An Approach to Include Web-based Interaction Into Systems of Systems

M. A. Barcelona[1,2], G. López-Nicolás[1,2], E. Meléndez[1] and F. Callao[3]
[1]*ITAINNOVA, Aragón Institute of Technology, c/ María de Luna 7, Zaragoza, Spain*
[2]*IWT2 Research Group, Universidad de Sevilla, Av. Reina Mercedes s/n, 41012 Sevilla, Spain*
[3]*MEDITERRANEO ENVIRONMENTAL GROUP, c/ Botiguers 3 Of. 1-J, Valencia, Spain*

Keywords: Web Engineering, Model-driven Engineering, Interaction Flow Modeling Language, Systems of Systems.

Abstract: Emerging applications are composed by (semi)autonomous subsystems in a decentralized, distributed and heterogeneous environment, known as Systems of Systems (SoS). In this paper, we present a Web Engineering approach in order to include Web-based interaction into SoS. To this end, a model-based paradigm is used, by the analysis and integration of an interaction modeling language and the generation of Web-based runtime components. The feasibility of the approach is analyzed by a real-world case study in the design and development of a selective collection of waste tool.

1 INTRODUCTION

From the first monolithic programs, based on predefined functions and autonomous behavior, the development of systems have had a great evolution in the latest years. Emerging applications are not just large-scale and complex. They are also characterized by decentralized, distributed, networked compositions of heterogeneous and (semi)autonomous elements (Samad and Parisini, 2011). These new systems are, in fact, Systems of Systems (SoS) (Johnson, 2009). Although most of first SoS were developed in the context of embedded solutions with no human interaction, Web-based technologies have evolved as the facto standard to add human interaction into existing SoS.

Web Engineering is concerned with establishment and use of sound scientific, engineering and management principles and disciplined and systematic approaches to the successful development, deployment and maintenance of high quality Web-based systems and applications (Murugesan et al., 2001).

In the area of distributed systems, Service-Oriented Computing (SOC) is the computing paradigm that utilizes services as elements to support the development of rapid, low-cost and easy composition of distributed applications even in heterogeneous environments (Papazoglou, 2003). SOC is currently the most-used paradigm for distributed and SoS development.

Model-Driven Software Engineering (MDE) is a software development paradigm which focuses on creating and exploiting domain models, abstract representations of knowledge and activities that govern a specific domain application, as a means of alleviating the complexity and express domain concepts effectively (Schmidt, 2006).

In this paper, we present a Web Engineering approach in order to include Web-based interaction into SoS. To this end, a model-based paradigm is used, by the analysis and integration of an interaction modeling language and the generation of Web-based runtime components. The feasibility of the approach is analyzed by a real-world case study in the design and development of a Web-based tool for selective collecting waste domain.

This paper is structured as follows. Section 2 presents the background, namely the existing *CERVANTES* approach as well as the motivation of current study. Section 3 exposes the steps done in order to integrate the Web-based interaction into the framework. A real-world case study is detailed in Section 4. Finally, Section 5 concludes and outlines future work.

2 BACKGROUND AND MOTIVATION

In 2001 we set out to create a methodological approach, named *CERVANTES* (Barcelona et al., 2014), for the design and development of distributed SoS

Barcelona, M., López-Nicolás, G., Meléndez, E. and Callao, F.
An Approach to Include Web-based Interaction Into Systems of Systems.
In *Proceedings of the 12th International Conference on Web Information Systems and Technologies (WEBIST 2016) - Volume 1*, pages 336-341
ISBN: 978-989-758-186-1
Copyright © 2016 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

based on these assumptions: i) it was an architecture-driven development so that current and future functionalities should be considered in the architectural design; ii) the design of the subsystems should be independent of technology, platforms, programming languages or communication protocols; iii) the internal behavior of each subsystem must be (semi)formally described giving rise to subsequent validation, and; iv) runtime logs will have valuable information about usage of the system in order to reconfigure a deployment or redesign to improve performance, even dinamically.

Since its definition, *CERVANTES* has been widely used for the systems design and development within the group, being present in many solutions, which are currently deployed in different domains and real environments. In 2014 we evolved the proposal by using a MDE approach (García-Borgoñón et al., 2014). Nowadays it's composed by three elements: i) an architecture-driven methodology for the development of SoS; ii) a metamodel (which is shown in Figure 1) and a design-time tool in order to model SoS and performs automatic code generation, and; iii) a Runtime Framework, developed in different specific platforms and languages (namely C#.NET, Java and C++) to support the deployment and execution of generated code.

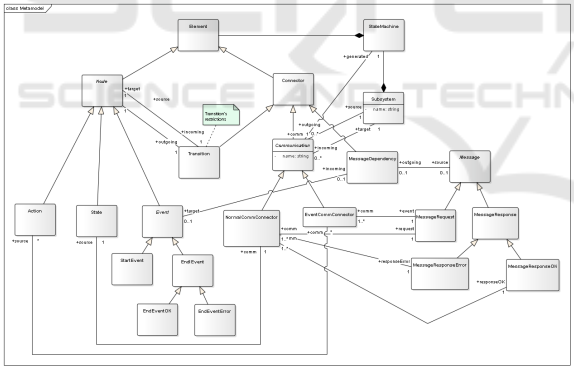


Figure 1: *CERVANTES* metamodel.

In the application of the proposal in practice we have evidenced the need to include human interaction into different subsystems, issue that has been solved by the development of graphical-user interfaces (GUI) components from scratch, by using several technologies as desktop, mobile or Web-based applications.

This way the motivation of this study is to try to solve the lack of an integrated model-based approach that covers, not only the modeling and generation of different SoS but also the human interaction. In particular, our purpose is to include Web-based interaction into *CERVANTES* based SoS.

3 METHODOLOGY

This section shows the methodological approach followed to solve the described problem which consists of three issues: i) the analysis and integration of an interaction modeling language; ii) the development of the Web-based interaction elements, and; iii) the integration of new Web widgets into the existing Runtime Framework. Each issue is shown in the following subsections.

3.1 Interaction Modeling

The first need was to use, extend or create a interaction modeling language in order to, following a model-based approach, model the Web-based interactions of a SoS. Our motivation was not to create a new proposal but to try to identify a language that may be adopted by industry in practice. Although the initial purpose was to model Web-based interaction, in future work desktop or mobile based interactions will be required, so that a platform independent model would be better than a specific one.

A review of the state-of-the art was performed in order to identify potential proposals to be adopted by our approach. Some initiatives have been created regarding conceptual modeling of sofware applications, as: i) The Web Modelling Language (WebML) (Ceri et al., 2007) for data-intensive Web applications; ii) Web Application Extension for UML (WAE) (Conallen, 2002), a UML extension for describing Web application interfaces and teh client-server interactions; iii) WebDSL (Groenewegen et al., 2008), a domain-specific language to define entities, pages and business logic.

A summary of related work is presented by Brambilla et al. (Brambilla et al., 2014) who cited more proposals focused on Web interfaces as well as some approaches that apply model-driven techniques for multi-device user interaction modeling as TERESA (Berti et al., 2004), MARIA (Paterno et al., 2009), MBUE(Meixner et al., 2011), UsiXML (Vanderdonckt, 2005) and UCP (Raneburger et al., 2011).

The Interaction Flow Modeling Language (IFML) supports the specification of the front end of applications independently of the technological details of their realization. It addresses the composition and context of the view, the commands, actions, effects of interaction and parameter binding (Brambilla and Fraternali, 2014). The Object Management Group (OMG) (OMG, 2016) adopted the IFML (OMG, 2015) as a standard in July 2014.

IFML supports all requirements for our purpose and, as it is a standard supported by OMG, we have

selected as modeling language to be included into our approach. The *CERVANTES* methodology uses Enterprise Architect as Integrated Development Environment (IDE), so that we have developed a set of UML Profiles in order to create model editors based on these metamodels, namely the IFML Profile (Figure 2) and the *CERVANTES* Profile (Figure 3).

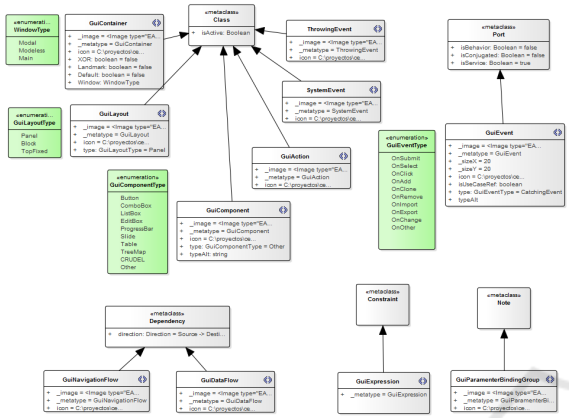


Figure 2: IFML Profile.

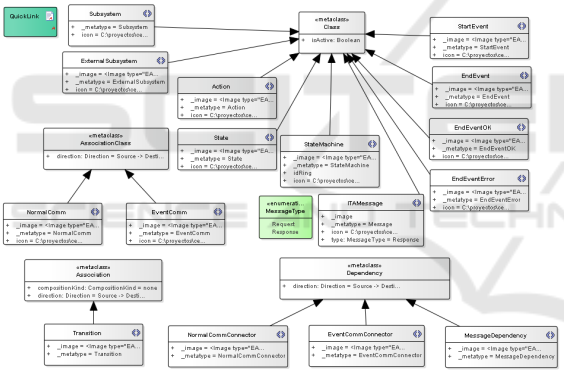
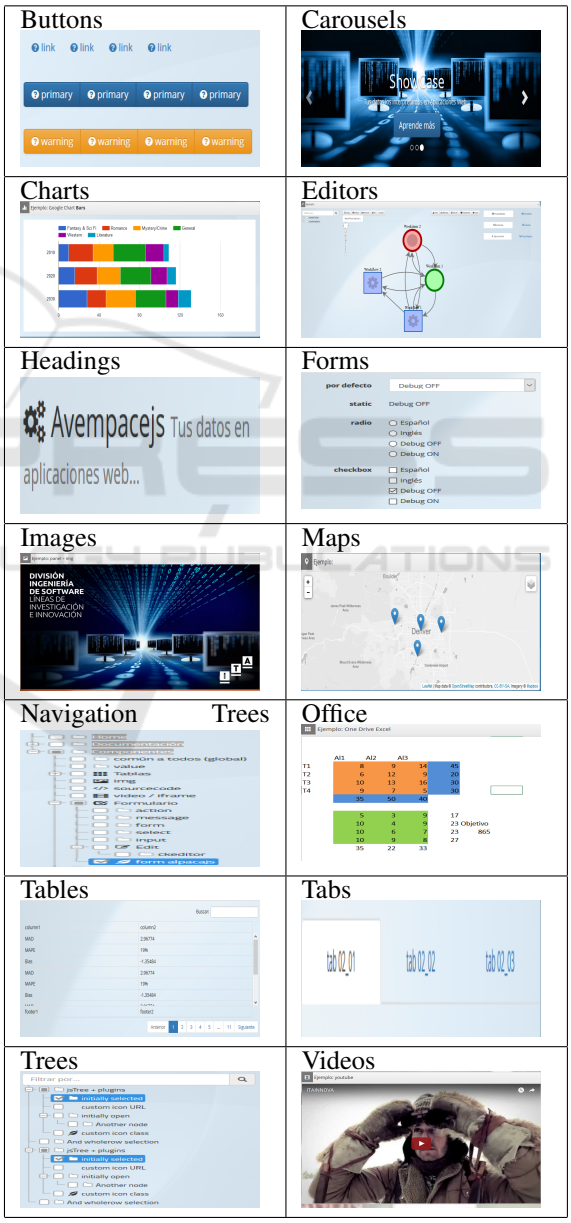


Figure 3: *CERVANTES* Profile.

3.2 Web-based Interaction

The second step was related to the execution of Web-based components. For this purpose, a Web Runtime Framework known as *Avempacejs* has been developed and integrated in the *CERVANTES* approach. This Framework is based on main Web standards as HTML5 and CSS3 and it's been implemented by Javascript language. With this Framework a single page is created and contents are change depending on the user interaction over the widgets and Web-components, by using AJAX communications in JSON or REST protocols. It's composed by a set of Web widgets designed to perform the Web interaction with the end-user of a system. These components are divided into three categories: i) Web containers: their purpose is to composite other components, like panels, blocks, divisors or tables; ii) Web contents and interactions: they manage the way data are exposed to the end-user and handle the interaction by events or actions, like buttons, inputs, selects, trees, menus, tables, graphics or multimedia, and; iii) Hidden elements: used to handle events or actions among several components. Some of the Web-components developed in this Framework are exposed in Table 1.

Table 1: Some of the widgets included in the Avempacejs Framework.



3.3 Runtime Integration

The *CERVANTES* approach includes a Runtime Framework (García-Borgoñón et al., 2014), a basic infrastructure that supports the SoS execution. As this framework did not contain any human interaction module, we have extended this component with a set of classes to handle interaction, by using IFML concepts at platform independent level. Figure 4 shows the list of new elements added into the infrastructure, that handle requests and responses to different specific platform technologies (i.e. desktop, Web, mobile).

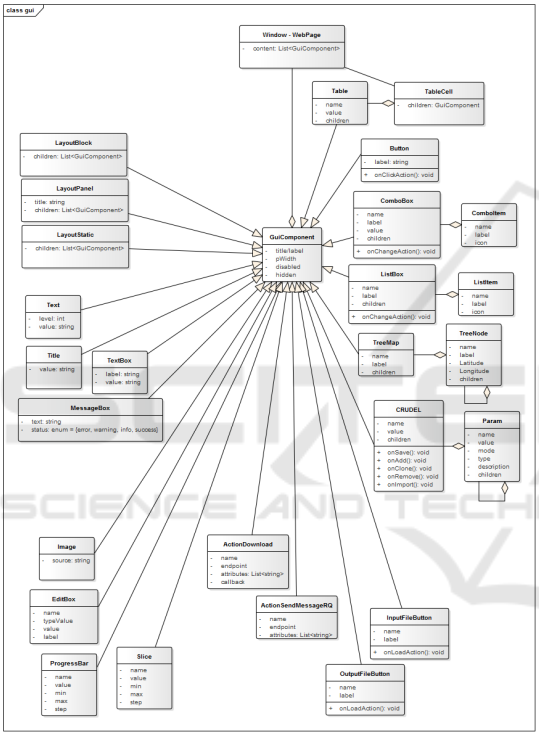


Figure 4: Extension into the *CERVANTES* Runtime Framework to include Web-based interaction.

The Web-based integration is achieved by using JSON message interchanges, as it's shown in Figure 5. Once a message is received, a specific JSON endpoint handles the request and it's translated into the common message system, as it was incoming from any other subsystem. When it's processed, the response message is formatted into the JSON and sent to the Web client. The serialization and de-serialization process is done automatically by the *CERVANTES* system, and may be extended to cover different interaction technologies and communication protocols.

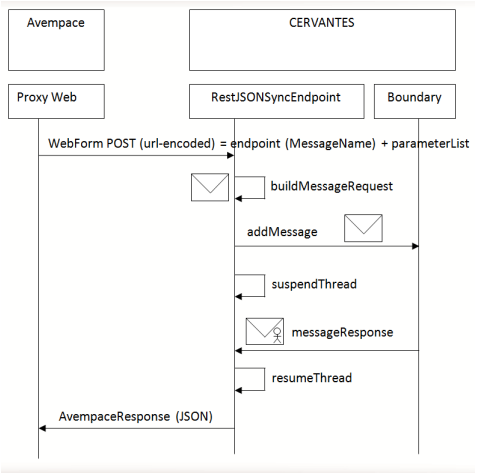


Figure 5: Runtime integration of *CERVANTES* and Web-based components.

4 CASE STUDY

The feasibility of the approach has been analyzed by a real-world case study in order to design and develop a selective collection of waste (SECOWA) tool (ITAINNOVA and MEDENGROUP, 2015). We have followed the *CERVANTES* approach in the modeling of the subsystem even when the first version of the product did not contain a pure SoS architecture, something that may be included in future increments.

This way, the first step was to define the subsystems model according to our approach, as it is shown in Figure 6, where a main SECOWA subsystem interacts with a Web-client by the interchange of a set of *Messages* according to the request and response pattern. By using a Model-To-Text transformation included in the *CERVANTES* IDE, corresponding C#.NET code was automatically generated, regarding *Entities*, *Messages* and *State Machines* (Barcelona et al., 2014).

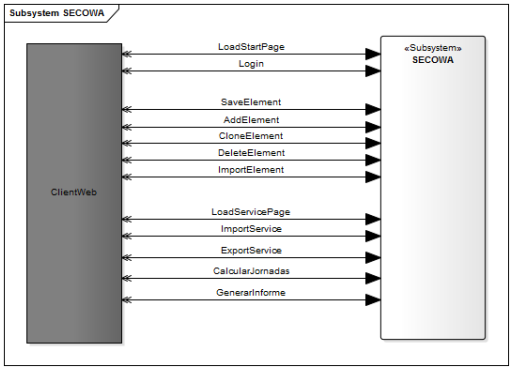


Figure 6: Case Study *CERVANTES* subsystems model.

As the SECOWA subsystem requires Web-based interaction, according to our proposal to make use of IFML standard, the next step was to create the IFML model, as Figure 7 shows. Main forms or windows, controls, events and the interaction flow between different connectors were included and redefined with the end-user, before generating any line of code. As the Model-To-Text transformation from IFML model was not available, this task was performed in order to get the practical experience of modeling the interaction flow by using IFML with the end-user, and make sure that all information included in the model is enough to develop the transformation to achieve code generation. Then all *GUIElements* component according to the concepts described in Section 3.3 were coded manually.

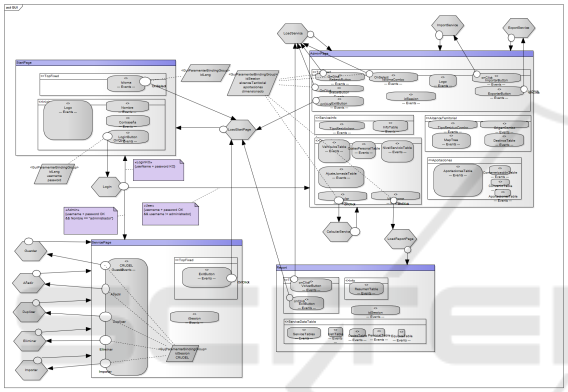


Figure 7: Case Study IFML model.

As a result of the case study a complete Web-based solution was developed by using the approach exposed in this paper. A screenshot of the final version is shown in Figure 8.

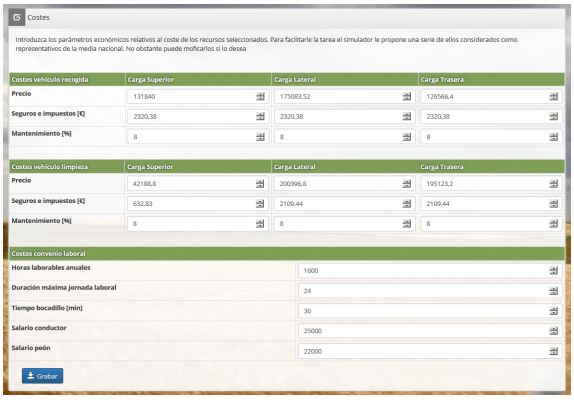


Figure 8: Case Study Screenshot.

5 CONCLUSIONS AND FUTURE WORK

This paper presents a Web Engineering approach in order to include Web-based interaction into SoS. Taking the *CERVANTES* methodology for SoS development, the purpose of this study was to try to solve the lack of an integrated model-based approach that covers, not only the modeling and generation of different SoS but also the human interaction. In particular, our purpose was to include Web-based interaction into *CERVANTES* based SoS.

To solve this problem we have covered three areas: i) the analysis and integration of an interaction modeling language; ii) the development of the Web-based interaction elements, and; iii) the integration of new Web widgets into the existing Runtime Framework.

After a review of related work regarding interaction modeling languages, as IFML supports all requirements for our purpose and, as it is a standard supported by OMG, it's been selected as modeling language to be included into our approach. A specific IFML Profile has been developed and integrated in the *CERVANTES* IDE.

The Web-based interaction was achieved by the development of a set of widgets and plugins integrated in the *Avempacejs* Runtime Framework, based on main Web standards as HTML5 and CSS3 and implemented by Javascript language. With this Framework a single page is created and contents are change depending on the user interaction over the widgets and Web-components, by using AJAX communications in JSON or REST protocols.

The *CERVANTES* Runtime Framework has been extended by a human interaction module that contains IFML concepts at platform independent model. These elements handle requests and responses to different specific platform technologies (i.e. desktop, Web, mobile). The first implementation covers Web-based interaction which is achieved by using JSON message interchanges.

Finally, The feasibility of the approach has been analyzed by a real-world case study in order to design and develop a selective collection of waste tool. A *CERVANTES* subsystem model has been created as well as the IFML model. Main forms or windows, controls, events and the interaction flow between different connectors were included and redefined with the end-user, before generating any line of code. By using a Model-To-Text transformation included in the *CERVANTES* IDE, corresponding C#.NET code was automatically generated. As this transformation does not cover the new interaction module, all *GUIElements* components were coded manually.

In conclusion, we have proposed and analyzed the application of an approach that uses IFML to include Web-based interaction into *CERVANTES* based SoS. Some features are going to be added in new increments and are included as future work as follows. Firstly, the Model-To-Text transformation in order to generate automatically the *GUIElements* from the IFML model. Secondly, as IFML model is platform independent, these transformations will be improved to cover not only Web-based interaction but also mobile and desktop applications. Finally we plan to create a basic IFML model from each subsystem according to its interface, by using a Model-to-Model transformation, so that each *CERVANTES* component may be emulated by user interaction.

ACKNOWLEDGEMENTS

This research has been supported by: the MeGUS project (TIN2013-46928-C3-3-R), the SoftPLM Network (TIN2015-71938-REDT) of the Spanish Ministry of Economy and Competitiveness and the Aragón Region European Social Fund.

REFERENCES

Barcelona, M. A., García-Borgoñón, L., Calvo, J. I., and Escalona, M. J. (2014). CERVANTES: Un framework para el diseo y desarrollo de sistemas distribuidos. In *Jornadas de Ingeniera del Software y Bases de Datos (JISBD), Cadiz, Spain, Croatia, September 16-19, 2014*.

Berti, S., Correani, F., Mori, G., Paternò, F., and Santoro, C. (2004). Teresa: a transformation-based environment for designing and developing multi-device interfaces. In *CHI'04 extended abstracts on Human factors in computing systems*, pages 793–794. ACM.

Brambilla, M. and Fraternali, P. (2014). *Interaction Flow Modeling Language: Model-Driven UI Engineering of Web and Mobile Apps with IFML*. Morgan Kaufmann.

Brambilla, M., Mauri, A., and Umuhoza, E. (2014). Extending the interaction flow modeling language (ifml) for model driven development of mobile applications front end. In *Mobile Web Information Systems*, pages 176–191. Springer.

Ceri, S., Matera, M., Rizzo, F., and Demaldé, V. (2007). Designing data-intensive web applications for content accessibility using web marts. *Communications of the ACM*, 50(4):55–61.

Conallen, J. (2002). *Building Web applications with UML*. Addison-Wesley Longman Publishing Co., Inc.

García-Borgoñón, L., Barcelona, M. A., Calvo, J. I., Ramos, I. M., and Escalona, M. J. (2014). CERVANTES: A model-based approach for service-oriented systems development. In *Information Systems Development: Transforming Organisations and Society through Information Systems - Proceedings of the 23rd International Conference on Information Systems Development, ISD 2014, Varaždin, Croatia, September 2-4, 2014*.

Groenewegen, D. M., Hemel, Z., Kats, L. C., and Visser, E. (2008). Webdsl: a domain-specific language for dynamic web applications. In *Companion to the 23rd ACM SIGPLAN conference on Object-oriented programming systems languages and applications*, pages 779–780. ACM.

ITAINNOVA and MEDENGROUP (2015). Secowa tool, http://www.ita.es/secowa/.

Johnson, M. (2009). System of systems standards. systems of systems engineering-principles and applications. m. jamshidi.

Meixner, G., Seissler, M., and Breiner, K. (2011). Model-driven useware engineering. In *Model-Driven Development of Advanced User Interfaces*, pages 1–26. Springer.

Murugesan, S., Deshpande, Y., Hansen, S., and Ginige, A. (2001). Web engineering: A new discipline for development of web-based systems. In *Web Engineering*, pages 3–13. Springer.

OMG (2015). Interaction flow modeling language 1.0, http:// www.omg.org/spec/ifml/.

OMG (2016). Object management group, http://www.omg.org.

Papazoglou, M. P. (2003). Service-oriented computing: Concepts, characteristics and directions. In *Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on*, pages 3–12. IEEE.

Paterno, F., Santoro, C., and Spano, L. D. (2009). Maria: A universal, declarative, multiple abstraction-level language for service-oriented applications in ubiquitous environments. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 16(4):19.

Raneburger, D., Popp, R., Kavaldjian, S., Kaindl, H., and Falb, J. (2011). *Optimized GUI generation for small screens*. Springer.

Samad, T. and Parisini, T. (2011). Systems of systems. *The Impact of Control Technology*, pages 175–183.

Schmidt, D. C. (2006). Model-driven engineering. *COMPUTER-IEEE COMPUTER SOCIETY-*, 39(2):25.

Vanderdonckt, J. (2005). A mda-compliant environment for developing user interfaces of information systems. In *Advanced Information Systems Engineering*, pages 16–31. Springer.