SenSoMod – A modeling language for context-aware mobile applications

Julian Dörndorfer[1], Christian Seel[1] and Daniel Hilpoltsteiner[1]

[1] University of Applied Science Landshut, Computer Science, Landshut, Germany
{julian.doerndorfer,seel,daniel.hilpoltsteiner}@haw-landshut.de

Abstract. The success and ubiquity of mobile devices like smartphones and tablets changed the daily work activities of many employees and employers. For example, editing and managing customer data in a mobile version of a customer relationship management (CRM) system can be done anywhere and anytime. In addition, the sensors of the mobile devices make it possible to gather data about the current state of the business process execution and the user's environment. With these data provided by mobile devices and combined with external data from databases or web services, it is possible to recognize the context of a business process. Context recognition allows to adapt the business process by selecting the next process step or providing the necessary data, like the next customer near the current location. However, to use the advantages of context recognition, mobile business processes and applications have to be designed in a context-aware manner.

Keywords Context, Context-Aware Business Processes, Mobile Processes, Domain Specific Modeling Language

1 Introduction

Mobile devices like smartphones or tablets, are increasingly common for professional related activities [1, 2], such as waiters using a smartphone to accept the order from customers and billing the meal with it. With the emerging generation of digital natives, which are increasingly entering the job market, this change from stationary to mobile computer is likely to grow [1]. The widespread use of mobile devices is leading to a change in executing business processes.

Using business processes to standardize important procedures is a standard approach in theory and practice [3–7]. It leads to cost reductions and productivity gains as well as it serves to increase the customer satisfaction or sharpening the competitiveness of a company [8]. The nearly ubiquitous presence of mobile devices can be used to support the execution of business processes. FALK and LEIST [9] find that the use of mobile devices improves the quality and flexibility of business processes, and also saves time and costs during the execution. A second aspect is that mobile devices are providing many sensors or can be additionally equipped via *Bluetooth*, the *universal serial bus* (USB) or other proprietary protocols. Moreover, mobile devices are capable to request

Multikonferenz Wirtschaftsinformatik 2018,
March 06-09, 2018, Lüneburg, Germany

additional data from other sources, like databases or web services via their internet connection.

The aggregation and interpretation of sensor data enables detecting and creating the context of the users and supports them during the execution of a business process. According to DEY [10] context is "any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves". Therefore, context recognition can be used to automatically present useful information for the user and moreover pre-select the next possible process steps. Context recognition also improves the efficiency of business processes [11]. To support the design and implementation of supportive mobile context-aware applications, the business process languages also have to consider contextual influences [12]. In addition, the aggregation of context through the sensor data can be complex. For instance, the context parameter *weather* basically consists of the sensor data *humidity*, *wind speed* and *temperature*. Moreover, *weather* can also be a sensor for another context parameter, like *street conditions* in a navigation system. To use the opportunities of context recognition via mobile devices and ease the conduction of mobile context-aware business processes, this paper answers the following research questions.

RQ.1: How can a modeling language for context-aware applications be designed?
RQ.2: Can the language improve the implementation of mobile context-aware applications?

The remainder of this paper is structured as follows: In section 2 a brief overview of the existing literature will be given. Afterwards, the whole modeling approach is presented, which explains the sensor modeling language (SenSoMod) in detail (**RQ.1**). Section 4 shows the evaluation of SenSoMod (**RQ.2**). The paper ends with a conclusion and outlook to further research in context-aware mobile applications.

2 Related Work

Besides DEY context was also defined by others [13–15]. Not only is DEY's definition well known and accepted in the scientific community, but he also declares when an application is context-aware. It is context-aware "if it uses context to provide relevant information and/or services to user, where relevancy depends on the user's task" [16].

Some approaches to make business processes more flexible have been tried. ROSEMANN et al. [17] claim that modeling languages have to be more flexible to model context. Further, they state that an increased attention on flexibility took place in the research area, which leads to a decreasing time-to-market for products [18]. Therefore, the result is a demand for higher process flexibility [19]. In particular, ROSEMANN et al. show the limitation of the actual event-driven process chain (EPC) language and the lack of supporting context modeling. However, they do not present an appropriate way to integrate the identified context in business processes. The extension C-EPC is an approach to make EPC more configurable for decisions at runtime, but it doesn't address context in particular [20]. The approach from LA VARA et al. aims to reveal all

possibilities of a business process and integrate them into one process model. However, this leads to large and complex models. In [21] an approach to identify and apply context in a business process is introduced, but it is more of a theoretical framework to identify context and does not show how a context-aware business process could be designed. In [22] HEINRICH and SCHÖN mention that business processes have to consider "not-static" context events that change the process execution. Moreover, they present an algorithm which supports automated process planning for context-aware processes, but no modeling representation in BPMN. CONFORTI et al. present an approach to manage process risks by sensor evaluation [23]. Furthermore, they show a way to model these risks and when they occur. They only consider sensor data and context evaluation for process risks, but context and its evaluation can be used for more than risk analysis for business processes. DÖRNDORFER and SEEL present in their paper a BPMN extension for business processes, which is able to model context in business processes via a complete modeling technique [24]. They also developed a context-free grammar to state brief context expression for decisions depending on context. Furthermore, two extensions for UML are published by AL-ALSHUHAI and SIEWE. In the first paper, they are extending the class diagram with additional annotations [25]. The second paper [26] expands the activity diagram to mark context-sensitive areas or sequences. Additionally, two paper were found about domain specific languages for mobile applications [27] and for cloud computing [28]. However, both languages aim to ease the transformation of mobile and hybrid applications between different platforms. They do not present a context description language.

All the presented papers do not consider how complex context evaluation is. Furthermore, the implementation of supportive mobile applications is not supported by the modeling languages. Therefore, this paper presents a modeling language that shows how context parameters, which influence business processes, can be extracted from sensor information. Thus, this paper presents in terms of the design sciences paradigm by HEVNER [29] a new artifact.

3 The specification of the domain specific modeling language SenSoMod

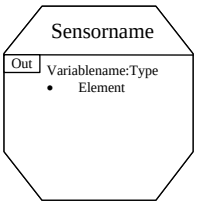
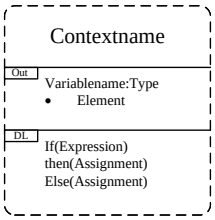
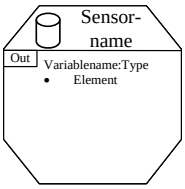
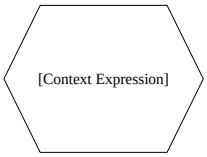
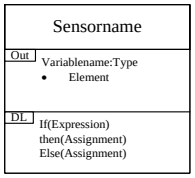
To use the possibilities of mobile devices more efficiently and ease the planning and implementation of supportive context-aware applications, standard business process languages have to consider context. Therefore, we chose the Context4BPMN [24] extension, which allows the design of context-aware business processes. The extension Context4BPMN enables to consider static and non-static context events in BPMN via a context expression, which can be stated through context annotations. It is therefore possible to design context-aware business processes. However, the extension of the BPMN does not specify how context can be evaluated by sensors. For instance, the context *location* has to be measured via GPS, with the values *at home* and *at work*. The difficulty is to state where *home* is and when to switch between these values. To address these questions and to further ease the design and implementation of mobile context-aware applications, which support the execution of context-aware business processes,

the sensor modeling language (SenSoMod) has been developed. It also allows to model context expressions from Context4BPMN and is therefore a possibility to integrate SenSoMod into BPMN.

Notation of the Sensor Modeling Language

Context is measured through sensors. We understand a sensor as any source of information for context. This can be a "usual" physical sensor – like a hygrometer or a temperature sensor –, a database, or an application from which information could be requested. Even a machine in an assembly hall that is accessible through a network connection can be a source of information. Therefore, different types of sensors have to be distinguished. There are atomic sensors which cannot be aggregated from other sensors. Two kinds of atomic sensors exist: *Physical sensors*, which measure physical quantities like temperature or humidity, and *virtual sensors* which are dedicated to non-physical quantities like databases, machines or stock states. Besides the atomic sensor, there is the computed sensor – a sensor relying on other sensors, which could be atomic sensors or other computed sensors. For example, the sensor *weather* could be the combination of the atomic sensors *humidity* and *temperature*. To address the different kind of sensor types we gave each of them a graphical representation (cf. **Table 1**).

Table 1. The notation of the sensor modeling language

Element	Notation	Element	Notation
Physical atomic sensor		Context	
Virtual atomic sensor		Context description	
Computed sensor		Flow	
Multiple Instances	III		

Knowing that the main feature to distinguish modeling constructs is the shape [30] (like the BPMN mainly relies on it [31] because shapes have a significant impact on object recognition [32]), the goal was to design similar objects like the sensor types

alike. Objects which are not similar like sensor and context description were designed in different 2D-shapes. The *physical sensor* is designated to depict physical quantities, whereas the *virtual sensors* are dedicated to non-physical quantities. To differentiate these two atomic sensors, the virtual sensors are marked by a database symbol to the left of the sensor name field. Beneath the name is the output area marked with *Out*. The area is intended to describe the structure of the outgoing objects of the sensor like the type and elements of it. First the name and – separated by a colon – the type of the return object have to be stated. If the object type has specific values – like enums, arrays or lists –, those can be stated in a bullet list. Alternatively, the return values can be specified in the JavaScript Object Notation (JSON) RFC-7159 [33] notation. Next to the two atomic sensors is the *computed sensor*. It can be used to combine different atomic sensors and/or *computed sensors*. Weather, for example, can be aggregated from different types of sensors. *Computed sensors* also have an area for describing outgoing elements. Furthermore, they have a *DL* area dedicated to describing the decision logic for their outgoing objects. This is necessary to express when a certain state from the *Out* area of a sensor will be returned. For example, location *at work* will be returned when the logged-in network name is *companyNetwork*. The accurate description of the decision logic language is presented in **Figure 1**. The *context* notation is the next element in the table. The name of the element has to match with the name of the involved context in the *context description.* Only *context* elements can be connected with a *context description*. Like the sensors, the *context* has the *DL* and *Out* areas. A *context* is based on at least one sensor, whatever type it is. The *context description* is the graphical representation of an expression to describe a contextual influence in a business process. The context expression language is part of the Context4BPMN extension [24]. To connect the different elements and show the sequence stream, the *flow* element has to be used. *Multiple instances* is the last element in the table and is an attribute for any type of sensors. It indicates that a sensor occurs in more than one instance, and is placed to the right of a sensor name. An example, using the introduced elements, is given in **Figure 2**. **Figure 1** depicts the logical decision language which has been developed. The language is a context free grammar in the Extended Backus Naur Form (EBNF) [34] and serves to briefly and precisely express the decision in the *DL* area of the *sensor* and *context* elements. A decision logic consists of a *LogicTerm* and can have a default assignment. The *LogicTerm* itself is built out of an *Expression* and an *Assignment*. If the expression is evaluated as *true*, the variable will be assigned to the designated value. The assigned variable has to exist in the output area of the computed sensor or context.

To shorten some basic definitions, like integer numbers or date, we link with "-->" to standards, like doubles. In order to express the structure of the object returned to a request by a sensor, the "Out" area is provided. First the name and – separated by a colon – the type of the return object have to be stated. Optionally, the values of the type (if existing) could be stated in a bullet list. Alternatively, the return values could be specified in the JSON notation.

An example for a mobile context-aware business process are traveling salesperson in a company's sales department (**Figure 2**). They partly work at the customer side to sell the products from the company as well as in the company to participate in meetings or further training. Moreover, some paper work can also be done at home. Therefore,

the first context parameter is the *location* with the three variables *at the customer*, *at the office* and *at home*.

```
<Decisionlogic>::=  <LogicTerm>  ["else("  <Assignment>  ")"]  |  <LogicTerm>",
   "<LogicTerm> ["else(" <Assignment> ")"]
<LogicTerm>::= "If(" <Expression> ")then(" <Assignment> ")"
<Assignment> ::= <Variable> "=" <Value>
<Expression>::= <Variable> [<MathematicalOperator> <Constant>]
   <Comparison> <Value> | <Expression> <LogicOperator> <Expression>
<Comparison>::= "=" | "!=" | "<=" | ">=" | "<" | ">"
<MathematicalOperator>::= "*" | "/" | "+" | "-"
<LogicOperator>::= "&&" | "||"
<Constant>::= --> "DoubleNumber"
<Interval>::= -->"IntegerNumber" "msec" | "sec" |"min" | "hours" | "days"
<Variable>::= <Character>
<Value>::= <Character>
<Character> = -->"StringIdentifier in UTF-8"
```

Figure 1. EBNF Grammar for the Decision Logic (DL)

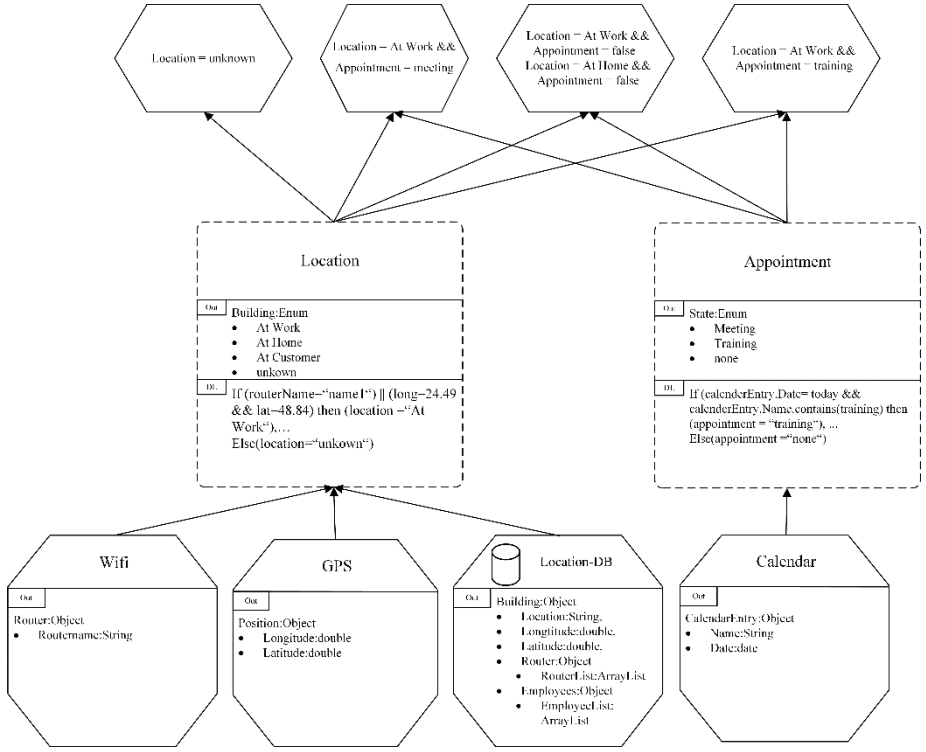


Figure 2. Sensor model for the contexts *location* and *appointment*

The next context parameter is *appointment* with its variables *meeting, training* and *none*. If the context is *location* is *at work* and context *appointment* is *training* the mobile application reminds the user to attend the sales training. Therefore, they are capable to automatically adapt the application to the context of the user. At the bottom line, the sensors are represented. For the context *location*, which is represented in the dotted rectangle above, the physical sensors *Wifi*, *GPS* and the virtual sensor *Location-DB* are needed. The GPS sensor returns an object, consisting of a long- and latitude double to compute the position of the salesperson. In the center of the figure, the context elements are represented. The *DL* areas show when an *appointment* is a *meeting* or *training*, respectively when the *location* is *at work, at home, at the customer* or *unknown*. For example, if the calendar entry is today and the entry description contains the name *training*, the salesperson should be part of the training. At the top of the figure are the context expressions related to the context elements. They are also the connection to the related context-aware business process. This example shows that it is possible to model the context evaluation for mobile context-aware business processes with SenSoMod. It enables to model the aggregation of information from the basic sensor information to the context expressions.

4 Evaluation

The aim of the design science research (DSR) is to create artifacts for a specific purpose that solve a practice oriented problem [29, 35]. One core activity in DSR is the evaluation of the created artifact to prove and justify its usefulness [36]. In the literature of DSR several methods to conduct an evaluation of a developed artifact exist, like mathematical proofs, case studies, surveys, or (lab) experiments, to name just a few [37, 38]. Since the experimental approach is suited for a systematic assessment of an information system artifact's qualities [39], it was chosen as the approach to evaluate the domain specific modeling language SenSoMod.

4.1 Evaluation Design

A classic experiment is conducted by having two groups divided into a so-called *test group* and a *control group*. The *test group* is "exposed to the stimulus (like the new or better design)" [39] whereas the *control group* is not. The reactions of the participants of both groups are compared to each other with the aim to distinguish reality (results of the experiment) from wishful thinking (expected improvement of the artifact). The participants of the experiment should be randomly assigned to either the test or the *control group*. For the data acquisition, one can use for instance a questionnaire, observation, or data transmission instruments (like an eye-tracker). From this data the researchers can derive whether an artifact proved its usefulness or not [39].

The described framework to conduct an experiment in the information system area was used to evaluate SenSoMod. The participants of the two groups had to fulfill the same task: In an office building, there are printers that will report to a smartphone application to change the cartridge when it is empty. If the user is nearby the mobile app

gives a notification. The modeling task of this use case was the information aggregation (printer sensors and smartphone sensors). In the *test group*, the participants had to model this task with SenSoMod, whereas the participants of the *control group* were free to use their preferred modeling language to succeed the task. The task had to fulfill certain criteria like representing a real-world example which is practical and complex enough to rule out trivial models, but at the same time, it had to be manageable for the participants. After the modeling, the *control group* saw a video which explained the language to them and showed how to model the task with SenSoMod. At the end, both groups received a questionnaire to judge the key features of SenSoMod. In addition, the participants in both groups were clocked during the task to observe how the new language can cope with known languages (like UML or BPMN). The aim of the evaluation was to target experienced developers and modelers to implement mobile applications, since this is the target group for the language. In total 15 practitioners took part in the experiment, 10 in the *test group* and 5 in the *control group*. The unbalanced groups are caused by randomly selected practitioners deciding not to attend the experiment after they were chosen.

4.2 Results

After conducting the experiment, the submitted questionnaires were carefully examined by two researchers independently. The results of each group were compared to each other to identify differences. All results from the participators were applicable in terms of consistency and completeness, and remain for further analysis.

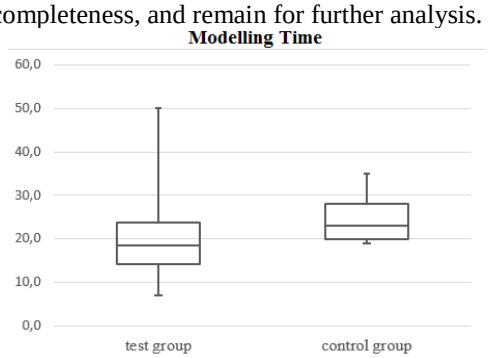


Figure 3. Box plot diagram of the modeling time

The upper limit for the modeling time in the *test group* is 50 minutes (cf. **Figure 3**). However, with Grubbs' [40] formula to determine outliers, this value is a weak outlier. So, if this value can be left out the upper limit would be 35 minutes. The upper limit of the *control group* is also 35 minutes. The average time in the *test group* was 18.1 minutes (with the outlier 21.3 min) with a standard deviation of 7.8 minutes (12.1 min), while in the *control group* it was 25 minutes with a standard deviation of 5.9 minutes. The conclusion of this data is that modeling with SenSoMod does not take longer than with other (preferred) modeling languages, although the *test group* had never seen the language before. It also seems that the *test group* is slightly faster. However, this trend should be investigated further with a bigger group of participants to confirm it. Both

groups had to answer a questionnaire at the end of the experiment. All questions regarding their opinion of SenSoMod were answered on a Likert-scale [41] with 5 answer options ranging between *applies* and *applies not*. The results for every question and group can be seen in **Table 2** and **Table 3**.

Table 2. Answers from the *test* group

	applies	applies rather	neither	applies rather not	applies not
Did you find the modeling language SenSoMod understandable? (All)	10%	70%	0%	20%	0%
Did you find the modeling language SenSoMod understandable? (Experienced modeler)	14%	71%	0%	14%	0%
Were you able to distinguish the individual elements of SenSoMod well from each other?	40%	20%	10%	30%	0%
Do you think that modeling only with SenSoMod can facilitate the implementation of mobile context-sensitive applications? (All)	10%	40%	30%	10%	10%
Do you think that modeling only with SenSoMod can facilitate the implementation of mobile context-sensitive applications? (Experienced developer)	20%	60%	20%	0%	0%
Do you think that a modeling tool for SenSoMod, which enables automatic generation of classes and methods, can facilitate the implementation of mobile context-sensitive applications?	50%	40%	0%	0%	10%
Do you think that SenSoMod makes it easier to collaborate with the business user of the app, by displaying the context aggregation graphically, thus making it easier to explain how the application adapts to the context?	40%	50%	0%	0%	10%

Table 3. Answers from the *control* group

	applies	applies rather	neither	applies rather not	applies not
Did you find the modeling language SenSoMod understandable? (All)	20%	60%	0%	20%	0%
Did you find the modeling language SenSoMod understandable? (Experienced modeler)	25%	50%	0%	25%	0%
Were you able to distinguish the individual elements of SenSoMod well from each other?	20%	60%	0%	20%	0%
Do you think that modeling only with SenSoMod can facilitate the implementation of mobile context-sensitive applications? (All)	40%	20%	20%	20%	0%
Do you think that modeling only with SenSoMod can facilitate the implementation of mobile context-sensitive applications? (Experienced developer)	50%	25%	0%	25%	0%
Do you think that a modeling tool for SenSoMod, which enables automatic generation of classes and methods, can facilitate the implementation of mobile context-sensitive applications?	20%	20%	40%	20%	0%
Do you think that SenSoMod makes it easier to collaborate with the business user of the app, by displaying the context aggregation graphically, thus making it easier to explain how the application adapts to the context?	60%	20%	20%	0%	0%

To get a better understanding how experienced modelers and developers think about SenSoMod, we distinguished between experts and inexperienced participants for two question. Overall the result is positive for SenSoMod. In both groups, SenSoMod was seen as understandable (slightly better within the group of experienced modelers) and can help to create mobile context-aware applications (also slightly better within the group of experienced developers). Only the differentiation between the elements was not clearly supported. The penultimate question targets the next step for the language, to create a tool able to generate classes from the model. Participants of both groups majoritarian stated that this could help to implement context-aware apps. Overall, this experiment showed that SenSoMod can be supportive for implementing apps which

change their behavior based on context information. However, the results of this experiment have to be confirmed in a second round with more participants.

5 Outlook and Further Research

The main contribution of this article is to show how the modeling language for the context evaluation can be used to ease the implementation of supportive mobile context-aware business processes. It provides the possibility for model engineers to plan such processes in a precise, detailed and comprehensive way. It also enables programmers to reuse the decision logic used in the sensor model for the source code of the application. Therefore, it can improve the interaction between modelers and programmers and accelerate the adaption of business processes. To further confirm the positive results of the evaluation a second evaluation should take place with more participants.

There are some tasks for further research in this area: Since mobile context sensitive business processes obviously need to measure context and are supported by an application on smart devices, an automated or semi-automated way to generate code from the business process would be helpful to increase the flow between modeling and implementation phase. The logical context expressions can be used to generate decisions in the application program.

References

1. Prümper, J., Lorenz, C., Hornung, S., Becker, M.: Abschlussbericht der Studie: „Mobiles Arbeiten". spring Messe Management GmbH, Frankfurt am Main (2016)
2. Morabito, V. (ed.): Trends and challenges in digital business innovation. Springer, Cham (2014)
3. Hammer, M., Champy, J.: Reengineering the corporation. A manifesto for business revolution. Harper Business, New York, NY (1993)
4. Becker, J., Kugeler, M., Rosemann, M. (eds.): Process management. A guide for the design of business processes. Springer, Berlin (2011)
5. Vom Brocke, J., Rosemann, M. (eds.): Handbook on Business Process Management 2. Strategic Alignment, Governance, People and Culture. Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg (2010)
6. Scheer, A.-W.: ARIS - Business Process Modeling. Springer, Berlin u.a. (2000)
7. Bichler, M., Frank, U., Avison, D., Malaurent, J., Fettke, P., Hovorka, D., Krämer, J., Schnurr, D., Müller, B., Suhl, L., et al.: Erratum to. Theories in Business and Information Systems Engineering. Bus Inf Syst Eng (2016)
8. Harmon, P., Wolf, C.: The State of Business Process Management 2016 (2016)
9. Falk, T., Leist, S.: Effects of mobile solutions for improving business processes. In: Avital, M., Leimeister, J.M., Schultze, U. (eds.) ECIS 2014 proceedings. 22th European Conference on Information Systems ; Tel Aviv, Israel, June 9-11, 2014. AIS Electronic Library (2014)
10. Dey, A.K.: Understanding and Using Context. Personal and Ubiquitous Computing 5, 4–7 (2001)

11. Heinrich, B., Lewerenz, L.: A Novel Concept for the Usage of Mobile Information Services. In: Linnhoff-Popien, C., Zaddach, M., Grahl, A. (eds.) Marktplätze im Umbruch. Digitale Strategien für Services im mobilen Internet, pp. 319–329. Springer Vieweg, Berlin (2015)
12. Dörndorfer, J., Seel, C.: The impact of mobile devices and applications on business process management. In: Barton, T., Herrmann, F., Meister, V., Müller, C., Seel, C. (eds.) Prozesse, Technologie, Anwendungen, Systeme und Management 2016. Angewandte Forschung in der Wirtschaftsinformatik, pp. 10–19 (2016)
13. Schilit, B., Adams, N., Want, R.: Context-Aware Computing Applications. In: First Workshop on Mobile Computing Systems and Applications (WMCSA), pp. 85–90 (1994)
14. Schmidt, A., Beigl, M., Gellersen, H.-W.: There is more to context than location. Computers & Graphics 23, 893–901 (1999)
15. Weiser, M.: The Computer for the 21st Century. Sci Am 265, 94–104 (1991)
16. Dey, A.K.: Providing Architectural Support for Building Context-aware Applications. Georgia Institute of Technology, Atlanta, GA, USA (2000)
17. Rosemann, M., Recker, J.C., Flender, C.: Contextualisation of business processes. International Journal of Business Process Integration and Management 3 (1), 47–60 (2008)
18. Rosemann, M., Recker, J., Flender, C.: Designing context-aware Business Processes. In: Siau, K., Chiang, R., Hardgrave, B.C. (eds.) Systems analysis and design. People, processes and projects, pp. 51–73. M.E. Sharpe, Armonk, NY u.a (2011)
19. Soffer, P.: On the Notion of Flexibility in Business Processes. In: Proceedings of the CAiSE'05 Workshops, pp. 35–42 (2005)
20. Rosemann, M., van der Aalst, W.M.P.: A configurable reference modelling language. Information Systems 32, 1–23 (2007)
21. Saidani, O., Nurcan, S.: Towards Context Aware Business Process Modelling. In: Workshop on Business Process Modelling, Development, and Support, p. 1. Norway (2007)
22. Heinrich, B., Schön, D.: Automated Planning of Context-aware Process Models. University of Münster, Münster, Germany (2015)
23. Conforti, R., La Rosa, M., Fortino, G., ter Hofstede, A.H.M., Recker, J., Adams, M.: Real-time risk monitoring in business processes. A sensor-based approach. Journal of Systems and Software 86, 2939–2965 (2013)
24. Dörndorfer, J., Seel, C.: A Meta Model Based Extension of BPMN 2.0 for Mobile Context Sensitive Business Processes and Applications. In: Leimeister, J.M., Brenner, W. (eds.) Proceedings der 13. Internationalen Tagung Wirtschaftsinformatik (WI), pp. 301–315. St. Gallen (2017)
25. Al-alshuhai, A., Siewe, F.: An Extension of Class Diagram to Model the Structure of Context-Aware Systems. In: The Sixth International Joint Conference on Advances in Engineering and Technology (AET) (2015)
26. Al-alshuhai, A., Siewe, F.: An Extension of UML Activity Diagram to Model the Behaviour of Context-Aware Systems. In: Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), pp. 431–437 (2015)
27. Kramer, D., Clark, T., Oussena, S.: MobDSL. A Domain Specific Language for multiple mobile platform deployment. In: 2010 IEEE International Conference on Networked Embedded Systems for Enterprise Applications, pp. 1–7. IEEE (2010)

28. Ranabahu, A.H., Maximilien, E.M., Sheth, A.P., Thirunarayan, K.: A domain specific language for enterprise grade cloud-mobile hybrid applications. In: Lopes, C.V. (ed.) Proceedings of the compilation of the co-located workshops on DSM'11, TMC'11, AGERE!'11, AOOPES'11, NEAT'11, & VMIL'11 - SPLASH '11 Workshops, p. 77. ACM Press, New York, New York, USA (2011)
29. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design Science in Information Systems Research. MIS Q 28, 75–105 (2004)
30. Moody, D.: The "Physics" of Notations. Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. IIEEE Trans. Software Eng. 35, 756–779 (2009)
31. Genon, N., Heymans, P., Amyot, D.: Analysing the Cognitive Effectiveness of the BPMN 2.0 Visual Notation. In: Malloy, B., Staab, S., van den Brand, M. (eds.) Software language engineering. Third international conference, SLE 2010, Eindhoven, The Netherlands, October 12 - 13, 2010 ; revised selected papers, 377-369. Springer, Berlin (2011)
32. Biederman, I.: Recognition-by-components. A theory of human image understanding. Psychological Review 94, 115–147 (1987)
33. Internet Engineering Task Force: The JavaScript Object Notation (JSON) Data Interchange Format (2014)
34. Backus, J.W., Wegstein, J.H., van Wijngaarden, A., Woodger, M., Bauer, F.L., Green, J., Katz, C., McCarthy, J., Perlis, A.J., Rutishauser, H., et al.: Report on the algorithmic language ALGOL 60. Commun. ACM 3, 299–314 (1960)
35. March, S.T., Smith, G.F.: Design and natural science research on information technology. Decision Support Systems 15, 251–266 (1995)
36. Peffers, K., Rothenberger, M., Tuunanen, T., Vaezi, R.: Design Science Research Evaluation. In: Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B. et al. (eds.) Design Science Research in Information Systems. Advances in Theory and Practice, 7286, pp. 398–410. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
37. Cleven, A., Gubler, P., Hüner, K.M.: Design alternatives for the evaluation of design science research artifacts. In: Vaishanvi, V., Purao, S. (eds.) Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology - DESRIST '09, p. 1. ACM Press, New York, New York, USA (2009)
38. Venable, J., Pries-Heje, J., Baskerville, R.: A Comprehensive Framework for Evaluation in Design Science Research. In: Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B. et al. (eds.) Design Science Research in Information Systems. Advances in Theory and Practice, 7286, pp. 423–438. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
39. Mettler, T., Eurich, M., Winter, R.: On the Use of Experiments in Design Science Research. A Proposition of an Evaluation Framework. Communications of the Association for Information Systems 34, 223–240 (2014)
40. Grubbs, F.E.: Procedures for Detecting Outlying Observations in Samples. Technometrics 11, 1–21 (1969)
41. Likert, R.: A Technique for the Measurement of Attitudes. New York, NY (1932)