Towards a Microservices development approach for the Crisis Management field in Developing countries

Djilali Idoughi, Karima Ait Abdelouhab, Christophe Kolski

4th International Conference on Information and Communication Technologies for Disaster Management, ICT-DM 2017 (December 11-13), IEEE, Münster, Germany, pp. 1-6, ISBN 978-1-5386-2553-8.

Towards a Microservices development approach for the Crisis Management field in Developing countries

Djilali Idoughi, Karima Ait Abdelouhab,
Univ. Abderrahmane Mira of Bejaia
Bejaia, 06000, Algeria
{djilali.idoughi, aitabdelouhab}@univ-bejaia.dz

Christophe Kolski
LAMIH-UMR CNRS 8201
Univ. Valenciennes and Hainaut-Cambrésis
F-59313 Valenciennes 9, France
Christophe.kolski@univ-valenciennes.fr

***Abstract*— This article outlines a first attempt toward aligning IT assets and business processes within the crisis management (CM) field in Developing Countries. We propose to provide service designers/developers with a comprehensive framework to the design, implementation and deployment of microservices based software systems. The proposition is illustrated by a case study in Algeria.**

Keywords—Microservice; monolith; architecture; framework; SOA; Service design; software, crisis management.

I. INTRODUCTION

Many developing countries have been engaging and deploying much effort and investments in new types of Information and Communication Technologies across the whole country. Such effort is also being made in the software development industry. However, in many cases, IT still approaches software development in a traditional deployment way (waterfall, large teams, etc.) resulting in unclear requirements, huge costs and more risks of failure or delays. With the need for the digital transformation move of the most societies of those countries on one hand and the rise of cloud infrastructures and ubiquitous computing on the other hand, this has made the technological context even challenging for the software designers/developers in almost all the business and government domains, such as the crisis management (CM) domain for example. Sometimes emergency management is also used to designate synonymously disaster, crisis, catastrophe with slight differences, by scholars and practitioners [1].

From the business and ground field perspective, the CM is an important and particular domain. It is a special type of human complex organization in which the communication and collaboration between several types of actors become major management issues as it is pointed out in [2, 3]. And from the technological view, it is a big challenging due to its underlying heterogeneous and interoperability [4] cross boundaries features. Moreover, there are also some major requirements: (Technological, Process, Information, Collaboration and Visualization) to consider. In this paper, we outline a first attempt towards aligning IT assets and business processes within the crisis management field when encountered in developing countries and we propose to provide designers/developers with a comprehensive framework to the design, implementation and deployment of microservices based software systems.

The remainder of the paper is as follows. Section II gives an overview of the background of the microservices architecture or philosophy relative to software development discipline as well as some motivating design issues. In section III, we highlight the main phases of the proposed design framework as well as its main founding principles. In section IV, we illustrate the proposed approach on a typical case study relative to a Civil Protection Department (CPD) organization in Bejaia, Algeria. Finally, in section V, we dress some discussion issues and ideas relative to our proposal, whilst concluding and suggesting some research perspectives in section VI.

II. BACKGROUND AND MOTIVATIONS

This section dresses some relevant background on CM in developing countries and explains the major ideas and principles of microservices motivating our methodology.

A. About the crisis management (CM) domain

Many research and practical work for designing new crisis management systems were proposed in the literature. [5, 6] have related one design approach combining User Centered Design (UCD) principles [7, 8] agile characteristics and SOA paradigm [9],

for complex software development in the CM field. [10, 11] proposed an approach for Collaboration in Emergency Management Systems in Brazil. [2] presented a system design and development framework that addresses the communication and information decision making needs. [3] proposed "Sahana", an open source software for disaster management, evaluating how the disaster information system coordinates disparate institutional and technical resources in the wake of the Indian Ocean tsunami. In a recent work, [12] has tackled the User eXperience (UX) concept in Crisis Management Services in developing countries and proposed the User Experience Design of Interactive Services (UXD-IS) framework.

Some of the proposed approaches have suggested to evolve from monolithic architectural style of an application to a more appropriate one that encounters new software design requirements within the highly complex and agile IT environment. However, there is still another yet step to move forward such as breaking down services in a service-oriented architecture into microservices as pointed up by Fowler [13]. It is argued that microservices came about to help solve the frustrations developers were having with large applications that require change cycles to be tied together. Moreover, with the advent of cloud computing, according to the specialized literature, SOA seems to be lacking scalability and slowing down with work request changes, limiting application development. Where, some developers claim that microservices are just a more granular approach to service-oriented architecture; some others, however, consider microservices architecture as the natural evolution of SOA. So, we can see in microservices another yet platform-agnostic approach to application development and where SOA lives on in the layers of microservices management.

1) Monolithic application context in ICT-CM: Broadly, the (CM) activities can be grouped into three main phases: (1) preparedness, (2) response and (3) recovery. In developing countries, the communication and collaboration between several types of actors become major challenging issues [2] [3]. It becomes even more complex when many relevant interconnected systems and diverse types of information exchanged between them are concerned. These systems are generally based on business functions and placed within different siloed departments of the organization. The silo applications engender several problems, namely: the information redundancy and its processing, difficulties to reuse software components belonging to different applications of organization, inability to have a unified view of the organization's business processes, etc. In addition, this leads to other difficulties related to Human-Computer Interaction (HCI) involving different usage scenarios and new possibilities inherent to the mobility of human actors, cooperative tasks and communication devices [14].

2) The multi-layered view of ICT-CM system: Despite of the big investments made in modernizing their IT assets, the software development industry in these developing countries is still far away from the desired achievement of aligning the IT assets and their underlying organizations' businesses. Most of IT systems and Information systems are still being developed following some still traditional and conventional approaches even for those highly IT based and multi-layered systems architectures.

B. On microservices architecture

1) Microservices - Definition: There is no formal definition of the microservices architectural style. [15] defines "microservices architecture as a method of developing software applications of a suite of independently deployable, small, modular services in which each service runs a unique process and communicates through a well-defined, lightweight mechanism to serve a business goal." Unlike microservices, a monolith application is always built as a single, autonomous unit, where in a client-server model the server-side monolith application handles the HTTP requests, executes logic, and retrieves/updates the data in the underlying database.

Thus, a modification made to a small section of the application might require building and deploying an entirely new version and scaling specific functions of the application, would lead to scale the entire application instead of just the desired components.

2) The microservices architectural style has been proposed to cope with inherited problems of monolithic ones. Usually, a typical SOA model depends on more ESBs, whereas microservices use faster messaging mechanisms. Moreover, SOA also focuses on imperative programming, whereas microservices architecture focuses on a responsive-actor programming style. Finally, SOA models tend to have an outsized relational database, while microservices frequently use NoSQL or micro-SQL databases (which can be connected to conventional databases). The main founding characteristics of a microservices architecture are: Unique functionality, Technological flexibility, Reduced development team.

3) Agile & UCD practices in microservices design process: The Agile manifesto [16] promotes four key values: (1) individuals and interactions over processes and tools, (2) working software over comprehensive documentation, (3) customer collaboration over contract negotiation and (4) responding to changes over following a plan. Moreover, Agile methods are incremental, cooperative, and adaptive [17]. They encourage rapid and flexible response to changes by emphasizing on user involvement and his/her feedback, and on delivery of several small releases. Agile methods are being studied with action research in the context of rapid development and fielding of response oriented EMIS [10]. The collaboration between users and developers is an important aspect of UCD to building interactive software solutions, each one bringing their experience to bear. UCD is a

philosophy that tries to understand the users and their tasks. It is also an iterative approach increasing the chances of delivering a successful project. Consequently, we argue that there is a need and much remains to be done towards bridging the gap between existing monolithic applications design and envisioned modern IT based ones in order to provide designers/developers with a comprehensive framework for the design, implementation and deployment of microservices based CM applications.

C. Need for approaches leveraging the microservices concept in the development of ICT-CM software systems

Usually, most application development efforts use a project model which delivers completed piece of software. And on completion, the software is handed over to a maintenance organization and the project team that built it is disbanded [15]. The microservices approach envisions to building complex CM applications out of primitive services that are by themselves relatively simple. Thus, ICT-CM software systems need to be broken into simpler components. But the question is "how to divide up the pieces and what are the principles on which we decide to slice up our ICT-CM applications?"

We argue that this approach which organizes cross-functional teams around services, which in turn are organized around business capabilities can be one way to enhancing the design and development of software dedicated to the CM field. Moreover, this will bring agility to SOA based CM empowering more agile development practices rather than the enterprise-wide reuse of standard SOA.

III. A MICROSERVICES DESIGN FRAMEWORK FOR THE CRISIS MANAGEMENT DOMAIN

The following section outlines the characteristics of the CM as our research application domain to illustrate the design challenges, thus the motivations and requirements for an appropriate approach. We highlight the founding elements of our proposal, a methodological framework based on microservices architectural style.

A. The founding principles of the proposal

The proposal consists of a design framework which considers three main design views or levels (Fig. 1) when tackling an innovative CM project in developing countries. These are: CM phases; classical software engineering phases and specific microservice phases.

The microservice lifecycle is different from a traditional software development lifecycle in which there are some additional steps to consider. Moreover, the framework combining agile characteristics and user-centered design techniques focuses mainly on the sprint key points (short time cycles management, articulating between individual and collective work, motivating work for all and user meeting which are the main concern of a UCD method). That is, a sprint is considered to be a first step of an innovative and agile project. When it is validated after positive feedbacks and user tests, this can be followed by a deeply study on business, organizational and technical aspects. The approach consists to splitting up business processes organized around business capabilities into services. Such services take a broad-stack implementation of software for that business area, including user interface, persistent storage, and any external collaboration.

The development teams are cross-functional and with full range of required skills: user-experience, database, and project management [18]. The cross functional teams are responsible for building and operating each application and splitting it out into a number of individual communicating services. A particular development team should own the microservice over its full lifetime which is inspired from the "you build, you run it" amazon's principle [19]. The application calls many services to collect data and construct the visualization page for the user. One important benefit of the microservice approach is that teams are driven more by business scenarios than by technology. Smaller teams develop a microservice based on a user based scenario and use any technologies they choose.

B. The global view of the framework

The Fig. 1 illustrates an approach dividing design and development into two separate stages. It encourages separating the design stage from the development stage, and approaching it iteratively using agile development principles. This would allow development teams to continuously release software microservices. Continuous delivery lets business stakeholders verify, in real time, that an application is meeting the ultimate business objective.

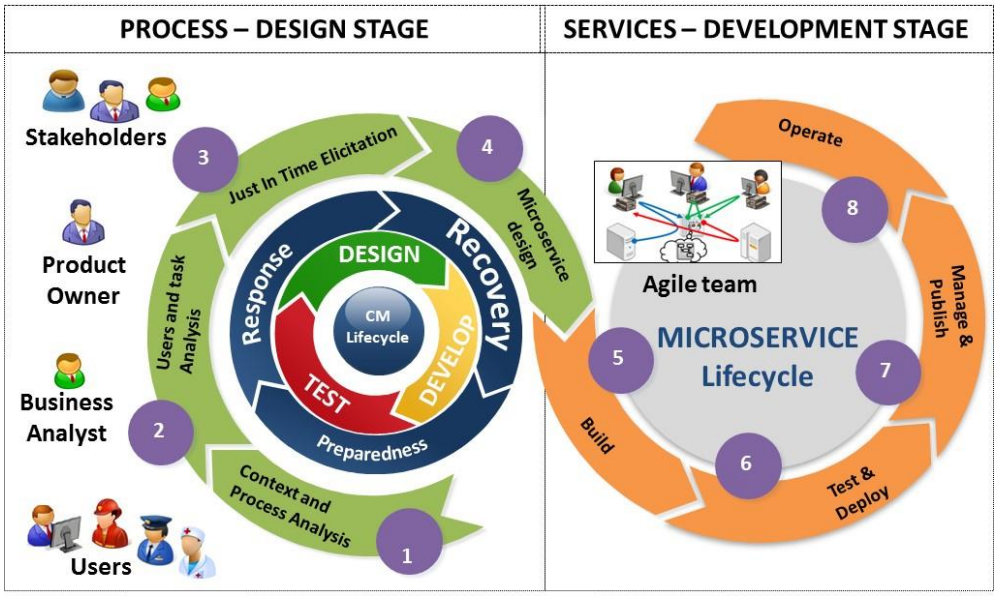

Fig. 1. Global view the CM-microservices design framework

C. Main phases

Phase 1. Context and Process Analysis

This phase considers the study of the complex organization in order to identify: (1) all the stakeholder's requirements (2) business objectives, and (3) to understand and communicate the business and ground field environment context in which the targeted microservices are to be developed (Fig. 2). The business domain is decomposed into functional areas giving rise to business use cases activities.

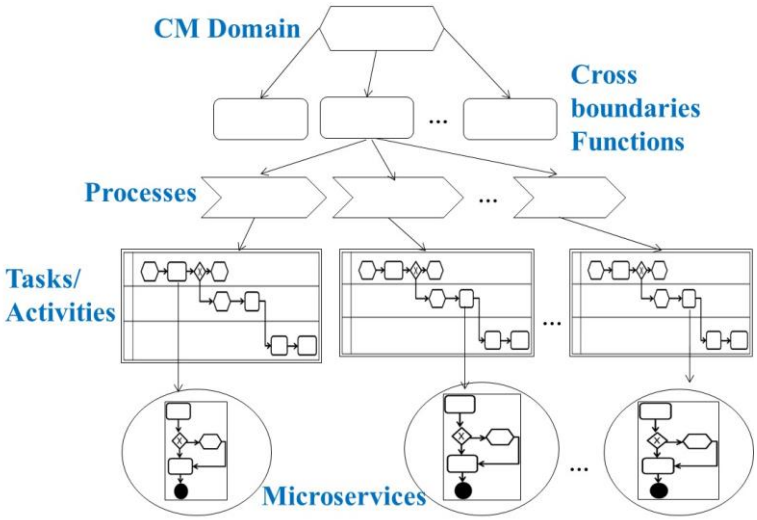

Fig. 2. The splitting up business processes model into microservices

Phase 2. Users and tasks Analysis

A task is a set of interdependent business services. Each task is broken down into several business services, themselves broken down into microservices with a unique functionality; each microservice follows its lifecycle development such that it is developed, deployed and maintained separately. The contexts for use of the service and the channels through which the users, employees, stakeholders, external partners interact with the service are also elicited. Each task should be tailored and exposed to suit the needs of each business channel and digital touchpoint (mobile, web, etc.).

Phase 3. Just-In-Time Requirements Elicitation

The aim is to define requirements Just-in-Time when they are needed. They are identified and expressed in terms of user stories [20] which are an effective way to understand the user's needs because they focus on the goal of the user, and the value the user expects from the use of the microservice. The user stories are often written by the user, thus integrating the user directly in the development process. Moreover, they include the role of the user and the activity he/she wishes to perform towards the achievement of some user goals, in the context of some constraints, such as acceptation test. Each user story encapsulates the whole knowledge (role, business goals, business value, acceptation test) about the potential user of the service [21]. The service designer uses User Goals (supported by business use cases) in order to identify and to describe business microservices relative to an actor.

Phase 4. Microservice development life cycle

The main steps are: (1) *Build*, using a standardized specification such as RAML, Swagger or ApiBlueprint, (2) *Test & Deploy*, using mockups and getting user feedback, (3) *Manage*, (4) *Publish* & *Operate*. This phase describes best practices in structuring development teams, team organization and responsibilities, automated testing, and continuous delivery of microservices in line with business and field practices. We also use best practices of agile development such as: coding standards, code ownership, continuous integration, continuous testing and refactoring, etc. Teams should be small enough to work locally together and focus entirely on a single sub-domain of the business, and include domain experts so that the language of that sub-domain is modelled in the solution. The ownership of the microservice includes everything from design to deployment and management.

IV. A CASE STUDY AND EMPRICAL VALIDATION

We will go through a typical crisis management case study in a specific developing country: Algeria. This project, at its preliminary stage, aims at developing relevant crisis management microservices dedicated to the Civil Protection Department (CPD) organization of the city of Bejaia.

The project management team members, the Head of civil protection department, the relief operations commander, the intervention planning office manager, the head of operational coordinating center, and the information transmission officer were involved in the design project.

A. Overview of the CPD organization

Cross-boundaries systems are connected to the CPD of the city of Bejaia and with its higher decisions authorities. The CPD is one of the main organizations involved in any crisis. It has cross-functional services and domains such as the general protection service, the main emergency fire and rescue unit service, the administration and logistics unit service and several other external services management. It also maintains a national system of prevention, preparedness and response to any disaster that could affect the population.

B. A typical crisis management scenario

Fig 3 illustrates broadly a typical CM scenario expressing the main roles/responsibilities specific to the emergency response function. These are: Request resources**,** Allocate, delay, or deny resources, Report and update situation, Analyze situation, Edit, organize, and summarize information, Maintain resources (logistics), Acquire more or new resources, Assign roles and responsibilities when needed, Coordinate among different resource areas. This scenario is presented to demonstrate how it is complex and hard to come out with microservices without a consistent and comprehensive approach to do so.

C. Empirical validation of the design framework

We have conducted an empirical validation of our framework on a typical but fully complex CM scenario. For the sake of place, we only illustrate partially the outcome of some phases. And similarly, the rest of the work would be carried out in an agile manner and iteratively till the deployment of the concerned microservices.

Phase 1. Context and Process Analysis

We have studied the IT environment context within which the presented scenario is executed. We also carried out a process analysis of the targeted organization. The Operation of disaster process is detailed and decomposed into sub-processes that will be as microservices candidates for the next phase.

Phase 2. Users and task Analysis

In this phase, we have conducted a user and task analysis and we come out with the appropriate support of various roles, such as first responders, command-and-control personnel, healthcare professionals, and various experts.

Phase 3. Just-In-Time Requirements Elicitation

Fig. 4 illustrates how, from a user story, some requirements relative to "*reservation resources for intervention*" are expressed by the chief of zone along with activities to perform. Those activities are translated to microservices.

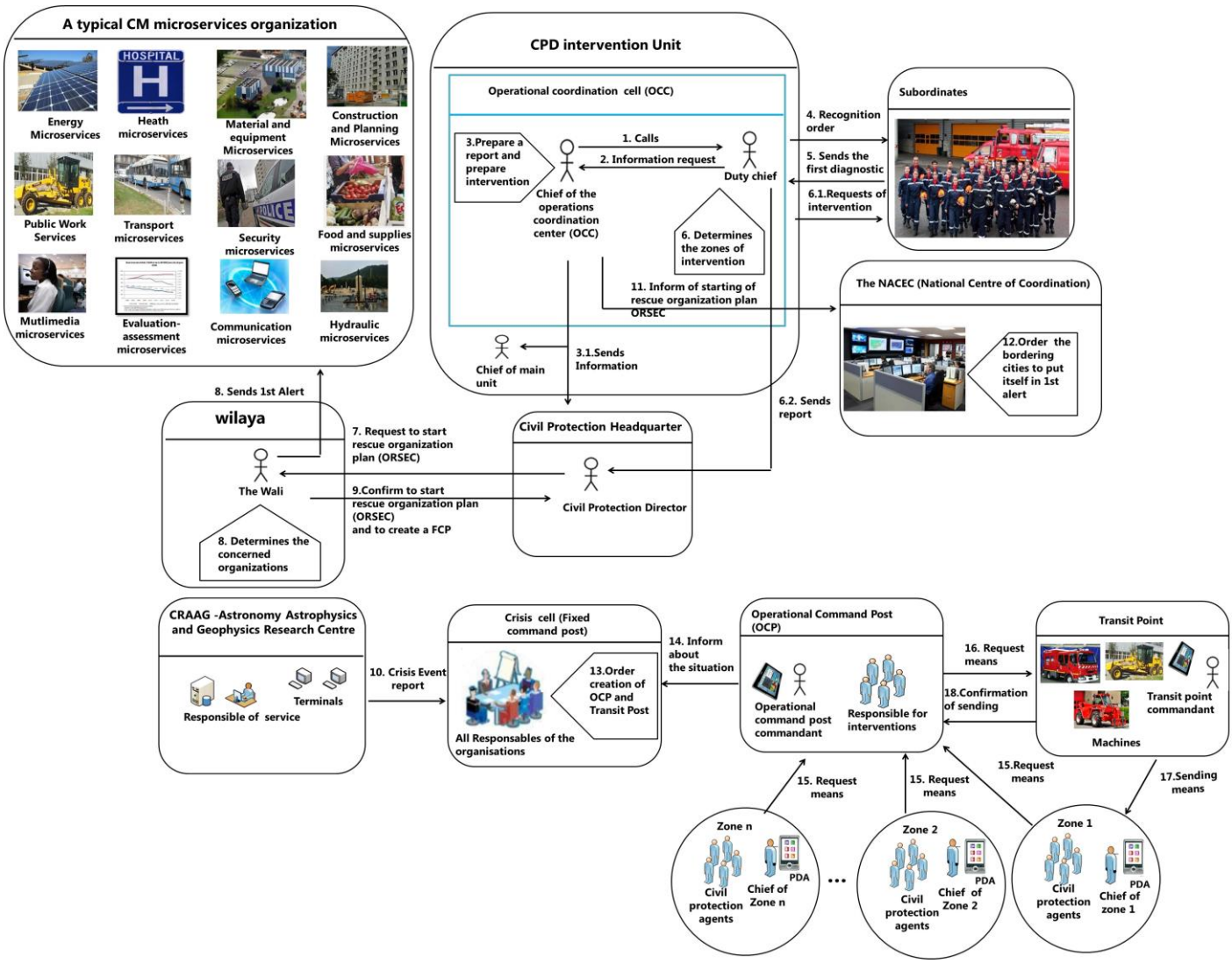

Fig. 3. A typical crisis management scenario

Phase 4. Microservice development life cycle

Fig. 5 shows that a single microservices API (Application Programming Interface) gateway can create multiple APIs—one for each platform we need to support (smartphone native applications, browsers, and server-side applications). Each of which requires a different set of microservices features that may require different protocols. After Code Review, API Testing, and deploying the microservices, we may have the mockups and rendering of the services. The development teams have been reduced to be small enough to work locally together focusing entirely on a single microservice.

V. DISCUSSION

From the designer's perspective, this approach seeks to make a software based microservices assist its users to enhance the business capability due to the smaller granularity of services. This can make it easier to create the personal relationships between service developers and their users. From the technology perspective, it is not only improving by using services.

From the service design perspective, the services model has been a key enabler in creating teams that can innovate quickly with a strong user focus. Each service has a team associated with it, and the development team is completely responsible for the service—from scoping out the functionality, to architecting it, to building it, and operating it. The approach can bring value to the business and field activities because it can be adapted for use in multiple contexts. Moreover, a service that is built and operates at scale to reach the users in new geographical regions can be delivered faster with features and capabilities to be able to respond to human users' demands in an agile way.

From the CM domain perspective in developing countries, the changing business and organization needs are affecting how we build applications and impact the factor of team skills. The approach suggests also that the geographically dispersed expertise can be captured and transcribed into microservices that can be discovered and used when needed by different stakeholders.

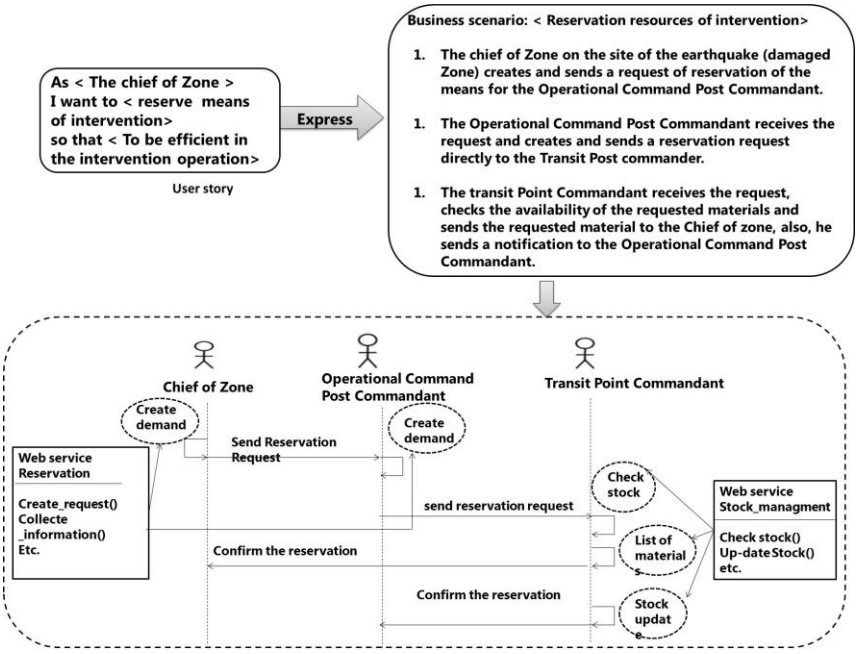

Fig. 4. The user story microservices correspondance

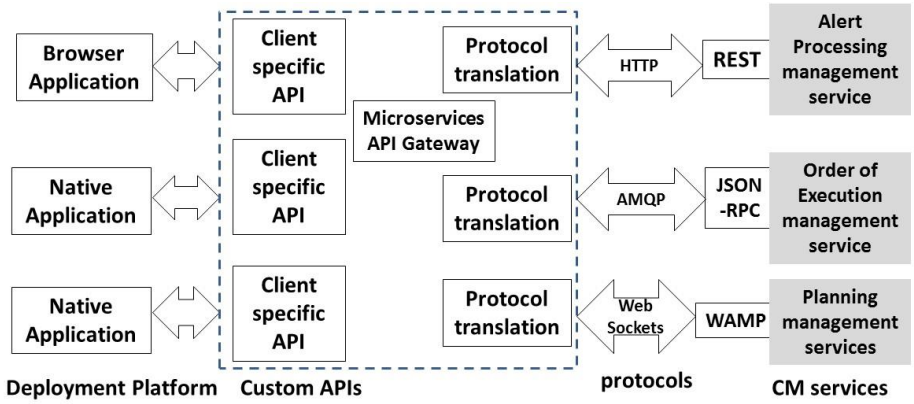

Fig. 5. Examples of designed microservices

VI. CONCLUSION

We have proposed an agile process combining UCD and service oriented paradigm for the development of microservices applied to the CM domain. We have outlined a first attempt toward aligning IT assets and business processes within the crisis management field in Developing Countries.

The proposition has been illustrated by a CM case study relative to the direction of the Civil Protection of Béjaïa (Algeria). A major benefit of the proposed framework is that it leads to highly flexible and agile software based microservices that should be able to meet rapidly changing business needs.

Finally, as a research perspective, we tend to go further towards implementation and deployment of the designed microservices in collaboration with different departments.

ACKNOWLEDGMENT

The authors gratefully acknowledge and express their warm thanks to the *Direction Générale de la Protection Civile de la wilaya de Béjaia, Algéria* and the University of Bejaia.

REFERENCES

[1] V. Zwass, "Series Editor's Introduction," In B. Van de Walle, M. Turoff, and S.R. Hiltz, eds., Information Systems for Emergency Management. Volume 16, Advances in Management Information Systems (Armonk, NY: M.E. Sharpe, 2010), pp. 23–45.

[2] M. Turoff, M. Chumer, B. Van de Walle, and X. Yao, "The Design of a Dynamic Emergency Response Management Information System (DERMIS)," The Journal of Information Technology Theory and Application (JITTA), 5:4, 2004, pp. 1-35.

[3] P. Currion, C. De Silva, and B. Van De Walle, "Open source software for disaster management," Comm. Of The ACM, 2007/Vol. 50, No. 3.

[4] X. Ferre, "Integration of Usability Techniques into the Software Development Process," In Proc. of the Int. Conference on Software Engineering, ICSE'2003, May 3-10, Portland, Oregon, pp. 28-35, 2003.

[5] K. Ait Abdelouhab, D. Idoughi, and C. Kolski, "A Framework combining Agile, UCD and SOA approaches for Collaborative Disaster Management system Design," In International Journal of Information and Communication Technology (in press), Inderscience.

[6] K. Ait Abdelouhab, D. Idoughi, C. Kolski, "Agile & user centric SOA based service design framework applied in disaster management," ICT-DM'2014, 1st IEEE International Conference on Information and Communication Technologies for Disaster Management, March 24-25.

[7] E. Not, C. Leonardi, C. Mennecozzi, F. Pianesi, and M. Zancanaro, "Beyond Usability: A New Frontier for User-Centered Design of "Future Internet" Services," LNCS 5468, pp. 107–116, Springer, 2009.

[8] ISO 13407, "Human-centered design processes for interactive systems," International Standard,1999.

[9] D. Groves, "Successfully planning for SOA," *BEA* Syst. World., 2005.

[10] A. Fruhling and G.J. de Vreede, "Field experiences with eXtreme programming: developing an emergency response system," Journal of Management Information Systems, 22, 4, 2006, pp. 39–68.

[11] M. Borges, "ICTs for Collaboration in Emergency Management Systems," IEEE International Conference on Information and Communication Technologies for Disaster Management, March 24-25.

[12] K. Touloum, D. Idoughi, and A. Seffah (2017), "User Experience in Service Design: A Case Study from Algeria", IT Professional, vol. 19, no. 1, pp. 56-58, Jan.-Feb., doi:10.1109/MITP.2017.1

[13] M. Fowler and J. Lewis, Microservices, 2014. http://martinfowler.com/articles/microservices.html.

[14] D. Idoughi, M. Kerkar, and C. Kolski, "Towards new web services based supervisory systems in complex industrial organizations: basic principles and case study," Computers in Industry 61, 2010, pp. 235–249

[15] VOGELS, www.acmqueue.com, 2006.

[16] Agile Alliance, "Manifesto for Agile Software Development," Technical report, 2001, Accessible at: http://www.agilealliance.org.

[17] P. Abrahamsson, O. Salo, J. Ronkainen, and J.Warsta, "Agile software development methods: Review and Analysis," Espoo, Finland, Technical Research Centre of Finland, VTT Publications 478, 2002.

[18] C. Mike, "User Stories Applied for agile software development", Publisher: Addison-Wesley Professional, march 2004

[19] A. Ivanyukovich, G. R. Gangadharan,V. D'Andrea, and M. Marchese, "Towards a service-oriented development methodology," Journal of Integrated Design and Process Science, vol. 9, no 3, 2005, pp. 53-62.

[20] B. Losada, M. Urretavizcaya, and I. Fernández-Castro, "A guide to agile development of interactive software with a User Objectives'-driven methodology," Science Computer Prog., vol. 78, 2012, pp. 2268–2281.

[21] K. Touloum, D. Idoughi, and A. Seffah, "User eXperience in service design: Defining a common ground from different fields," Proc. AHFE International Conf., 21-25 July 2012, San Francisco, pp. 2991-3003.