systems

MDPI

Article

System-of-Systems Design Thinking on Behavior

Christian Stary

Department of Business Information Systems, Communications Engineering,
Johannes Kepler University of Linz, 4040 Linz, Austria; Christian.Stary@jku.at;
Tel.: +43-732-2468-4320

Academic Editors: Gianfranco Minati, Eliano Pessa and Ignazio Licata
Received: 31 October 2016; Accepted: 6 January 2017; Published: 13 January 2017

Abstract: Due to the increasing digitalization of all societal systems, informed design of services and systems becomes pertinent for various stakeholders. This paper discusses the design of digital systems in a user-centered way with the help of subject-oriented design. The approach follows a communication-driven and network-centric perspective on a System-of-Systems, whereby system specifications encapsulate behavior and exchange messages, including relevant data, such as business objects. Systems can represent activities of human actors, as well as artefacts. Stakeholders can be actively involved in their roles in the design of a System-of-Systems. In the course of design, they identify and refine role-specific behavior, based on communication to other actors or systems. A System-of-Systems specification evolves as a network of cooperating behavior entities. It develops according to communication needs and system-specific capabilities, on the level of synchronized execution agents, or as an overlay mechanism on existing applications or sub networks. Since certain behavior sequences, such as decision-making procedures, are re-occurring in organizations or eco-systems, the design of complex systems can be facilitated by behavior patterns stemming from existing modeling experiences.

Keywords: System of Systems; design; subject-orientation; communication structures; collaboration network; interaction; diagrammatic specification; design patterns; choreography

1. Introduction

Digitalization means the continuous penetration of IT applications into domains and areas of economic and lifestyle concern. A typical example are calendars. Once becoming digital they have been connected to other systems, like room and meeting management systems, even though there exist diverse personal ecologies of calendar artifacts. As Dittmar et al. [1] could show "the changing demands in daily life, the availability of new tools, and the participants' knowledge about the costs and benefits of their calendar work and about the consequences of potential failures influence their tendency to explore and possibly integrate new calendar artifacts and appear implicated in the deliberate non-use of new technology". Such findings indicate that design of these artefacts need to be reconsidered for the sake of their applicability and social acceptance once being embodied in digital infrastructures.

One opportunity of digital artifacts to put potential users in control of design is their capability to be increasingly editable, interactive, reprogrammable, and distributable [2]. This capability goes beyond adding or visually arranging apps on a tablet or smartphone interfaces. Rather, it refers to creating interactive digital environments involving various stakeholders, such as developers, consumers, facilitators, brokers, etc. (cf. [3]). When looking for methodological support of stakeholder involvement in system design, first inputs in terms of theories of design pop up (cf. [4]), while structured guidance seems still to be lacking [5]. Hence, more attention needs to be drawn in investigating, creating the design of digital systems in a stakeholder-sensitive way. It needs to go beyond user involvement

as commonly pursued in software development, as already existing services or applications need to be configured in a context-sensitive and adaptive way. In addition, different stakeholders may be involved, as they may need to be represented by interaction or system features, e.g., when specifying individualized alarm systems in healthcare (in the context of this paper, stakeholders denote persons having interest or share in an active community, e.g., a societal sector or organization. They are involved in, or affected by, some course of action of this community).

Co-creation of services and products is not only of relevance in consumer settings, but also in industrial settings (cf. [6–8]). For instance, Tang et al. [7] address the co-creation of digital services and applications in the Web 2.0 digital ecosystem where companies can co-create business with their customers. However, the focus here is in combining product and technological capabilities, rather than the process and method of co-creation. In a distributed product and service setting, the process of design needs to be supported in a constructive and methodologically grounded way, as digital innovations in vehicle maintenance reveal, at least following a layered architecture of digital technology [9]. Feature or service layers provide the opportunity to various stakeholders to create, manipulate, and store different systems, either as apps, functions, or services. For stakeholder-centered development execution capabilities must exist independently, but still intertwined. The latter support implementing designed artefacts and putting systems into operation [10].

According to these requirements we follow an 'assemblage of systems' approach in System-of-Systems (SoS) design. This is one of the timely identified perspectives on SoS [11]. Thereby, we put the stakeholders in control of the development process, aiming to integrate (existing) system behaviors to achieve capabilities that cannot be achieved by the constituent systems. The resulting behavior forms a collaborative network system. Methodologically, we follow Maier [12], as the definition of SoS as "a collaborative assemblage captures a grouping distinctively different in terms of developing best practices" (p. 3149). We leverage socio-technical equilibria according to the stakeholder perspective without requesting upper layers in the hierarchy, as originally being superimposed for architecting SoS (cf. [13]). However, in case a network node represents a complex system, e.g., providing access to a module of an enterprise resource planning system, it can be considered as a hierarchical overlay to existing system functions. In order to allow the execution of system models, we follow the idea of providing communicating structures as inherent part of SoS specifications (ibid.). The proposed subject-oriented SoS architecture encapsulates communication tasks with functional ones as elementary specification structure. The triggered send- and receive-patterns involve the exchange of messages and establish a choreographic flow of control in the collaborative assemblage.

The next section of this paper provides a review on the design of System-of-Systems (SoS), in order to detail the selection of SoS type according to the objective of the work. Then, a framework, corresponding methodological steps for SoS, and diagrammatic notational elements, stemming from subject-oriented business process management, are introduced. We detail behavior modeling capabilities designed for SoS stakeholders. We use those diagrammatic modeling capabilities to introduce subject-oriented SoS thinking for designing and re-designing SoS from a stakeholder perspective. Hence, both the diagrammatical manifestation of SoS design, and the automated execution of SoS model representations can be considered a step towards interactive and dynamic SoS development. The final section concludes the paper referring to these achievements.

2. System-of-Systems (SoS) Design

This section introduces System-of-Systems (SoS) as a design entity and reviews methodological inputs capturing the process of SoS design. The first subsection deals with structural foundations, the second subsection with behavior issues, including the emergence of SoS behavior.

2.1. System-of-Systems

System-of-Systems (SoS) has been conceptualized for engineering addressing the construction and development of complex artefacts [14]. Due to the variety of application domains, a variety of definitions

and explanations exists (cf. [15]). However, they consider a system as "a group of interacting elements (or subsystems) having an internal structure which links them into a unified whole. The boundary of a system is to be defined, as well as the nature of the internal structure linking its elements (physical, logical, etc.). Its essential properties are autonomy, coherence, permanence, and organization" (ibid., p. 1). Complex systems are constituted "by many components interacting in a network structure", with most often physically and functionally heterogeneous components, and organized in a hierarchy of subsystems that contributes to the system function [13]. As Jaradat et al. [11] have shown, several structures and categorization schemes have been used in the history of complex systems interpreted as system-of systems, ranging from closed coupling (systems within systems) to loosely coupling (assemblage of systems). In architectural terms, these properties correspond to embodied systems cooperating in an interoperable way (cf. [16]), allowing for autonomous behavior of systems or components while being part of a network collaborating with other systems and, thus, contributing to the objective of the network [12].

Referring to structural and dynamic complexity, 'structural complexity derives from (i) heterogeneity of components across different technological domains due to increased integration among systems; and (ii) scale and dimensionality of connectivity through a large number of components (nodes) highly interconnected by dependences and interdependences. Dynamic complexity manifests through the emergence of (unexpected) system behavior in response to changes in the environmental and operational conditions of its components' ([15], p. 1). The review of the Technical Committee of the IEEE-Reliability Society concludes with considering a System-of-Systems (SoS) as a system that involves several systems "that are operated independently but have to share the same space and somehow cooperate" (ibid., p. 2). As such, they have several properties in common: operational and managerial independence, geographical distribution, emergent behavior, evolutionary development, and heterogeneity of constituent systems (ibid.). As Jaradat et al. [11] pointed out (p. 206), these properties affect setting the boundaries of SoS and the internal behavior of SoS and, thus, influences methodological SoS developments. More concrete, according to Jaradat et al. ([11], p. 206) SoS are distinct with respect to:

(1) *autonomy*, where constituent systems within the SoS can operate and function independently and the capabilities of the SoS depends on this autonomy;
(2) *belonging* (integration), which implies that the constituent systems and their parts have the option to integrate to enable SoS capabilities;
(3) *connectivity* between components and their environment;
(4) *diversity* (different perspectives and functions); and
(5) *emergence* (foreseen or unexpected).

A typical example are apps being available on a smartphone. They can be considered as systems. When adjusting them along a workflow, e.g., to raise alerts and guide a patient to the doctor, in case certain thresholds with respect to medical conditions are reached for a user, several systems, such as a blood pressure app, calendar app, and navigation app, need to be coordinated and aligned for personal healthcare. In this case, the smartphone serves as a SoS carrier, eventually supporting patient-oriented redesign of the workflow and, thus, the SoS structure. The smartphone can still be used as a device to talk to other people while serving as a communication infrastructure of medically relevant systems. The latter identifies the smartphone a SoS component.

2.2. SoS Design as an Informed Process

As SoS thinking is grounded in recognizing the network-centric and knowledge-based nature of systems [17], a realist perspective on developing them seems to be appropriate (cf. [18]). Its focus is on adaptation and flexibility and, thus, on "local context and expressing findings as broad principles of action and contingent approaches" (ibid., p. 424f.). Hence, any abstraction to describe and specify needs to be adequate to the situation of use for its stakeholders (cf. [19] for process representations). A system's situatedness is awareness about its world, such as the organization, society, or other

contingent systems, and its capability to induce changes in it (cf. [20]). 'The essence of situation awareness lies in the monitoring of various entities and the relations that occur among them. Since the properties of relations, unlike the properties of objects, are not directly measurable, one needs to have some background knowledge (such as ontologies and rules) to specify how to derive the existence and meaning of particular relations' [21].

Consequently, SoS development should lead to architectures allowing dynamic changes (cf. [22]). Situatedness of behavior is a key issue in engineering support of communities (cf. [23]). Prescriptions need to be adapted to the situation at hand, allowing for systems dynamics in the course of development. Most important, we need to recognize that stakeholders and support systems, in particular when considered from a system perspective, are an integral part of situations, as termed by cognitive scientists: actors are considered as embodied and interactively situated in worlds [24]. When analyzing the meanings attached to these terms a set of conditions for situatedness and embodiment can be derived, based on the conclusive assumption that external representational schemas are required for adaptation. While virtual actors in virtual worlds are neither considered situated nor embodied, awareness of evolving goals, various modalities for interaction and task accomplishment procedures could lead to a rich repertoire of interactions. Embedded actors could develop individual points of view, relative to their starting position workspaces, and have a capacity to develop a dedicated interaction space. None of these capabilities are possible without representational capacities, such as diagrammatic or formal notations.

Hereby, system thinking plays a crucial role, as it is a way of looking at situations as an ecosystem. A situation is analyzed in terms of how the different parts influence and relate to each other rather than decomposing it into parts that are studied in isolation [11,25]. System thinking focuses on actors in a mutually dependent setting. Their concern is how they work together and respond to each other even following complex paths of behavior. Design is focused on development work of systems while keeping the whole in mind. According to Frank [26] stakeholders creating systems through system thinking need a variety of cognitive competencies. They need to "understand the whole system beyond its elements, sub-systems, assemblies and components, and recognize how each element/sub-system/assembly/component functions as part of the entire system. They are multifaceted, able to consider issues from a wide range of perspectives and points of view and possess a generalist's perspective" (p. 276). They also need to "understand the interconnections and the mutual influences and interrelations among system elements. Systems thinking involves thinking about the system's interactions, interrelationships, and interdependencies of a technical, social, socio-technical or multi-level nature" (ibid.).

In doing so, developers lay ground for emergent properties of systems, effecting perspectives beyond engineering an isolated system, e.g., a navigation app. A systemic representation, such as a SoS specification, enables one not to get stuck on details, and tolerate ambiguity and uncertainty. From a methodological perspective, the "good (the right) questions" ([26], p. 277) need to be asked, and given information needs to be questioned constantly. Moreover, these questions enable curiosity and open-mindedness, in order to consider a system beyond the limited area of a certain expertise. Without a sense of vision, novel system behavior might not emerge, neither when optimizing, nor re-engineering existing systems.

3. Subject-Oriented Design of SoS

Subject-oriented SoS design seeks to assist system development by providing a methodology that presents behavior-relevant information in a manner analogous to natural language features, namely, subject, object, and predicate constructs from the stakeholder's perspective [27]. These characteristics enable stakeholders to be engaged more effectively via a simple and intuitive behavior representation and via human-centered elicitation.

As informed design has the intention of reducing the time spent for, and complexity of, modeling, development support should also include the execution of models. Having an easy to learn,

and deployable, behavior modeling tool intertwined with execution also enables other stakeholders to have a platform for probing with existing reference patterns or models when specifying their mental representation. They could directly specify their behavior, e.g., in terms of a flow of activities, to achieve a certain objective. In case behavior knowledge is not available in explicit representations so far, probing supports stakeholders expressing themselves successively in terms of executable actions or communication acts (cf. [28]). In contrast to explicit elicitation techniques, such as interviews, stakeholders need not have to rely on information provided by analysts. In settings involving external people, such as for interviewing, it cannot be assumed that analysts are familiar with the domain at hand [29]. Moreover, stakeholders—in particular, experts—forget tasks to mention they assume to be widely known, or have difficulties explaining what they do without actually doing it [30], as knowledge is inseparable from doing (cf. [31]).

Putting situated cognition theory in the context of system modeling, generated models in a natural and intuitive way should potentially have greater accuracy than what could traditionally be achieved with common acquisition and analysis techniques (cf. [32]). Reducing the requirement of involving external people enables a wider scope of application, as more stakeholders could participate in organizational change and development. Subject-oriented design is focusing on parallel processes; thus, stakeholders can detail their behavior specification individually and concurrently, after agreeing on respective communication interfaces.

An underlying concept of behavior-based SoS design is agency: according to Himma [33] the "idea of agency is conceptually associated with the idea of being capable of doing something that counts as an act or action. As a conceptual matter, X is an agent if and only if X is capable of performing action; breathing is something we do, but it does not count as an action. Typing these words is an action, and it is in virtue of my ability to do this kind of thing that, as a conceptual matter, I am an agent. ... Agents are not merely capable of performing acts; they inevitably perform them (in the relevant sense). ... The very concept of agency presupposes that agents are conscious." (p. 19)

Reflecting this understanding reveals the manner of involvement in a situation when humans are acting or interacting. It underpins the requirement to devote design effort to human-centered behavior. Moreover, it enables paradigmatic shifts towards communication and interaction, in addition to functional task accomplishment or functional role fulfilment. While traditional approaches to modeling mostly rely primarily on an exclusive functional perspective on task accomplishment, subject-oriented SoS design focuses initially on communication links between active components and, thus, interactions between systems.

3.1. Specification Constituents and Models

In subject-oriented SoS design, systems are viewed as emerging from both the interaction between systems (termed subjects) and their specific behaviors encapsulated within the individual subjects. Like in reality, subjects (systems) operate in parallel and can exchange messages asynchronously or synchronously. It is a view of SoS with operating systems as autonomous, concurrent behaviors of distributed entities. A system (subject) is a behavioral role assumed by some "actor", i.e., an entity that is capable of performing actions. The entity can be a human, a piece of software, a machine (e.g., a robot), a device (e.g., a sensor), or a combination of these. (SoS) Subjects can execute local actions that do not involve interacting with other subjects (e.g., calculating a price and storing a postal address), and communicative actions that are concerned with exchanging messages between SoS subjects, i.e., sending and receiving messages.

SoS subjects are one of five core symbols used in specifying designs. Based on these symbols, two types of diagrams can be produced to conjointly represent a system: subject interaction diagrams (SIDs) and subject behavior diagrams (SBDs). SIDs provide a global view of a SoS, comprising the subjects involved and the messages they exchange. The SID of a simple ordering process is shown in Figure 1. Subject behavior diagrams (SBDs) provide a local view of the process from the perspective of individual subjects. They include sequences of states representing local actions and communicative

actions, including sending messages and receiving messages. State transitions are represented as arrows, with labels indicating the outcome of the preceding state (see Figures 2 and 3).

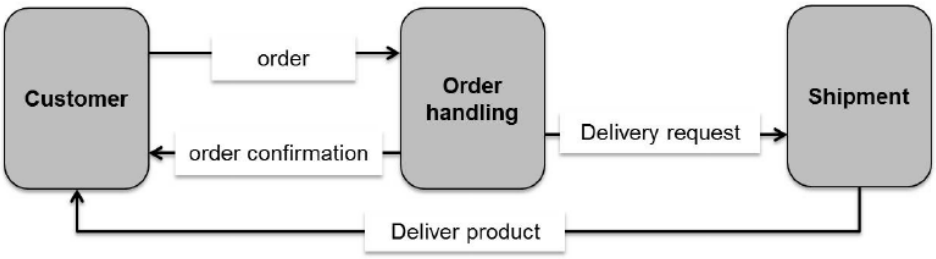

Figure 1. Order handling—subject interaction diagram.

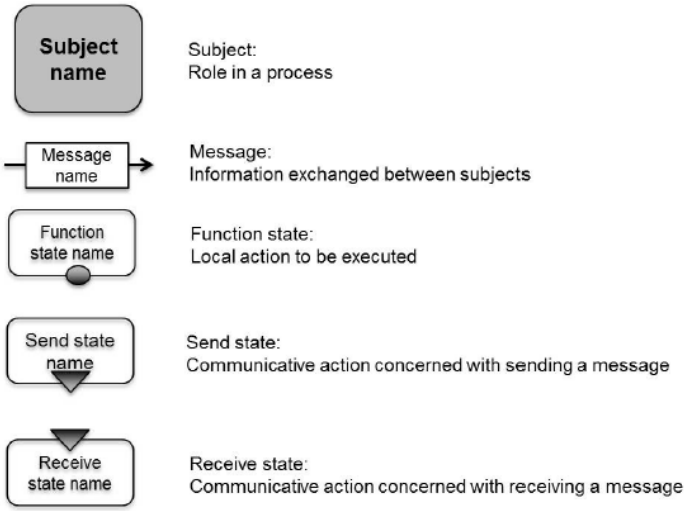

Figure 2. Diagrammatic elements in subject-oriented SoS design.

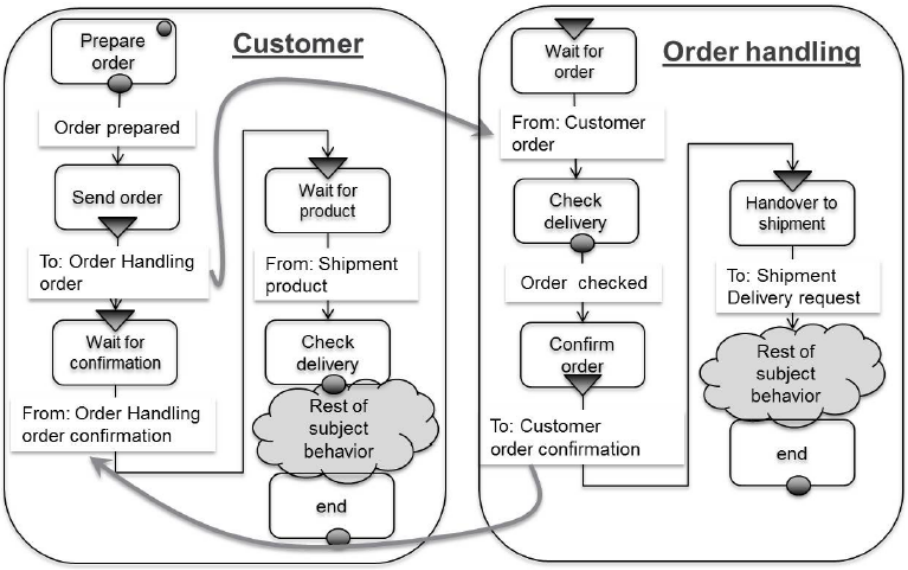

Figure 3. Order handling—Subject behavior diagrams "customer" and "order handling".

Given these capabilities SoS designs are characterized by:

- A simple communication protocol (using SIDs) and, thus,

- standardized behavior structures (enabled by SBDs), and
- scaling in terms of complexity and scope.

The approach allows meeting ad hoc, non-deterministic, and domain-specific requirements. The ultimate stage of scalability could be reached through dynamic and situation-sensitive formation of systems and their architecture beyond domains, referring to adaptability. As validated behavior specifications can be executed without further transformation (see Figure 4), stakeholders guide the implementation of their SoS specifications.

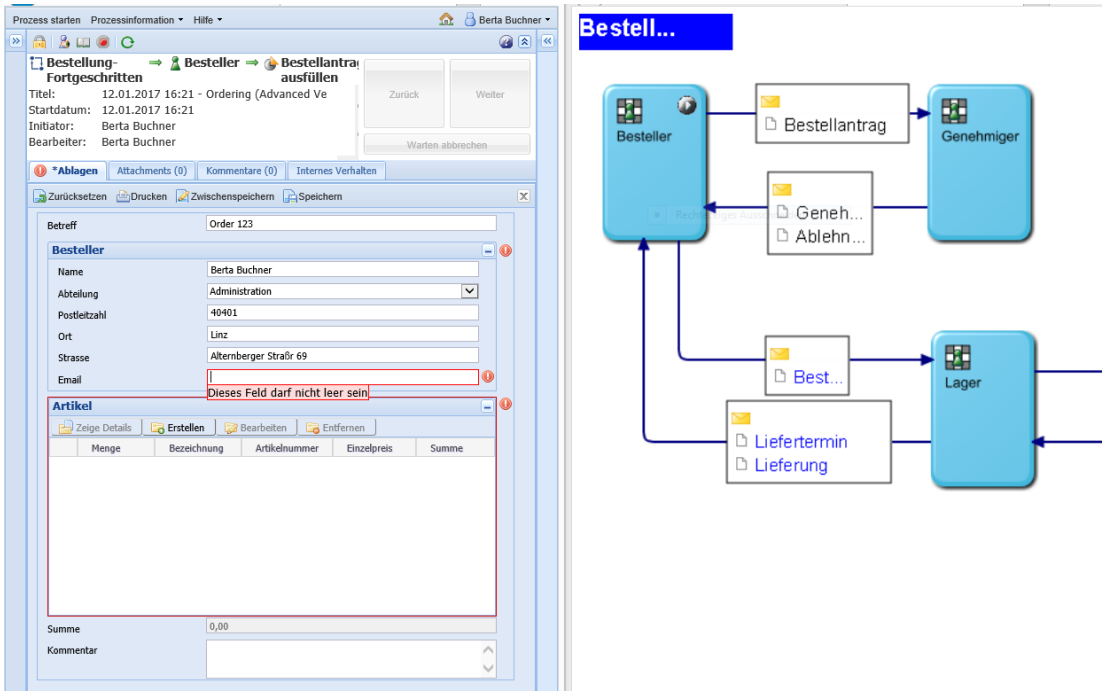

Figure 4. Execution of subject-oriented SoS behavior sequences.

3.2. Supporting Adaptive SoS Design

In design thinking workshops stakeholders work on subject-oriented models representing SoS from a cognitive, social, and organizational/domain perspective. The workshops include ad hoc models, presentations of pre-fabricated models, or user stories by stakeholders on their (work) situation, their task implementations, and their mental models of a domain. In the course of the workshops, both types of subject-oriented models, subject interaction diagrams (SIDs), and subject behavior diagrams (SBDs), are developed, in order to achieve an interactional understanding and corresponding SoS of the concerned organizational setting. Facilitators care about encapsulating system behavior and the required SoS communication structures. In addition, they also support probing SoS models through prototypical implementations. Since a group of stakeholders is usually affected by subject-oriented SoS representations—each subject can be represented by a person or system—the models need to be discussed and aligned according to the interaction patterns. In this way, the SoS models get validated. Starting point for all activities is the SoS scope (universe of discourse), e.g., getting help in case of an individual homecare emergency, handling customer orders once being processed, or revealing decision-making on challenging production processes.

As subject-orientation has been created by Business Process Management (BPM) practitioners for practitioners the approach has been applied in various (re-)design settings (cf. [34,35]). In the course of the performed case studies behavior patterns, or even reference models, evolved, sometimes guided by knowledge management methods (cf. [36]). In the tradition of the design thinking approach

these behavior patterns need to undergo application and review cycles to allow learning for further refinements and developments. They are offered not as a template, but rather as an opportunity for thought simplifying reflection of experienced/ envisioned situations and a corresponding model for SoS construction. In contrast to traditional pattern development, subject-oriented patterns focus on the communication structures and, thus, likewise representation of functional and interactional behavior. In the following we report on the patterns that could be revealed so far. We continue following the order processing scenario aiming to capture dynamic customer and supplier behavior, as well as product/service development.

Figure 5 shows patterns for proactive and reactive behaviors in subject-oriented notation. Reactive behavior is based on temporal sequences of loosely coupled receive—do—send states. Proactive behavior requires asking for inputs, namely sending requests, while performing regular tasks, and awaiting response—see Figure 6 for the proactive order handling system. Utilizing such a pattern, e.g., encapsulated by a system component "ensure re-confirmation", proactive behavior along a customer or supplier relation can be modelled in an effective way. The system component can be activated in different contexts and activated at runtime, given running system components by the instantiated subject. In one case the process request can be performed by an automated decision-making component, e.g., checking the availability of goods, in another case process request requires human-computer interaction.

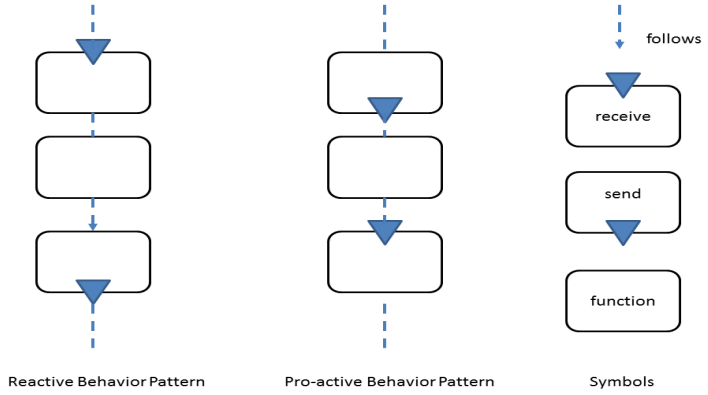

Figure 5. Subject-oriented representation of proactive and reactive behavior patterns.

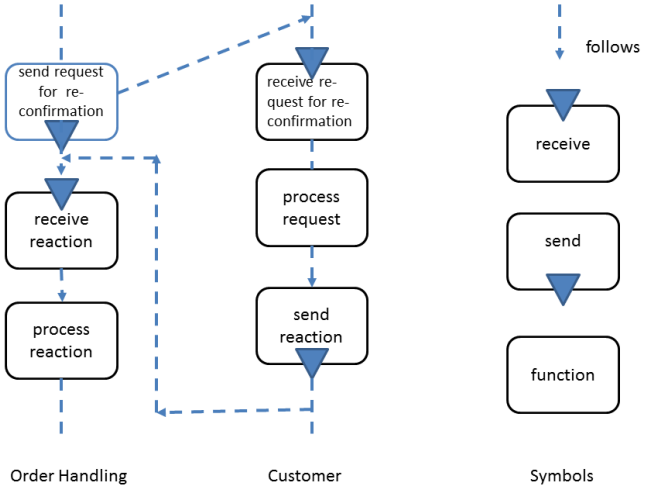

Figure 6. Proactive behavior pattern of order handling.

Of particular interest is the capability to combine pro- and reactive patterns. Consider a customer relation that is driven by incomplete orders over a longer period of time. In such cases, a proactive

reminder to complete the order could be effective for business operation, while the default could be a reactive SoS, waiting for the customer to become active. In that context, further examples utilizing that pattern are payment and supplier: consider traditional late providers and customers. They could become more reliable with respect to timely delivery when the proactive SoS component is activated.

A pattern-based approach of this kind can be a first step towards representing adaptive behavior of actors in self-organizing SoS. According to Sterling et al. [37], such systems should provide several qualities:

- Purposefulness, pursuing a goal and determining paths of action accordingly;
- Controlled autonomy, as a system component is able to pursue its own goals seemingly independently; and
- Situatedness of system components, i.e., being aware of the surrounding environment, being capable of perceiving changes, and responding appropriately.

Subject-oriented models allow meeting these qualities when explicitly representing the goal-relevant part of the SoS environment as subjects:

- A system component (subject) is a behavior abstraction for a specific purpose. It can stem from a functional role, type of application, or intention to capture behavior on an abstract level;
- A subject as a system component has controlled autonomy, as it encapsulates behavior to pursue a specific goal independently (while being interrelated through communication structures with other system components); and
- A subject is situated in an environment of subjects, and, most importantly, it is aware of this environment of the other subjects as it is exchanging messages within this environment (SoS). This mechanism allows not only perceiving changes and responding appropriately, but acquiring information about behavior of other subjects of the environment.

In particular, the latter property helps to proactively collect information of relevance to change behavior. Each SBD captures all possible local states a subject can be in, namely in terms of sending, receiving, and acting. They represent the actions it can perform, as well as the interfaces to other subjects. The local protocol is given by triggering actions internally (do), or externally (send). In this way the protocol for subject interaction is defined: receiving a message triggers an internal action of the addressed subject. The inputs to subjects trigger the evolution procedure in terms of proceeding from one local state to another depending on its own activities and the actions of other subjects (i.e., incoming messages from other subjects).

For further explicating SoS intelligence, North et al. [38] propose rules based on theories of individual rational behavior. Hereby, individuals collaborate with others when it is in their best (economic) interest to do so. Decision-making should be based on simple rules, in order to let rule structures evolve. In this way, bounded rationality can be taken into account. Reasoning regarding goals is progressively refined by means of procedures accounting for the limited knowledge and abilities of individual decision-makers (ibid.). In line with modular decision-making, patterns can be introduced, as shown in Figure 7. Such structures help mapping a system component's behavior effectively to executable patterns. Since subject-oriented SoS models can be executed after validation and in a concurrent way they allow for simulating behavior in complex situations.

A typical example is the aforementioned decision making on selecting the pro- or reactive pattern. Checking the reliability of a customer paying the bill, or of a supplier delivering a part in a certain period of time, requires applying a criteria-based selection. The pattern allows multiple criteria to be checked which in turn triggers subject behavior, either through activating other subjects or immediate acting (cf. considering a customer delaying payment of a good of high value which requires payment of suppliers in due time). In this way, behavior in SoS can be controlled on multiple system properties in a structured way.

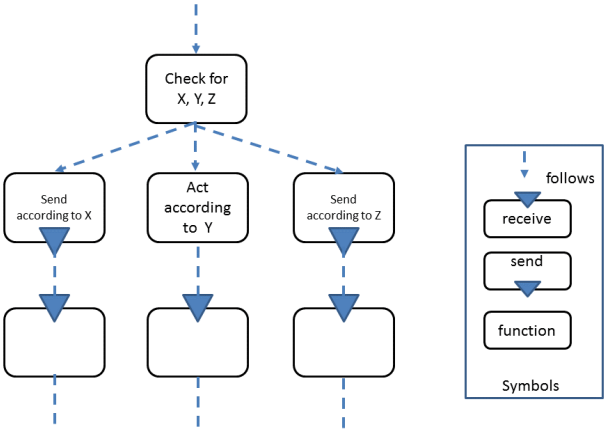

Figure 7. Decision-making patterns.

For adaptation subjects need to (i) become informed; and (ii) be selective with respect to their behavior. Following Gero et al. [39,40], the subject's "sensors" are monitor subjects that (actively) search the environment for situation-relevant data and produce direct input for an acting subject. In addition, a decision support subject can be invoked by an acting subject. It is provided with monitored information and situation-sensitive data to identify mismatches between the current and desired situation. Based on the results of the decision support process, the acting subject can decide which sequence of operations to execute.

Figure 8 provides the corresponding interaction scheme. It reveals that an <acting subject> can ask for both monitoring of the situation and decision support based on the monitored information. An <acting subject> is any subject in a situation that requires some action to accomplish a task. The monitor subject embodied in the environment either accepts requests to collect data on the current situation or does automatically receive data as a sensor, before processing and delivering the monitored data to the <acting subject>. The decision support subject is available for consultancy with respect to selecting the next action. This requires all available information on a certain situation.

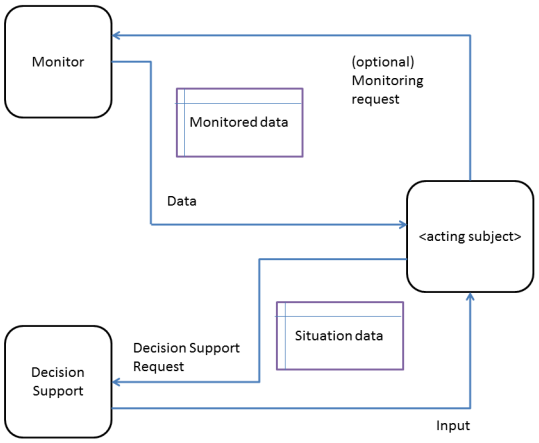

Figure 8. An <acting subject> can ask for monitoring a situation and decision support based on the monitored information.

Figure 9 demonstrates the possible impact for processing orders. Customer service asks the supply chain monitor subject for monitoring producing Part A. The monitored data of the supply chain are sent to customer service which in turn sends them together with order processing data to the decision support subject. Based on the results further handling of the order is triggered.

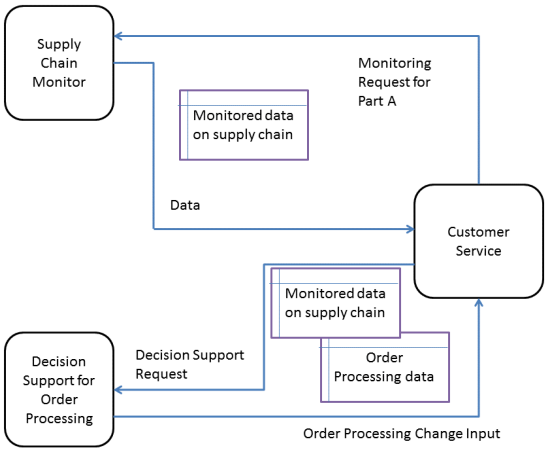

Figure 9. A customer service subject asks for monitoring the supply chain and decision support based on the monitored supply chain for changing order processing.

A monitoring subject can either receive and process environment data automatically or monitor the behavior of other subjects. Figure 10 shows both types of monitors. The environment monitor is a signal receiver processing them to deliver data, e.g., an event sensor indicating a delay in traffic due to changing weather conditions. The social monitor actively requests data from other subjects, e.g., whether certain data values have changed. Both could iterate for refining results.

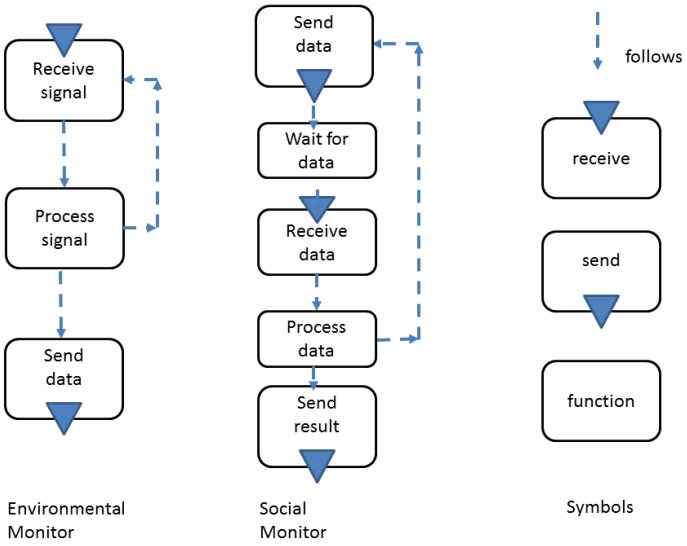

Figure 10. Push- and pull monitoring of a situation.

The communication of the <acting subject> can be conceptualized accordingly. Figure 11 details an <acting subject> releasing a request and waiting for the result (case of social monitoring). The request for monitoring can be modeled before any function state (action), thus providing for each critical function a preprocessing sequence. It considers environmental data beyond subject interaction and can also be captured in behavior descriptions. Once an <acting subject> has received monitor data it can act in response to the situation it is part of. The input data from the monitor are then processed along the subject's behavior. In case a subject acts requires decision support, it needs to activate the decision support subject.

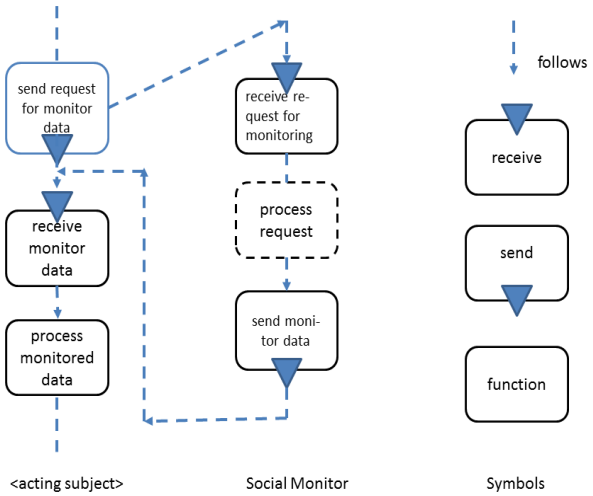

Figure 11. A <subject-in-charge> interacts with a monitor (encapsulating how the request is processed).

Referring to the running example, the supply chain is monitored by a social monitor. Once the request is completed they can be transferred to customer service for further processing, in our case asking for a decision based on order processing data.

The initial step hereby is requesting inference on situation-specific data, not only using the monitored data, but also data created or processed by the <acting subject> itself. The input data are sent to decision support as they describe the current situation of the <acting subject>. Processing the request for decision support requires a business rule engine that holds for the situation and environment (e.g., an organization) at hand. This concept can be cascaded for situation-relevant decision-making according to the scope of the SoS. The received results are processed based on available sets of rules and the situation data provided by the <acting subject>. The results are finally delivered to the <acting subject>—see Figure 12.

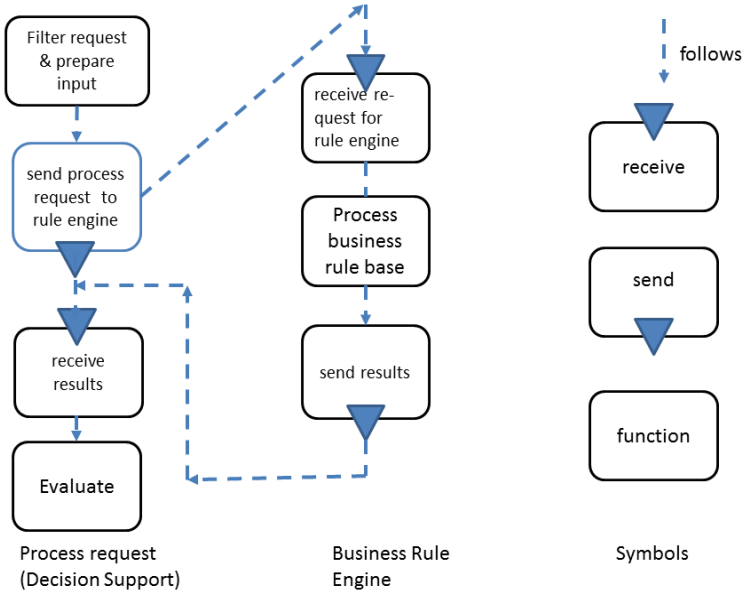

Figure 12. Rule processing.

The evaluated results are checked and a decision pattern can be triggered to complete a work task. In Figure 13 we exemplify a customer subject reflecting its order and deciding to opt for another product while ordering. The monitor subject is activated to find out whether the other product is

in stock. In order to avoid frustration in case it is not in stock, an alternative product is looked up—a decision support subject with respect to consumer behavior is activated and checked whether it is available, until the customer can be satisfied or informed about a lack of further alternatives.

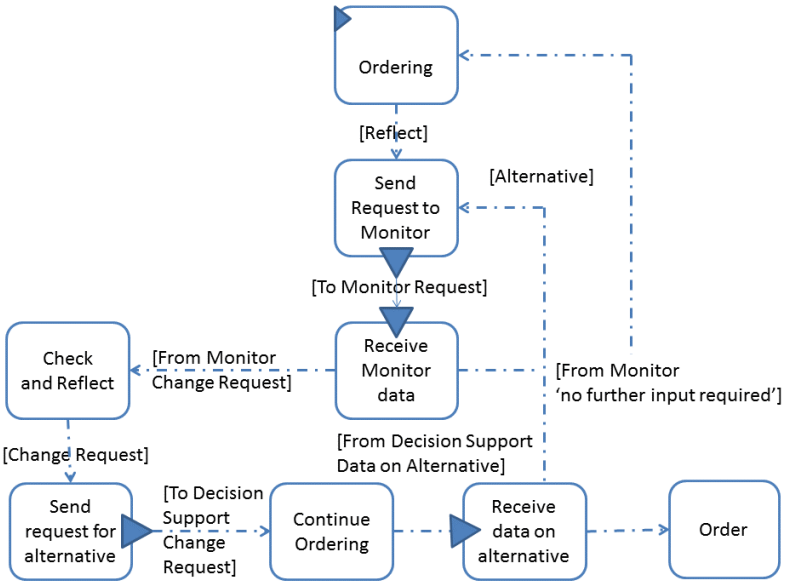

Figure 13. Case involving product monitoring and decision support.

This example shows a developed network as a SoS for a given scope. The utilized patterns allow composing a representation according to stakeholder needs and his/her understanding of a certain situation. Although it contains all relevant functions required for ordering, it can be implemented in a variety of ways, e.g., either by automated monitoring or manual observation.

Given monitoring and decision support, an acting subject (system) can perform in a reflexive way, using input data from the environment, either preprocessed from interaction behavior from other subjects or from observer components, such as sensors. Moreover, an acting subject can also activate a decision support subject. In this case, monitored data from the environment can be enriched with situation-sensitive data by the subject for further processing, e.g., according to general rules of encoding behavior. The results lead to informed decision taking of the acting subject at hand.

4. Conclusions

The penetration of digitalization into the variety of societal systems requires informed design of services and systems and involves stakeholders in the role of designers. Rather than looking for purely task or functional support, digital system design could follow a communication-driven perspective. In the presented work, system specifications encapsulate behavior of active entities (in terms of doing) and their exchange of messages (sending, receiving). A System-of-Systems specification is a network of self-contained behavior entities, developing according to communication needs and system-specific capabilities. Reoccurring behavior patterns can be used for complex system design, as they form the backbone of intelligent systems.

Once model elements, as well as contextual parameters, can be determined, a situation model can be constructed that can be used to learn from previous modeling steps to predict future activities. Such a model is also of particular use, in case reference or re-occurring behaviors are to be represented. It could guide stakeholders when describing a situation. Dai et al. [41] reveal opportunities to predict human activity and social links with greater accuracy, given human mobility data. As stakeholders move, they might be tagged or their device can be tracked location-wise. Stakeholders, when moving through their workspace, provide inputs to pattern mining and prediction opportunities according

to their digital footprints, including fixing an object to represent it as a subject or business object, annotating information, such as marking a piece of work incomplete, and digital media consumption (video or audio).

This rich data can be used to analyze human behavior patterns, build networks of actors, and predict future activities of individuals using their activity space. In particular, the predictions of an individual's future activity allow more relevant recommendations to be provided for modeling individual behaviors. While previous research mainly focused on location prediction—location prediction algorithms are based on the prior knowledge of the probability distribution of the mobile velocity and/or direction of movement—recent work investigates human behavior prediction based on only locations and GPS data obtained from smart phones [41].

Such prediction frameworks allow categorically quantifying upcoming target activity frequency such as no activity, normal user activity, and high-frequency activity, or other more refined categorization based on user (or business) data, e.g., referential role behavior or preferences. Hence, not only the existence of an activity, but also frequency-related quantities, can be taken into account. Recent prediction frameworks, as Dai et al.'s, tend to utilize the more general concept of partial repetitive behavior (instead of the stronger periodicity condition), and work with landmark behaviors in terms of representative user activity temporal patterns, where reference representations can be derived from probing certain patterns. Future research will go beyond predicting series of mobile phone features to modeling tool chain features, such as identifying an object, tagging it as a subject or context, storing it, and relating it to the already stored model elements.

Chen et al. [42] have explored parameters like the duration of task accomplishment, the amount of resources spent on tasks, etc., from cloud users to identify patterns for prediction. This mechanism can be used in the context of stakeholder (user) modeling to capture the dynamics of the series of incoming modeling data, in order provide an effective prediction towards the cognitive workload (i.e., the interarrival time between objects to be processed), including resources/tools that can be requested in terms of processing engines for pattern matching, validating, and processing models based on the historical data.

In summary, besides content- or domain-specific issues, research questions with respect to upcoming stakeholder behavior will be studied in future research.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Dittmar, A.; Dardar, L. Personal ecologies of calendar artifacts. *J. Interact. Sci.* **2015**, *3*, 2. [CrossRef]
2. Kallinikos, J.; Aaltonen, A.; Marton, A. The Ambivalent Ontology of Digital Artifacts. *MIS Q.* **2013**, *37*, 357–370.
3. Gruber, M.; De Leon, N.; George, G.; Thompson, P. Managing by design. *Acad. Manag. J.* **2015**, *58*, 1–7. [CrossRef]
4. Storni, C.; Binder, T.; Linde, P.; Stuedahl, D. Designing things together: Intersections of co-design and actor–network theory. *CoDesign* **2015**, *11*, 149–151. [CrossRef]
5. Acheson, P.; Daglo, C.; Kilicay-Ergin, N. Model Based Systems Engineering for System of Systems Using Agent-Based Modeling. *Procedia Comput. Sci.* **2013**, *16*, 11–19. [CrossRef]
6. Alam, I.I. Moving Beyond the Stage Gate Models for Service Innovation: The Trend and the Future. *Int. J. Econ. Pract. Theor.* **2014**, *4*, 637–646.
7. Tang, T.; Wu, Z.; Karhu, K.; Hämäläinen, M.; Ji, Y. Internationally distributed living labs and digital ecosystems for fostering local innovations in everyday life. *J. Emerg. Technol. Web Intell.* **2012**, *4*, 106–115. [CrossRef]
8. Zheng, M.; Song, W.; Ming, X. A Framework for Integrating Industrial Product-Service Systems and Cyber-Physical Systems. In *Proceedings of the 8th International Conference on Cross-Cultural Design, Toronto, ON, Canada, 17–22 July 2016*; Lecture Notes in Computer Science; Springer International Publishing: Berlin, Germany, 2016; Volume 9741, pp. 628–637.

9. Yoo, Y.; Boland, R.J., Jr.; Lyytinen, K.; Majchrzak, A. Organizing for innovation in the digitized world. *Organ. Sci.* **2012**, *23*, 1398–1408. [CrossRef]
10. Kim, K.; Altmann, J.; Baek, S. *Role of Platform Providers in Software Ecosystems*; Technology Management, Economics, and Policy Program (TEMEP). Discussion Paper No. 2015:120; Seoul National University: Seoul, Korea, 2015.
11. Jaradat, R.M.; Keating, C.B.; Bradley, J.M. A histogram analysis for system of systems. *Int. J. Syst. Syst. Eng.* **2014**, *5*, 193–227. [CrossRef]
12. Maier, M.W. Research challenges for systems-of-systems. In *Proceedings of the International Conference on Systems, Man and Cybernetics, Waikoloa, HI, USA, 10–12 October 2005*; IEEE: New York, NY, USA, 2005; Volume 4, pp. 3149–3154.
13. Kotov, V. *Systems of Systems as Communicating Structures*; Hewlett Packard: Bristol, UK, 1997.
14. Jamshidi, M. (Ed.) *System of Systems Engineering: Innovations for the Twenty-First Century*; John Wiley & Sons: New York, NY, USA, 2011; Volume 58.
15. IEEE-Reliability Society. Technical Committee on 'Systems of Systems'. In *Systems-of-Systems*; White Paper; IEEE: New York, NY, USA, 2014; p. 5.
16. Stary, C.; Wachholder, D. System-of-systems support—A bigraph approach to interoperability and emergent behavior. *Data Knowl. Eng.* **2016**, *105*, 155–172. [CrossRef]
17. Lane, J.A.; Epstein, D. *What Is a System of Systems and Why Should I Care?* Report USC-CSSE-2013-001; University of Southern California: Los Angeles, CA, USA, 2013.
18. Best, A.; Greenhalgh, T.; Lewis, S.; Saul, J.E.; Carroll, S.; Bitz, J. Large-system transformation in health care: A realist review. *Milbank Q.* **2012**, *90*, 421–456. [CrossRef] [PubMed]
19. Margaria, T.; Boelmann, S.; Doedt, M.; Floyd, B.; Steffen, B. Customer-oriented business process management: Vision and obstacles. In *Conquering Complexity*; Hinchey, M., Coyle, L., Eds.; Springer: London, UK, 2012; pp. 407–429.
20. Campos, J.; Lopez-Sanchez, M.; Rodriguez-Aguilar, J.A.; Esteva, M. Formalizing Situatedness and Adaption in Electronic Institutions. In *Proceedings of the COIN 2008*; LNAI 5428; Springer: Berlin, Germany, 2009; pp. 126–139.
21. Matheus, C.J.; Baclawski, K.; Kokar, M.M.; Letkowski, J.J. Using SWRL and OWL to capture domain knowledge for a situation awareness application applied to a supply logistics scenario. In *Proceedings of the International Workshop on Rules and Rule Markup Languages for the Semantic Web, Galway, Ireland, 10–12 November 2005*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 130–144.
22. Rolland, C.; Prakash, N.; Benjamen, A. A Multi-Model View of Process Modelling. *Requir. Eng.* **1999**, *4*, 169–187. [CrossRef]
23. Barwise, J.; Perry, J. Situations and attitudes. *J. Philos.* **1981**, *78*, 668–691. [CrossRef]
24. Dobbyn, C.; Stuart, S. The Self as Embedded Agent. *Minds Mach.* **2003**, *13*, 187–201. [CrossRef]
25. Senge, P. *The Fifth Discipline: The Art and Practice of the Learning Organization*; Doubleday: New York, NY, USA, 2005.
26. Frank, M. Engineering Systems Thinking: Cognitive Competencies of Successful Systems Engineers. *Procedia Comput. Sci.* **2012**, *8*, 273–278. [CrossRef]
27. Fleischmann, A.; Schmidt, W.; Stary, C.; Obermeier, S.; Börger, E. *Subject-Oriented Business Process Management*; Springer: Berlin, Germany, 2012.
28. Herrgard, T. Difficulties in diffusion of tacit knowledge in organizations. *J. Intell. Cap.* **2000**, *1*, 357–365. [CrossRef]
29. Parsaye, K.; Chignell, M. *Expert Systems for Experts*; Wiley: New York, NY, USA, 1988.
30. Grosskopf, A.; Edelman, J.; Weske, M. Tangible business process modeling methodology and experiment design. In *Business Process Management Workshops*; Lecture Notes in Business Information Processing; Rinderle-Ma, S., Sadiq, S., Leymann, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; Volume 43, pp. 489–500.
31. Brey, P. The Epistemology and Ontology of Human-Computer Interaction. *Mind Mach.* **2005**, *15*, 383–398. [CrossRef]
32. Harman, J.; Brown, R.; Kannengiesser, U.; Meyer, N.; Rothschädl, T. Model while you do. Engaging an S-BPM vendor on process modeling in 3D worlds. In *S-BPM in the Wild—Value Creating Practice in the Field*; Fleischmann, A., Schmidt, W., Stary, C., Eds.; Springer: Berlin, Germany, 2015.

33. Himma, K.E. Artificial Agency, Consciousness, and the Criteria for Moral Agency. *Ethics Inf. Technol.* **2009**, *11*, 19–29. [CrossRef]
34. Fleischmann, A.; Schmidt, W.; Stary, C. *S-BPM in the Wild: Practical Value Creation*; Springer Publishing Company, Incorporated: Berlin, Germany, 2015.
35. Neubauer, M.; Stary, C. *S-BPM in the Production Industry. A Stakeholder Approach*; Springer Publishing Company, Incorporated: Berlin, Germany, 2017.
36. Stary, C. Non-disruptive knowledge and business processing in knowledge life cycles-aligning value network analysis to process management. *J. Knowl. Manag.* **2014**, *18*, 651–686. [CrossRef]
37. Sterling, L.; Taveter, K. *The Art of Agent-Oriented Modeling*; MIT Press: Cambridge, MA, USA, 2009.
38. North, M.J.; Macal, C.M. *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation*; Oxford University Press: Oxford, UK, 2007.
39. Gero, J.S.; Fujii, H. A computational framework for concept formation for a situated design agent. *Knowl. Based Syst.* **2000**, *13*, 361–368. [CrossRef]
40. Gero, J.S.; Kannengiesser, U. The Situated-Function-Behaviour-Structure framework. In *Artificial Intelligence in Design'02*; Gero, J.S., Ed.; Kluwer: Dordrecht, The Netherlands, 2002; pp. 89–104.
41. Dai, P.; Ho, S.S. A smartphone user activity prediction framework utilizing partial repetitive and landmark behaviors. In *Proceedings Mobile Data Management*; IEEE: New York, NY, USA, 2014; Volume 1, pp. 205–210.
42. Chen, S.; Ghorbani, M.; Wang, Y.; Bogdan, P.; Pedram, M. Trace-Based Analysis and Prediction of Cloud Computing User Behavior Using the Fractal Modeling Technique. In *Proceedings Big Data Congress*; IEEE: New York, NY, USA, 2014; pp. 733–739.