Business Process Variability and Public Values

Boris Shishkov[1,3(✉)] and Jan Mendling[2]

[1] Institute of Mathematics and Informatics,
Bulgarian Academy of Sciences, Sofia, Bulgaria
[2] Institute for Information Business,
Vienna University of Economics and Business, Vienna, Austria
jan.mendling@wu.ac.at
[3] Institute IICREST, Sofia, Bulgaria
b.b.shishkov@iicrest.org

Abstract. A business process is a structure of inter-related activities that are executed in order to achieve a specific business objective. Organizations often maintain multiple variants of a given business process because of changing conditions, different regulations in different countries, or other contextual factors. We aim at specifying the relationship between a generic business process and its different variants, taking the perspective of public values, such as privacy, accountability, and transparency. The business process variants in turn may be a basis for software specifications – in this, business processes would be bridging between societal demands (possibly concerning public values) and the corresponding technical (software) functionalities. Our contribution is featuring a meta-model that describes business processes on a value-independent level; they can be extended towards value-specific business process variants that can be related in turn to software architectures. We reflect this in proposed value operationalization guidelines, using concepts from business process design as a basis; those guidelines assume coming firstly through technology-independent artefacts and secondly – through technology-specific artefacts, to arrive at software specifications that are adequate with regard to public-values-related demands.

Keywords: Business process variability · Public values · Software design

1 Introduction

A **business process** is a *structure of inter-related activities* that are executed in order to achieve a specific *business objective* [29]. Activities of business processes are increasingly supported by **ICT** (**Information and Communication Technology**), no matter if they are *intellectual* (e.g. business calculations) or *manual* (e.g. warehouse picking), *routine* (e.g. standard calculations) or *non-routine* (e.g. brain scan assessment) [7]. Against this background, there is an increasing awareness of the importance of **public values** ("*values*", for short), such as *privacy*, *accountability*, and *transparency*, especially as it concerns business processes [8]. The consideration of such *values* in the development of **software products** is for these reasons an important concern in recent debates on *ethical design* [3, 11]. If *business process*

B. Shishkov (Ed.): BMSD 2018, LNBIP 319, pp. 401–411, 2018.
https://doi.org/10.1007/978-3-319-94214-8_31

design and *software specification* must take *values* explicitly into account, it is an open question how to adequately **operationalize** values, "translating" them into *value-sensitive software artefacts*. A key problem in this regard is that *values are abstract, non-functional, and relevant to social sciences* while *software specifications are technical, functional, and relevant to computing paradigms*. So far, **Requirements Engineering** (**RE**) has been addressing the *domain-imposed* and *user-defined requirements* with regard to the software system-to-be, touching upon *functional* and *non-functional concerns*. Further, a non-functional requirement on externally observable properties of a system may lead to functional requirements on the internal structure of the system [1]. Therefore, considering values as non-functional requirements could seem "tempting" in this regard. Nevertheless, since non-functional requirements are usually *technical* (for example: *recoverability*, *response time*, and so on) and the requirements engineering experience is in translating non-functional requirements to functional requirements, *it is still a question how to "translate" values to functional requirements*. Furthermore, **Value-Sensitive Design** (**VSD**) is about *weaving values in the design* of (technical) systems. For this, the role of values is advocated among stakeholders, such that there is such an awareness since the very early stage of a (software) project [8]. Nevertheless, VSD is abstract whereas engineering is concrete and hence we argue that VSD does not have "solid" bridges to engineering.

In addressing the above problem, we are inspired by the observation that *business processes* are *de facto* "bridging" between *societal public demands* and the corresponding *technical* (*software*) *functionalities*. That is because business processes are essentially *human-driven* and at the same time it is through business processes that people and enterprises *utilize ICT* [29]. Thus, it is expected that *business processes could play a crucial role in closing the gap between values and software functionalities*, as much as they are considered as a means of implementing strategies [26]. Our contribution is featuring a **formal meta-model that describes business processes on a value-independent level, which can be extended towards value-specific business process variants**. The underlying idea builds on the observation that organizations often maintain multiple **variants** of a given business process because of *changing conditions*, *different regulations in different countries*, or *other contextual factors* [30]. Correspondingly, our meta-model specifies the *relationship between a generic business process (featuring the base features of the business process) and its different variants*, by the help of *values*; further, if information systems are to be developed, **we consider business processes as a key foundation with regard to the specification of software**. We provide corresponding guidelines concerning the **operationalization of values in the context of software specification,** using business process design concepts as a basis.

Further in the paper: In Sect. 2, we consider *essential concepts* discussed above: *value*, *requirements*, *business process variability*. Our proposed *meta-model* and *guidelines* are presented in Sect. 3. In Sect. 4, we relate our work to *relevant streams of research*: *feature modeling*, *aspect-oriented design*, and *configurable process models*, such that we position further our contribution, emphasizing on its useful features. We conclude the paper in Sect. 5.

2 Background

Referring to the notions considered in the Introduction, we will firstly elaborate on *values and their relation to requirements* (Subsect. 2.1) and secondly – *on business process variability* (Subsect. 2.2).

2.1 Values and Requirements

We argue that essentially, **`values`** are desires of the general public (or public institutions/organizations that claim to represent the general public), that are about properties considered societally valuable, such as *respecting the privacy of citizens* or *prohibiting polluting activities*. Even though values are to be broadly accepted (that is why they are **`public`**), they may concern individuals (for example: considering privacy) [32]. Hence, put broadly, values concern the *societal expectations with regard to the way services should be delivered*. Furthermore, we argue that "values" become actual "values" only if *resources are committed for this* (for example, a government finds privacy so important that time and money are invested to regulate and enforce privacy); otherwise things only remain at the level of "hollow" abstract desires (such as for example: "Make the World a better place") that are stated but are never effectively realized.

Since most current *technical systems* are essentially *goal-driven* [25], it is interesting to analyze values vs. goals conceptually, acknowledging that *enterprises can adopt values as part of their goals*. This is often done *under public pressure or through legislation*, as values may *conflict* with enterprise goals, such as *profitability* or *cost-saving* [14]. Values may also *differ from the user goals* because often values concern *third parties* and the user would not care about third parties as long as the user demands are fulfilled. Hence, values may be reflected in goals even though those would usually be *societal* (third-party) *goals* and not *enterprise goals* or *user goals*.

We propose a value categorization (Fig. 1) according to which values are desires relevant to particular *persons* (either *physical* or *legal* persons) or their *societal environment*. As such, values may either concern a particular individual or society altogether. Hence, we can distinguish between *individual values* (for example, privacy) and *societal values* (for example, sustainability). We also distinguish between *basic values* (for example, love), *moral values* (for example, justice), *physical values* (for example, nature), and *virtual values* (for example, intelligence). This categorization is inspired by [24, 28, 34].

Further, values may be different with different *stakeholders* and might differ among *countries* and *cultures*, and eventually change over *time*. Next to that, agreeing on a particular value would not necessarily mean agreeing on its *operationalization* [9]. There may be *different operationalizations and implementations of the same value*. Finally, different values might be in conflict between each other (referred to as *value tensions*), meaning that fulfilling one value and fulfilling another value would not be possible at the same time. Nevertheless, resolving value tensions might resort to ideas for handling goal conflicts [33] and is therefore outside the scope of this paper.

Since *weaving values in software functionalities* could only be materialized through considering the corresponding *requirements specification*, it is important to discuss the relation between the value concept and the requirement concept. As seen from the current discussion (see above, see the Introduction), *values* are *desires* or *goals*, not *requirements*. Values are abstract and not directly related to an enterprise or software system [8], as opposed to requirements [1]. Moreover, values are construct by and for society and not by and for the enterprise domain in which a specific system will be used. Those domains may overlap but are not the same. Values that are adopted as goals by an enterprise would thus impact the requirements on a system that the enterprise wants to introduce in order to realize its goals. Hence, *the impact of values cannot be limited to non-functional requirements*. It is therefore considered important to clearly distinguish *values* from *requirements* and acknowledge the limitations of requirements engineering with regard to the development of value-sensitive software systems.

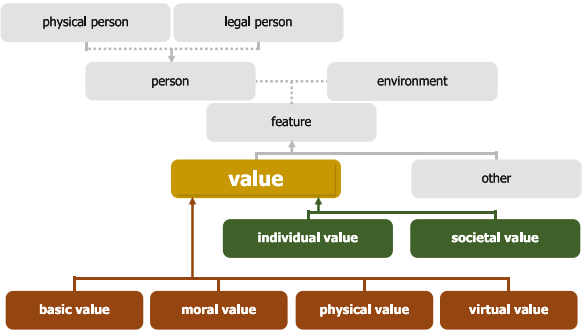

Fig. 1. Categorizing values

At the same time, it is important to align those concepts and position requirements accordingly. According to [2], *RE* is partially about achieving a *coherent description of the causal relationships between the phenomena in a particular domain*. This concerns the *domain-imposed requirements*. They are important because whatever we design, our designed artefact would be functioning in its environment or domain [29]. For this reason, the *regulations* and *rules* governing that domain should also have an impact on the designed artefact and its behavior [20]. Next to that, *RE* is also about *what the user wants* the designed artefact to do – this concerns the *user-defined requirements* [29]. Obviously, it is most important that the user-defined requirements are consistent with the domain-imposed requirements – otherwise, the designed artefact would be irrelevant with regard to its environment. Still, the user-defined requirements go beyond that and bring to the design the particular user demands. As for *values*, we argue that they are relevant in two ways. Firstly, they are relevant to the domain-imposed requirements because of the *consideration of societal issues that concern the domain*. Secondly, they are also relevant to the user-defined requirements because the *delivery of a value-sensitive service to the user* would differ from what the service delivery would have been with no consideration of values.

Therefore, it is not trivial to simply impose values on top of a traditional software development process that assumes analysis, requirements specification, design, implementation, and testing [15]. For this reason, we propose a *meta-model* and corresponding *design guidelines*.

2.2 Business Process Variability

The idea of *adjusting the behavior* according to pre-configured parameters has been included in various proposals for configurable business process models [27]. Configurable business process models are business process models that make various *alternative design choices* explicit by the help of configurable elements such as *activities* and *gateways*. These alternatives have to be selected *at design time* in order to arrive at a configured business process model. This configured business process model is a specific **`variant`** of the *generic business process model* and it can then be used for the *implementation of software systems*.

The general idea of configurable business process models fits well the overall ambition of *value-driven modeling*. A specific approach to configurable business process models is: a *questionnaire-driven configuration* [19], *aggregated business process models* [23] and *configurable multi-perspective process models* [17]. Those have in common that *the configuration can be tied to specific configuration parameters*. In this way, several configurable elements can be configured together. In our view, this concept is highly suitable to address the idea of value-driven modeling since *values can be used to configure entire parts of the business process model* in order to respond to corresponding demands.

3 Proposal

In this section, we propose a conceptualization and design guidelines of value-driven modeling.

3.1 Conceptualization

In order to *close the gap between abstract values and software specifications* in the context of enterprise systems and business processes, we make several **`assumptions`**:

- When considering an *organization* and a *software system that is supporting it*, we assume that *there are values that need to be reflected in the software design* (otherwise there would be no relevance to the topic of the current paper), and:
  - It is assumed that those values are *known*;
  - It is assumed that they *have to be weaved into the system design* and it is outside the paper's scope discussing why is it beneficial for society that this happens.
- We assume that all relevant values are identified *at design time*.
- We assume that for any value, there is a known *corresponding business process variant*; then, selecting it is expected to lead to the value fulfillment.

Based on those *assumptions* and the above *elaborations*, we introduce our main modeling concepts (meta-model) using the notations of UML - Class Diagram [31]. Figure 2 shows that there are *five key concepts* (**`society`**, **`value`**, **`business process variant`**, **`software architecture`**, and **`information system`**) represented as **classes** (*named boxes*), complemented by corresponding **associations** (*lines*).

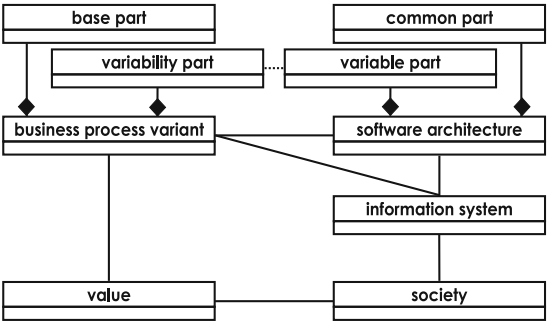

Fig. 2. Concepts

Values are *defined in the society* as represented by the association line between "value" and "society" and a *value* has its *corresponding business process variant* as defined by the association line between "value" and "business process variant" (this is according to the assumptions). In turn, a *business process variant* consists of one or more *base parts* and one or more *variability parts* as defined by the composition signs in the figure. Further, it is the *business process variant* that essentially concerns the *architecture* of the supportive software because the software would have to support partially or completely the business process variant, which is specified by the association line between "business process variant" and "software architecture". A *software architecture* in turn consists of one or more *common parts* and one or more *variable parts*, represented by the composition signs in the figure, and it is the *business process variability parts* that guide the design of corresponding *software architecture variable parts* (this is indicated by the dotted line in the figure). Finally, this results in an *information system* that has both *human* and *technical* aspects and is therefore related not only to its *underlying software architecture* but also the *supported business process variant*. This is indicated by the corresponding association lines. It is therefore the *information system* that would eventually deliver services to its customers that are value-sensitive, thus relevant in terms of values to the society.

In this way, **`values are reflected in business process variants`** that in turn **`shape the software architecture`** that is underlying with regard to the **`information system`** that guarantees those values for the society.

3.2 Guidelines

As suggested already, our proposed design guidelines are about the **value-driven modeling of business process variants** and a further **mapping towards software specifications**. This is represented in Fig. 3 using the notations of UML - Activity Diagram [31]. As the figure suggests, we have *two parallel processes*, a **value-independent** one and a **value-specific** one. The former is about the *base part modeling* (reflecting invariant business process behavior) and its mapping towards corresponding design that is featuring *core parts of the software architecture*; the latter is about the project-driven *consideration of a particular value* that leads to the modeling of corresponding *business process variability issues* that are in turn mapped towards corresponding design that is featuring *variable parts of the software architecture*. Further, all *business-process-modeling-related tasks* are **technology-independent** (see the brick-backgrounded area in the figure) while all *software-design-related tasks* are **technology-specific** (see the dotted area in the figure). Finally, the technology-independent activities are to be mutually *in synch*, just as the technology-specific activities (this is indicated by the dashed lines in the figure).

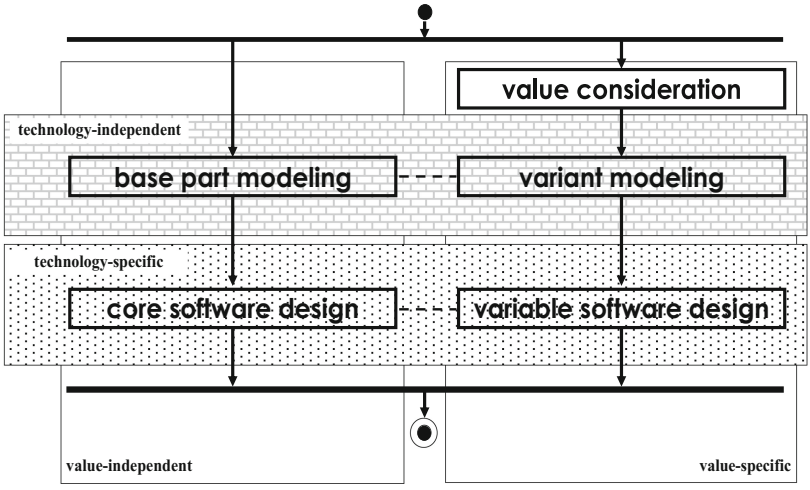

Fig. 3. Design guidelines

That is how we vision the **bridging role of business process variants** with regard to *values that need to be operationalized* and the corresponding *software specifications*.

Nevertheless, those guidelines need *further* (technical) *elaboration* and a follow up *validation*. This is left beyond the scope of the current paper and is planned as future research. Still, we have identified and studied *relevant research streams* (see the following section) that essentially ground our proposed guidelines (and the meta-model), emphasizing on their useful features.

4 Applicability – Related Work

The work presented in this paper is related to the following streams of research: **feature modeling**, **aspect-oriented design**, **configurable process models**.

Feature Modeling (FM) is concerned with the specification of *commonalities and variations of a software product*, e.g. to support the development of *software product lines* [6]. In *FM*, **variants are designed top-down**. Various methods exist for facilitating FM with *Feature-Oriented Domain Analysis* (*FODA*) being the most prominent one [12]. **Our work not only shares the emphasis on variations with FM but it also offers a novel perspective, by anchoring the choice of different variants on the abstract level of the business model**.

Aspect Orientation (AO) approaches variation from the perspective of *integrating additional functionality where needed* [13]. This is considered useful for adding **crosscutting concerns** to a software system such as to address *security requirements*, *response time*, *recoverability issues*, and so on. Specific operations can be used to *weave such functionalities into the original software core*. *AO* has been further integrated with *process modeling languages* like *BPMN* [4] in order to address *crosscutting business concerns* such as *compliance*, for example. **In our work, we envision the weaving (together) of separate building blocks, which partially builds on ideas of AO**.

In the research area of Configurable Process Models, several *languages* have been proposed to support the *specification of variability* [16], with *C-EPCs* [22], *Pesoa for BPMN* [21], and *Provop* [10], superimposed variants for *UML Activity Diagrams* [5] being among the most prominent ones. Those languages essentially share the idea of *full specification of variation at design time*. **Our work builds upon the idea that variants can be specified and selected on an abstract level**. This idea has been *instantiated* with *C-EPC* by the help of a *questionnaire with closed questions*, such that *answer options* can be translated to corresponding *variants* [18].

5 Conclusion

Considering software specifications that are based on business process models, we have addressed in this paper business processes in their role of bridging between societal public demands and the corresponding technical (software) functionalities, touching upon public values (desires of the general public). In particular, we have considered a value-driven specification of business process variants as a way of closing the gap between abstract public values and required corresponding value (software) operationalizations. This assumes not only identifying the business process variants but also reflecting them in turn in corresponding software specifications – this all prepared at design time. Hence, when considering a generic (value-independent) business process and a value-related demand, a business process variant may be specified (it is already value-specific but technology-independent); in turn, the business process variant could be reflected in a technology-specific software specification. It is expected that such a

design approach would increase effectiveness and efficiency when it is about the development of value-sensitive software systems. We have proposed a meta-model and design guidelines accordingly, and we have related our work to relevant streams of research. For this reason, the current paper is considered to have both analytical and propositional value. Nevertheless, the lack of sufficient elaboration and also the lack of validation frame our work as research in progress. We plan as future research to elaborate further on our proposal and validate its applicability by means of case studies.

Acknowledgement. This work is supported by the TU Delft - *Delft Pilot* project and we would like to thank *Jeroen van den Hoven* for his support and guidance.

References

1. Akkermans, H., Gordijn, J.: What is this science called requirements engineering? In: Proceedings of RE 2006 - 14th IEEE International Requirements Engineering Conference, Minneapolis/St. Paul, USA, pp. 273–278 (2006)
2. Anish, P.R., Daneva, M., Cleland-Huang, J., Wieringa, R.J., Ghaisas, S.: What you ask is what you get: understanding architecturally significant functional requirements. In: Proceedings of RE 2015 - 23rd International Requirements Engineering Conference, Ottawa, ON, pp. 86–95 (2015)
3. Baldini, G., Botterman, M., Neisse, R.: Ethical design in the internet of things. In: Sci Eng Ethics (2016)
4. Charfi, A., Müller, H., Mezini, M.: Aspect-oriented business process modeling with AO4BPMN. In: Kühne, T., Selic, B., Gervais, M.-P., Terrier, F. (eds.) ECMFA 2010. LNCS, vol. 6138, pp. 48–61. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13595-8_6
5. Czarnecki, K., Antkiewicz, M.: Mapping features to models: a template approach based on superimposed variants. In: Glück, R., Lowry, M. (eds.) GPCE 2005. LNCS, vol. 3676, pp. 422–437. Springer, Heidelberg (2005). https://doi.org/10.1007/11561347_28
6. Czarnecki, K., Kim, C.H.P.: Cardinality-based feature modeling and constraints: a progress report. In: Proceedings of International Workshop on Software Factories, San Diego, CA, pp. 16–20. ACM (2005)
7. Frey, C.B., Osborne, M.A.: The future of employment: how susceptible are jobs to computerisation? In: Technological Forecasting and Social Change, vol. 114 (2017)
8. Friedman, B., Hendry, D.G., Borning, A.: A survey of value sensitive design methods. In: A Survey of Value Sensitive Design Methods, 1, Now Foundations and Trends, 76 p. (2017)
9. Grenholm, C.-H., Kamergrauzis, N. (eds.): Sustainable Development and Global Ethics. Acta Universitatis Upsaliensis, Uppsala (2007)
10. Hallerbach, A., Bauer, T., Reichert, M.: Capturing variability in business process models: the Provop approach. J. Softw. Maint. Evol. **22**, 6–7, 519–546 (2010)
11. Heersmink, R., Hovenvan den, J., Eckvan, N.J., Bergvan den, J.: Bibliometric mapping of computer and information ethics. Ethics Inf. Technol. **13**, 241–249 (2012)
12. Kang, K.C., Cohen, S.G., Hess, J.A., Novak, W.E., Peterson, A.S.: Feature-Oriented Domain Analysis (FODA) Feasibility Study (No. CMU/SEI-90-TR-21). Carnegie-Mellon Univ. Pittsburgh (1990)

13. Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Loingtier, J.-M., Irwin, J.: Aspect-oriented programming. In: Akşit, M., Matsuoka, S. (eds.) ECOOP 1997. LNCS, vol. 1241, pp. 220–242. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0053381
14. Kinsella, W.J.: From big science to postmodern science: technology-intensive research in an era of competing public values. In: Proceedings of International Symposium on Technology and Society Technical Expertise and Public Decisions, Princeton, NJ, pp. 15–24. IEEE (1996)
15. Kruchten, P.: The Rational Unified Process, an Introduction. Addison-Wesley, Boston (2003)
16. La Rosa, M., Van Der Aalst, W.M., Dumas, M., Milani, F.P.: Business process variability modeling: a survey. ACM Comp. Surv. (CSUR) **50**(1), 2 (2017)
17. La Rosa, M., Dumas, M., Ter Hofstede, A.H.M., Mendling, J.: Configurable multi-perspective business process models. Inf. Syst. **36**(2), 313–340 (2011)
18. La Rosa, M., Van der Aalst, W.M., Dumas, M., Ter Hofstede, A.H.M.: Questionnaire-based variability modeling for system configuration. Softw. Syst. Model. **8**(2), 251–274 (2009)
19. La Rosa, M., Lux, J., Seidel, S., Dumas, M., ter Hofstede, A.H.M.: Questionnaire-driven configuration of reference process models. In: Krogstie, J., Opdahl, A., Sindre, G. (eds.) CAiSE 2007. LNCS, vol. 4495, pp. 424–438. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72988-4_30
20. Liu, K.: Semiotics in Information Systems Engineering. Cambridge University Press, Cambridge (2000)
21. Puhlmann, F., Schnieders, A., Weiland, J., Weske, M.: Variability mechanisms for process models. PESOA-Report TR, 17, pp. 10–61 (2005)
22. Recker, J., Rosemann, M., van der Aalst, W., Mendling, J.: On the syntax of reference model configuration – transforming the C-EPC into lawful EPC models. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 497–511. Springer, Heidelberg (2006). https://doi.org/10.1007/11678564_46
23. Reijers, H.A., Mans, R.S., Van der Toorn, R.A.: Improved model management with aggregated business process models. Data Knowl. Eng. **68**(2), 221–243 (2009)
24. Reynaers, A.-M.: It takes two to tangle: public-private partnerships and their impact on public values. Ph.D. thesis, Universidad Autónoma de Madrid (2014)
25. Rong, G., Liu, X., Gu, S., Shao, D.: A goal-driven framework in support of knowledge management. In: Proceedings of ASPEC 2017 – 24th Asia-Pacific Software Engineering Conference, Nanjing, pp. 289–297 (2017)
26. Rosemann, M., vom Brocke, J.: The six core elements of business process management. In: vom Brocke, J., Rosemann, M. (eds.) Handbook on Business Process Management 1: International Handbooks on Information Systems. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-642-00416-2_5
27. Rosemann, M., Van der Aalst, W.M.P.: A configurable reference modelling language. Inf. Syst. **32**(1), 1–23 (2007)
28. Schwartz, S.H.: An overview of the schwartz theory of basic values. Online Read. Psychol. Cult. **2**(1), 11 (2012)
29. Shishkov, B.: Enterprise Information Systems, a Modeling Approach. IICREST, Sofia (2017)
30. Sinnhofer, A.D., Pühringer, P., Oppermann, F.J., Potzmader, K., Orthacker, C., Steger, C., Kreiner, C.: Combining business process variability and software variability using traceable links. In: Shishkov, B. (ed.) BMSD 2017. LNBIP, vol. 309, pp. 67–86. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78428-1_4
31. UML: The website of the Unified Modeling Language (2018). http://www.uml.org

32. Van den Hoven, J.: Value sensitive design and responsible innovation. In: Owen, R., Bessant, J., Heintz, M. (eds.) Responsible Innovation: Managing the Responsible Emergence of Science and Innovation in Society. Wiley, Hoboken (2013)
33. Van Lamsweerde, A., Darimont, R., Letier, E.: Managing conflicts in goal-driven requirements engineering. IEEE Trans. Softw. Eng. **24**(11), 908–926 (1998)
34. Zhang, J.: A brief study of the hierarchy value thought of the pre-qin confucianism. In: Li, D. (ed.) Values of Our Times. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38259-8_17