consecutive process executions to analyze the need to update the previous results, and thus avoid unnecessary usage of resources when possible.

REFERENCES

1. Albuquerque, J. P., Herfort, B., Brenning, A. and Zipf, A. (2015) A Geographic Approach for Combining Social Media and Authoritative Data towards Identifying Useful Information for Disaster Management. *International Journal of Geographical Information Science*, 29(4): 667-689.
2. Albuquerque, J. P. and Zipf, A. (2012) Collaborative information systems for disaster management: Building resilience against disasters by combining participatory environmental monitoring and vulnerability communication. *In Proc. of the Alumni Seminar Natural Hazards*, Teresópolis, RJ.
3. Assis, L. F. F. G. d., Behnck, L. P., Doering, D., Freitas, E. P., Pereira, C. E., Horita, F. E. A., Ueyama, J., Albuquerque, J. P. (2016) Dynamic sensor management: Extending sensor web for near real-time mobile sensor integration in dynamic scenarios. *30th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, [S.l.: s.n.].
4. Bhat, M. A., Shah, R. M., and Ahmad, B. (2011) Cloud computing: A solution to geographical information systems (GIS). *International Journal on Computer Science and Engineering*, 3(2):594-600.
5. Câmara, J. H., Almeida, T. T., Carvalho, D. R., Ferreira, T. B., Balardino, A. F., Oliveira, G. V., Fonseca, F. J., Ramos, R. S., Souza, W. D., and Lisboa-Filho, J. (2014) A comparative analysis of development environments for voluntary geographical information web systems. *Proceedings of the XV Brazilian Symposium on Geoinformatics*, Campos do Jordão, SP, Brazil.
6. Clements, P., Kazman, R., Klein, M. (2002) Evaluating Software Architectures: Methods and Case Studies. [S.l.]: Addison-Wesley.
7. de Oliveira, M. G., de Souza Baptista, C., Campelo, C. E., Acioli Filho, J. A. M., and Falcão, A. G. R. (2014) Automated production of volunteered geographic information from social media. *Proceedings of the XV Brazilian Symposium on Geoinformatics*, Campos do Jordão, SP, Brazil.

8. Degrossi, L. C., de Albuquerque, J. P., Fava, M. C., and Mendiondo, E. M. (2014) Flood citizen observatory: a crowdsourcing-based approach for flood risk management in Brazil. In *The 26th International Conference on Software Engineering and Knowledge Engineering, Vancouver, Canada*, pages 570–575.
9. Erskine, M. and Gregg, D. (2012) Utilizing volunteered geographic information to develop a real-time disaster mapping tool: A prototype and research framework. In *CONF-IRM 2012 Proceedings*.
10. Flowerdew, R. (1991) Spatial data integration. *Geographical Information Systems*, v. 1, p. 375–387.
11. Goodchild, M. F. (2007) Citizens as sensors: the world of volunteered geography. *GeoJournal*, 69(4):211–221.
12. Goodchild, M. F. and Glennon, J. A. (2010) Crowdsourcing geographic information for disaster response: a research frontier. *International Journal of Digital Earth*, 3(3):231–241.
13. Horita, F. E. A., Albuquerque, J. P., Degrossi, L. C., Mendiondo, E. M., and Ueyama, J. (2015). Development of a spatial decision support system for flood risk management in brazil that combines volunteered geographic information with wireless sensor networks. *Computers & Geosciences*, 80:84 – 94.
14. Horita, F. E. A., Fava, M. C., Mendiondo, E. M., Rotava, J., Souza, V. C., Ueyama, J., and Albuquerque, J. P. (2014) AGORA-GeoDash: A geosensor dashboard for real-tim flood risk monitoring. In *Proceedings of the 11th International ISCRAM Conference*, pages 304–313, Pennsylvania, USA.
15. Kotsev, A., Pantisano, F., Schade, S., and Jirka, S. (2015) Architecture of a service-enabled sensing platform for the environment. *Sensors*, 15(2):4470–4495.
16. Lu, X. (2005) An investigation on service-oriented architecture for constructing distributed web gis application. In *Proceedings of the IEEE International Conference on Services Computing*, pages 191–197.
17. Mansourian, A., Rajabifard, A., Zoej, M. V., and Williamson, I. (2006) Using SDI and web-based system to facilitate disaster management. *Computers & Geosciences*, 32(3):303–315.
18. Mediondo, M. (2010) *Integrated Urban Water Management: Humid Tropics*, chapter Reducing vulnerability to water-related disasters in urban areas of the humid tropics, pages 109–127. CRC Press.
19. Meissen, U. and Fuchs-Kittowski, F. (2014) Towards a reference architecture of crowd-sourcing integration in early warning systems. In *Proceedings of the 11th International ISCRAM Conference*.
20. Moreira, R. d. B., Degrossi, L. C., and de Albuquerque, J. P. (2015) An experimental evaluation of a crowdsourcing-based approach for flood risk management. In *Proceedings of the 12th Workshop on Experimental Software Engineering*.
21. Munich RE (2014) Annual statistics 2014. Retrieved from <http://www.munichre.com/en/reinsurance/business/non-life/natcatservice/annual-statistics/>.
22. Poser, K. and Dransch, D. (2010) Volunteered geographic information for disaster management with application to rapid flood damage estimation. *Geomatica*, 64(1):89–98.
23. Rezeg, K., Laskri, M. T., and Servigne, S. (2010) Geospatial Web Services Semantic Discovery Approach Using Quality. *Journal of Convergence Information Technology*, 5(2):28–35.
24. Schnebele, E., Cervone, G., and Waters, N. (2014) Road assessment after flood events using non-authoritative data. *Natural Hazards and Earth System Science*, 14(4):1007–1015.
25. Schnebele, E., Oxendine, C., Cervone, G., Ferreira, C., and Waters, N. (2015) Using non-authoritative sources during emergencies in urban areas. In Helbich, M., Jokar Arsanjani, J., and Leitner, M., editors, *Computational Approaches for Urban Environments, volume 13 of Geotechnologies and the Environment*, pages 337–361. Springer International Publishing.
26. UNISDR, United Nations Office for Disaster Risk Reduction (2014) Brazil – disaster statistics. Retrieved from <http://www.preventionweb.net/english/countries/statistics/?cid=24>.
27. Walenciak, G. and Zipf, A. (2010) Designing a web processing service application profile for spatial analysis in business marketing. In *Proceedings of the AGILE Conf. on Geographic Information Science*, pages 1–8, Guimarães, Portugal.
28. Zhao, P., Foerster, T., and Yue, P. (2012). The geoprocessing web. *Computers & Geosciences* , 47:3 – 12.

A GAMIFICATION-BASED SOCIAL COLLABORATIVE ARCHITECTURE TO INCREASE RESILIENCE AGAINST NATURAL DISASTERS

A Gamification-based Social Collaborative Architecture to increase resilience against natural disasters

Flávio E. A. Horita¹, Luiz F. F. G. Assis¹, Raul E. S. Castanhari¹, Seiji Isotani¹,
Wilmax M. Cruz¹, João Porto de Albuquerque¹

¹Institute of Mathematics and Computer Science (ICMC) – University of São Paulo
(USP)

Postal Code 668 – 13.560-970 – São Carlos – SP – Brazil

{horita, luizffga, raul, sisotani, wilmcruz, jporto}@icmc.usp.br

Abstract. *Natural disasters have led to the need for urgent measures to mitigate the effects of these accidents. The question of awareness is extremely important in dealing with disasters, and collaborative systems are very serviceable for this. In this context, the aim of this work is to outline a gamification-based social collaborative architecture (GSCA) to increase resilience against natural disasters. We have combined techniques to produce and share information through the interaction of a large number of people, such as those employed in social media and serious games. This architecture was evaluated and analyzed using the Voluntary in Action, a web platform which applies the components defined in the GSCA. The results suggest that gamification is a system with a great potential to help increase the knowledge of a community about disasters.*

1. Introduction

The increasing occurrence of natural disasters around the world has attracted the attention of various global agencies (Mendiondo, 2010), and led to the adoption of a number of preventive and responsive measures (Baharin et al., 2009), such as giving priority to key information to allow management strategies to be employed to deal with these accidents (Bui et al., 2000), (Lu et al., 2006).

In assisting government agencies to cope with disaster management, there have been a number of volunteers who have provided valuable and updated information to community organizations and thus helped them in their recovery (Goodchild, 2007), (Gill and Bunker, 2012). However, it is difficult to manage young volunteers during a disaster because they usually act in a flexible and task-based manner in new social media, which means that the results of their participation can be seen immediately (Neubauer et al., 2013).

For this reason, collaborative systems have been employed to gather a large number of people together to carry out various tasks for several purposes, exchange experiences with each other and tackle serious problems in a distributed way (Doan et al., 2011).

This paper aims to outline an architecture to assist in the management of natural disasters. In undertaking this, common features found in collaborative and recommendation systems, social networks and geoinformatics are used for the dissemination of knowledge. These features are enhanced by gamification techniques which increase the interaction that occurs in the activities carried out by volunteers.

The remainder of the paper is divided into four sections: Section 2 outlines the background, Section 3 describes the Gamification-based Social Collaborative Architecture (GSCA), Section 4 shows its evaluation and analysis, and Section 5 makes some final considerations and recommendations for future work.

2. Background

2.1. Natural Disaster Management

In recent years, increasing attention has been paid to environmental risks and their potential impact on natural disasters, particularly in light of the frequency with which they occur. A natural disaster is an occasion when a society is afflicted by a natural event and the damage is so great that external assistance is required (De Longueville et al., 2010).

The events in Haiti, Chile, Pakistan in 2010, Queensland between 2010-11 and Japan and Christchurch in 2011 are examples of recent disasters. Together, they accounted for 333,944 victims and economic losses estimated at US\$199-327 billion. In addition, disasters and climate change that occurred in China in 2008 caused damage of approximately US\$111 billion to its economy (Tu et al., 2009).

One way to mitigate this damage is the adoption of countermeasures to prevent a simple event from turning into a catastrophe (Baharin et al., 2009), (Poser and Dansch, 2010). Disaster management is an important alternative to help in this kind of situation (Baharin et al., 2009).

The information that is relied on for disaster management is of crucial importance and must be accurate, timely, and complete with regard to the current state of environmental variables, so that it can be used to support predictions about upcoming changes and associated impacts (Baharin et al., 2009).

However, owing to the specific needs and preferences of each person or official agency, a good deal of information related to disasters and climate change is scattered over different areas and is disconnected of poor quality (Tu et al., 2009). As a result, several issues affect the way information is created and shared, and when critical "success factors" are analyzed, there is clearly a failure to give priority to key information required for disaster management (Bui et al., 2000), (Lu et al., 2006).

2.2. Collaborative Systems

The growth of the web has provided some important benefits, such as the creation of an environment where teams that are scattered in geographical terms can carry out activities in a collaborative manner (Bafoutsou and Mentzas, 2002). To achieve this, collaborative computing relies on computers to support coordination and cooperation between two or more people who come together to solve a problem (Borenstein, 1992), (Schooler, 1996). The study of these systems forms part of a large area of interdisciplinary research, *Computer Supported Cooperative Work* (CSCW).

Commercial products are often mentioned such as CSCW groupware. According to Ellis et al. (1991), this term more commonly refers to the computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment". On the other hand, Schooler (1996) states that the essential feature of groupware are the creation of shared workspaces among employees.

According to Bafoutsou and Mentzas (2002), in the literature there are several approaches that provide taxonomies and classifications of collaborative systems. These classifications and taxonomies are designed to separate these systems on the basis of their properties and characteristics.

2.3. Crowdsourcing and Volunteered Geographic Information (VGI)

The emergence of Web 2.0 and evolution of mobile devices that allow content creation, have become the basis for the emergence of a new paradigm, where users in general, (i.e. citizens) become established as producers of data and information (Niko et al., 2011), which in many cases, is more detailed and of a higher quality than that provided by official agencies (Goodchild, 2007), (Elwood, 2008), (De Longueville et al., 2010), (Ostermann and Spinsati, 2011).

Heipke (2010) invented the term “crowdsourcing” for this phenomenon which involves content production being carried out by a third party, that is assigned to intelligence and knowledge. It is based on the experience of volunteers, who are independent in the way they use their free time and are located in remote and diverse areas.

In a similar context, but more closely linked to geographical issues, Goodchild (2007) used the term VGI which he defined as a collection of digital spatial data produced by individuals and informal institutions, as well as by ordinary citizens who use appropriate tools to gather and publish their observations and geographical knowledge on the web.

Some of the systems which employ these concepts are Wikimapia, OpenStreetMap, Mechanical Turk, CloudCrowd, and Wikipedia, and together they compose a set of systems defined as Crowdsourcing Systems which enlist volunteers to assist in building long- term artifacts that are beneficial to the whole community (Doan et al., 2011).

In this context, Doan et al. (2011) raised some important challenges that crowdsourcing systems have to face. First, how should these systems recruit volunteers and maintain their assistance. Second, what different kinds of support can users provide and how they can be combined. Finally, although this huge amount of data is useful to obtain knowledge, it causes several problems with regard to its quality (Elwood, 2008), (Goodchild and Li, 2012).

2.4. Gamification

Deterding et al. (2011) states that the term "Gamification" was coined by the digital industry. The first record of the use of this term was around 2008, but it was most widely employed in 2010. Moreover, several parallel terms that are still being used, can be found in the literature, and new terms are still being incorporated , such as: “productivity games”, “surveillance entertainment”, “funware”, “playful design”, “behavioral games”, “game layer” or “applied gaming”. As a result, the definition of gamification is "the use of game design elements in non-game contexts".

The identification of game elements is still a question that is much discussed, and in the literature, there are a few works that offer definitions such as "classic game model" of Juul (2005) and "Ten Ingredients of Great Games" of Reeves and Read (2009). Interpretations that are either very strict or very tolerant of these elements, can make the game either very limited or unlimited. One solution provided by Deterding et

al. (2011) is to restrict the scope of the term, by only considering the elements that are found in most games, but not necessarily all of them.

2.5. Related Works

Tools and technologies based on social media concepts, have been widely employed to assist in activities and procedures related to disaster management (Pohl et al., 2012), (Horita et al., 2013). Among these, there are works whose main goal is to employ extensive data analysis techniques and information generated by Twitter, Facebook, Flickr and YouTube (Kaewkitipong et al., 2012), (Sinnappan et al., 2010), (Pohl et al., 2012), (Ahmed, 2011). There are also those based on the features that characterize crowdsourcing systems (Doan et al., 2011).

One group, in particular, stands out in the way it displays collaborative features to allow the fast creation and easy sharing of geographic content. The interactive maps created by the Ushahidi platform is one of the examples because they offer resources that can support the gathering and sharing of information with a view to solving problems emerging from an environment of uncertainty (Okolloh, 2009). Similarly, we also include platforms like Elva, Google Crisis Map, and LRA Crisis Tracker in this category. On the other hand, OpenStreetMap and Wikimapia are open technologies that provide resources for producing general street maps and georeferenced elements in a collaborative way (Haklay and Weber, 2008), (Goodchild, 2007).

In addition, there are several tasks which are hard for computer systems to execute but can be easily performed by humans, such as image recognition (Von Ahn and Dabbish, 2008). There are also technologies to recruit volunteers to conduct an analysis with the aid of aerial images taken in the affected areas like MapMill and Tomnod.

This study seeks to set out an architecture for providing resilience against natural disasters by using concepts and features of social media, collaboration, and crowdsourcing systems, all of which are combined with gamification techniques (Deterding et al., 2011) and recommender systems (Melville and Sindhvani, 2010) to increase the interaction, participation and assistance of members of communities.

3. Gamification-based Social Collaborative Architecture (GSCA) to increase resilience to natural disasters

According to Neubauer et al. (2013), voluntarism is the main pillar of disaster relief and occurs when people show a willingness to help communities without any financial self-interest. It should be noted that most volunteers are between 30 and 50 years old and married. This means there is a need to increase the participation of those younger than 30 and a good strategy is to offer flexible and task-based activities in new social media, where the results of their involvement can be seen in an immediate manner.

To address this question, the aim of this study is to develop a collaborative social architecture, with elements of gamification and social media components, called GSCA. This architecture is designed to create communities that are resilient to natural disasters, and hence reduce the extent of the damage that they cause. According to Pohl et al. (2012), these components can encourage the use of architecture, because people often use social media to document situations that they are involved in.

Elements of gamification, as well as social media, are coupled with GSCA with a view to encouraging volunteers to participate further (Deterding et al., 2011). For

example, volunteers at GSCA can take part in special missions, which include the features of games like scores, badges and rankings which reflect the abilities and achievements of the participants. This architecture is based on Crowdsourcing (Doan et al., 2011) and Volunteered Geographic Information (VGI), which provide content to support the acquisition of knowledge.

According to Goodchild (2007), formerly relevant geographic information was only provided by official agencies; however, with the emergence of Web 2.0 and evolution of mobile devices, this geographic information has often come to be shared. It is known that communities have shared a good deal of information, and although individual information was often imprecise, a collective analysis of the data has often had a considerable effect.

Three specific objectives must be attained before this architecture can be implemented into a platform: (1) Providing knowledge to communities about natural disasters; (2) Mapping the reactions of communities to natural disasters; and, (3) Assisting emergency agencies in natural disaster management.

Figure 1. Components of GSCA

As can be seen in Figure 1, the volunteers are the central component of the architecture because, as shown earlier, it is they who both evaluate and increase the amount of information available and extend its range. The four other components, communication, missions, knowledge, and profile, provide mechanisms to support the central element and the volunteers, as well as enabling them to carry out their activities. The following sections outline each of these components in detail.

3.1. Communication Component

Communication between users occurs in three ways: (1) they can exchange messages individually or collectively, synchronously or asynchronously, (2) they can create forums to discuss ideas, which should address specific issues related to natural disaster management, and (3) they can create communities by bringing together individuals with common interests.

3.2. User Profiles Component

The profile of the volunteers is characterized by personal information, their degree of interest in forums and communities, and their network of volunteers (friends). Moreover, users can find out about recent activities from other users and news related to

natural disasters.

As well as this, volunteers can participate in missions with different purposes, where they can be awarded points for their level of performance. The possession of performance skills denotes that a volunteer is able to carry out some kind of task and can thus be designated as a fireman, policeman, soldier, and so on.

A system of overall ranking is employed for all the participants, and this encourages them to become more full involved in a spirit of competitiveness (Von Ahn and Dabbish, 2008). In addition, the “missions” performed by specific users are listed chronologically in their profile to ensure more control of their activities, and this system can be applied to all the participants.

3.3. The Missions Component

This component of the architecture enables users to perform interactive activities while at the same time, allowing them to generate and acquire knowledge related to natural disasters. The goal of these activities is to encourage the participation of a large number of people as well as to ensure that they will continue to assist societies to become resilient to accidents such as floods or tornadoes.

This can be accomplished by offering the activities in the form of "missions", in which the participants volunteer to fulfill certain tasks with specific goals and rewards. These missions are conducted collaboratively with maps that represent real or fictitious regions, for example, forests, cities or risk areas. The architecture can also recommend specific missions for determined volunteers depending on their scores and proven skills.

3.4. Knowledge Component

The knowledge component is composed of three main elements: (1) news related to natural disasters, (2) exchange of experiences between volunteers and (3) missions.

News related to natural disasters serves to centralize information and keep participants updated. The exchange of experiences (related to disaster management) between participants occurs via the “Communication Component”. Finally, missions help volunteers to increase their knowledge through some tasks related to natural disaster management (as described in the “Missions Component”).

4. Evaluation and Analysis

In this section, the assessment and analysis of the outlined architecture are described in detail.

4.1. Potential Scenario: Voluntary in Action Platform

In a disaster situation, all the information related to the local context is important for faster and more accurate decision-making (Baharin et al., 2009). However, due to adverse structural and organizational factors, it is difficult for emergency agencies (EAs) to have proper information. According to Vivacqua and Borges (2012), available services enable communities to provide data about their region in a voluntary way. For example, an emergency scenario about the availability of active hydrants in a region, can be solved by using volunteer information.

However, this approach causes some difficulties. The quantity and quality of reported information depends on the number of volunteers willing to help with data related to their community (Goodchild and Li, 2012). Moreover, three important factors

arise from this: (1) personal motivation; (2) professional volunteers; and (3) the possibility of creating and sharing information maliciously or for criminal purposes (Coleman et al., 2009). Finally, there is also the need to adopt mechanisms to recruit volunteers so that they can provide useful assistance to the EAs (Vivacqua and Borges, 2012).

One solution could be achieved via the integration of gamification techniques that involve practices of collaborative systems, the main objective of GSCA. Thus, a web platform called *Voluntary in Action* is displayed as an implementation of this architecture. Figure 2 shows an example of mapping performed on the platform mission.

Figure 2. Components of GSCA

This mission allows the tagging of geographic features shown on a particular map. In the scenario outlined by Vivacqua and Borges (2012), this platform enables volunteers to navigate a map and by using specific components, create the reference points of existing elements in the real world, such as parks, schools and fire hydrants.

In addition, the volunteer can see the score of the volunteers of his team, his badge, overall ranking and ranking mission. These implemented elements are defined as functionalities of gamification. Thus, (as shown in Figure 2), by tagging the element, the volunteer earns some points and this allows him to improve his rankings and the merit of his badges, as well as encouraging him to compete with other volunteers. It also encourages him/her to participate in diverse activities and increase the quantity of georeferenced elements.

The platform also stores data related to how volunteers interact with each other using a chat line to achieve the mission. Through the platform, it is possible to detect factors that may influence the achievement of future actions related to disasters, and carry out missions as a basis to make recommendations and improvements.

In a complementary way, in addition to the knowledge generated in the mapping mission, there is also the possibility of forming data through image analysis, which is

conducted by volunteers or by shared web tools such as Google Street View. This means that Figure 3 has another mission platform, (georeferencing), which carries out an analysis of random images to define locations where there is evidence of useful elements in disaster management.

Figure 3. Georeferencing Mission

As can be seen in Figure 3, in this mission a random image is displayed, and includes a magnifying glass feature, which aims to answer some of the available questions shown on the left-hand side. In the context described by Vivacqua and Borges (2012), this question was linked to the presence or absence of fire hydrants. The gamification resources, such as rankings and scores for created elements are employed to encourage volunteers to get involved.

Figure 4 depicts the platform homepage, where users can view the scope of the project through the possible missions that they can select from the menu options of the platform, and obtain some news related to natural disasters.

Figure 4. Platform Homepage

Finally, Figure 5 displays a user profile, which contains his personal information, his performed missions in chronological order (timeline), his friends and communities, as well as his overall ranking and some missions that have been recommended to him.

Figure 5. User Profile

A timeline's mission of a user is an important element in the user profile. The timeline can help the user to manage his activities on the basis of his performed missions, and in addition, it can encourage other participants to carry out certain tasks. Moreover, this platform can also suggest missions to a user by drawing on his personal information and his performance in old missions.

5. Final Considerations

The damage caused by natural disasters has encouraged the development of systems that support the management of such events. Many of these attempts aim at providing considerable assistance to the general population and, more specifically, to people which live in risk areas. In this manner, the use of collaborative techniques has proved to be effective for solving certain complex tasks.

People can overcome traditional models of data analysis and machine learning when they work on tasks that are simple for human beings but difficult for computers and, moreover, they can achieve great results in a much shorter period of time compared with the solutions provided by more traditional techniques. The use of collaborative techniques (including gamification techniques which are used to maximize its results) provides an innovative alternative and can be used in various scenarios.

In this context, the effective participation of people can lead to communities who are resilient to natural disasters. In this paper, we have outlined a gamification-based social collaborative architecture which is designed to work in this scenario. Moreover, the information it provided for its application in the *Voluntary in Action* platform, shows the benefits of increasing the amount of information since this can be useful in assisting the activities of the emergency agencies in their preventive and reactive work.

This architecture can lead to the generation, acquisition and dissemination of necessary knowledge and thus reduce the frequency and intensity of natural disasters, by fostering health and safety awareness among people who live in a risk area.

In future work, the platform could provide specific features for professionals of emergency agencies, will be added. In this way, they can create their own missions to increase the knowledge generated by the platform, and thus eliminate the problem of a possible lack of information. Furthermore, there will be a chance to locate professionals who are working in risk areas through maps processed in real time, and thus aid the coordination of field activities.

References

- Ahmed, A. (2011). Use of social media in disaster management. In ICIS 2011 Proceedings.
- Bafoutsou, G., and Mentzas, G. (2002). Review and functional classification of collaborative systems. *International Journal of Information Management* 22, 4, 281 – 305.
- Baharin, S., Shibghatullah, A., and Othman, Z. (2009). Disaster management in Malaysia: An application framework of integrated routing application for emergency response management system. In *International Conference of Soft Computing and Pattern Recognition (SOCPAR)*, pp. 716 – 719.
- Balabanovic, M., and Shoham, Y. (1997). Fab: content-based, collaborative recommendation. *Commun. ACM* 40, 3, pp. 66–72.
- Borenstein, N. S. (1992). Computational mail as network infrastructure for computer-supported cooperative work. In *Proceedings of the 1992 ACM conference on Computer-supported cooperative work, CSCW '92*, ACM, pp. 67–74.
- Bui, T., Cho, S., Sankaran, S., and Sovereign, M. (2000). A framework for designing a global information network for multinational humanitarian assistance/disaster relief. *Information Systems Frontiers* 1, 4, 427–442.
- Coleman, D. J., Georgiadou, Y., Labonte, J., Observation, E., and Canada, N. R. (2009). Volunteered Geographic Information : the nature and motivation of producers. *International Journal of Spatial Data Infrastructures*.
- De Longueville, B., Luraschi, G., Smits, P., Peedell, S., and De Groeve, T. (2010). Citezens as sensors for natural hazards: a VGI integration workflow. *Geomatica* 1.
- Deterding, S., Dixon, D., Khaled, R., and Nacke, L. (2011). From game design elements to gamefulness: defining "gamification". In *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments* (New York, NY, USA, 2011), ACM, pp. 9–15.
- Doan, A., Ramakrishnan, R., and Halevy, A. Y. (2011). Crowdsourcing systems on the world-wide web. *Commun. ACM* 54, 4, 86–96.
- Ellis, C. A., Gibbs, S. J., and Rein, G. (1991). Groupware: some issues and experiences. *Commun. ACM* 34, 1, 39–58.
- Elwood, S. (2008). Volunteered geographic information: future research directions motivated by critical, participatory, and feminist GIS. *GeoJournal* 72, 3-4, 173–183.

- Goodchild, M. F. (2007). Citizens as sensors: the world of volunteered geography. In *GeoJournal*, pp. 1–12.
- Goodchild, M. F., and Li, L. (2012). Assuring the quality of volunteered geographic information. *Spatial Statistics* 1, 0, 110–120.
- Haklay, M., and Weber, P. (2008). Openstreetmap: User-generated street maps. *Pervasive Computing, IEEE* 7, 4, 12–18.
- Heipke, C. (2010). Crowdsourcing geospatial data. *ISPRS Journal of Photogrammetry and Remote Sensing* 65, 6, 550–557.
- Horita, F. E. A., Degrossi, L. C., de Assis, L. F. G., Zipf, A., and de Albuquerque, J. P. (2013). The use of volunteered geographic information (VGI) and crowdsourcing in disaster management: a systematic literature review. In *19th Americas Conference on Information Systems (AMCIS)*.
- Juul, J. (2005). *Half-Real: Video Games between Real Rules and Fictional Worlds*. The MIT Press.
- Kaewkitipong, L., Chen, C., and Ractham, P. (2012). Lessons learned from the use of social media in combating a crisis: A case study of 2011 Thailand flooding disaster. In *Thirty Third International Conference on Information Systems*.
- Lu, D.-K., Pettit, S., and Beresford, A. (2006). Critical success factors for emergency relief logistics. *WHAMPOA - An Interdisciplinary Journal* 51, pp. 177–184.
- Mendiondo, E. M. (2010). Reducing vulnerability to water-related disasters in urban areas of the humid tropics. In *Integrated Urban Water Management Humid Tropics*, Paris, France, pp. 109–127.
- Melville, P., and Sindhvani, V. (2010). *Encyclopedia of Machine Learning*. Springer, ch. Recommender Systems, pp. 829–838.
- Niko, D. L., Hwang, H., Lee, Y., and Kim, C. (2011). Integrating user-generated content and spatial data into web gis for disaster history. *Computers, Networks, Systems, and Industrial Engineering* 2011.
- Neubauer, G., Nowak, A., Jager, B., Kloyber, C., Flachberger, C., Foitik, G., and Schimak, G. (2013). Crowdtasking – A New Concept for Volunteer Management in Disaster Relief. *International Symposium on Environmental Software Systems*, 345–356.
- Okolloh, O. (2009). Ushahidi or “testimony”: Web 2.0 tools for crowdsourcing crisis information. *Participatory Learning and Action* 59, 65–70.
- Ostermann, F. O., and Spinsanti, L. (2011). A Conceptual Workflow For Automatically Assessing The Quality Of Volunteered Geographic Information For Crisis Management. In *The 14th AGILE International Conference on Geographic Information Science*.
- Pohl, D., Bouchachia, A., and Hellwagner, H. (2012). Supporting crisis management via sub-event detection in social networks. In *2012 IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*.
- Poser, K., and Dransch, D. (2010). Volunteered geographic information for disaster management with application to rapid flood damage estimation. *Geomatica* 64, 1, pp. 89–98.

- Reeves, B., and Read, J. L. (2009). *Total Engagement: Using Games and Virtual Worlds to Change the Way People Work and Businesses Compete*. Harvard Business School Press.
- Schooler, E. Conferencing and collaborative computing. *Multimedia Systems* 4 (1996), pp. 210–225.
- Sinnappan, S., Farrell, C., and Stewart, E. (2010). Priceless tweets! a study on twitter messages posted during crisis: Black Saturday. In *ACIS 2010 Proceedings*.
- Tu, Y., Li, Q., and Liu, R. (2009). A geospatial information portal for emergency management of natural disasters. In *2009 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, vol. 2, pp. 404–407.
- Vivacqua, A. S., and Borges, M. R. (2012). Taking advantage of collective knowledge in emergency response systems. *Journal of Network and Computer Applications* 35, 1, 189 – 198.
- Von Ahn, L., and Dabbish, L. (2008). Designing games with a purpose. *Commun. ACM* 51, 8, pp.58–67.