On the Applicability of the Notion of Entropy for Business Process Analysis

Peter De Bruyn, Philip Huysmans, Gilles Oorts and Herwig Mannaert
Normalized Systems Institute
Department of Management Information Systems, University of Antwerp, Prinsstraat 13, 2000 Antwerp, Belgium
{*peter.debruyn, philip.huysmans, gilles.oorts, herwig.mannaert*}*@ua.ac.be*

Keywords: Entropy, Business Process Analysis, Normalized Systems.

Abstract: Contemporary organizations need to be able to dynamically adapt, improve and analyze their business processes. While many approaches are available in literature, few of them tend to use typical engineering concepts for this purpose. In this paper, we employ the concept of entropy as defined in statistical thermodynamics to advance the field of business process analysis. After reviewing some existing literature on entropy and its application in business topics, we show how earlier insights from entropy reasoning in Normalized Systems theory may offer opportunities to be applied in business process engineering as well. The necessary entropy concepts are defined in a business process context, entropy occurence in business processes is illustrated and some initial principles for controlling the resulting entropy are discussed. Finally, some implications for both theory and practice (e.g., Service-Oriented Architectures) are reviewed.

1 INTRODUCTION

In current business environments, organizations are increasingly confronted with more demanding customers and fiercer competitors forcing them to continuously change and adapt their business plans, products, services and business processes. Consequently, a lot of research on the *improvement, optimization and change of business processes* has been performed recently. Typical variations of terms referring to this issue include Business Process Reengineering (BPR), Core process redesign, process innovation, business process transformation, organizational reengineering, Total Quality Management (TQM), etcetera (O'Neill & Sohal, 1999). Many of those approaches mainly provide overall project management related best practices and life cycles such as "secure management commitment", "discover process opportunities" and "inform stakeholders" (Kettinger, Guha & J.T.C., 1995) or general optimization techniques such as "have those who use the output of the process perform the process" and "link parallel activities instead of integrating their results" (Hammer, 1990). Such general guidelines have clearly proven their value in the past. However, while often claiming terms as "*design*" and "*engineering*", it is remarkable to note how few approaches actually apply traditional engineering concepts as the core of their method to optimize or change the considered business processes.

Hence, in this paper we will try to advance the field of business process analysis by applying the *entropy* concept from thermodynamics for this purpose. We will do so by applying the theoretical framework of *Normalized Systems (NS)*. NS is an approach to design evolvable modular structures, based on theoretically proven theorems. While it was originally applied at the level of software architectures (Mannaert, Verelst & Ven, 2011, 2012), its relevance at the organizational level has already been demonstrated previously (Van Nuffel, 2011; Huysmans, 2011). However, the NS theorems were initially proven and derived from the concept of stability as defined in systems theory. Recently, the existing theorems have been confirmed by reasoning based on the concept of entropy while simultaneously suggesting new insights (and two new theorems) at the software architecture level (Mannaert, De Bruyn & Verelst, 2012). Consequently, we will try to make an initial attempt in this paper to verify whether it is valuable to analyze business processes from the NS entropy viewpoint as well and whether new insights seem to emerge correspondingly.

The remainder of this paper will be structured as follows. In Section 2 some related work on the concept of entropy will be discussed, including some previous attempts to use the concept in management and

De Bruyn P., Huysmans P., Oorts G. and Mannaert H.
On the Applicability of the Notion of Entropy for Business Process Analysis.
DOI: 10.5220/0004461901280137
In *Proceedings of the Second International Symposium on Business Modeling and Software Design (BMSD 2012)*, pages 128-137
ISBN: 978-989-8565-26-6

organizational research. Next, we will briefly summarize the essence of Normalized Systems theory in Section 3 and its extension based on entropy. Section 4 will discuss the usefulness of analyzing business processes from the NS entropy viewpoint, whereas Section 5 will deal with some of the resulting implications for theory and practice. Finally, some conclusions will be presented in Section 6.

2 RELATED WORK ON ENTROPY

In this section we will first provide some definitions and context regarding the concept of entropy. Next, we will discuss some earlier attempts of applying entropy reasoning to business and management topics.

2.1 Basic Concepts

Entropy, referring to the second law of thermodynamics, is generally considered to be a fundamental property in engineering sciences. The concept has been described and studied from many different perspectives, but all have basically the intention of describing the irreversibility of nature. Typically, more specific interpretations associated with entropy include (1) complexity, perceived chaos or disorder (Anderson, 2005), (2) uncertainty or lack of information (Shannon, 1948) and (3) the tendency of constituent particles in a system to dissipate or spread out (Leff, 1996). At one point, most interpretations can be brought back to the phenonomen that the modules (or in their most elementary form: particles) in a system have the natural tendency to interact (being coupled) in an uncontrolled way unless additional structure (i.e, energy or effort) is introduced in the system.

In this paper — as done previously in NS (Mannaert et al., 2012) — we will start from the statistical thermodynamics perspective towards entropy. Here, it was defined by Boltzmann in 1872 as the number of possible microstates consistent to the same macrostate of that system (Boltzmann, 1995). The *macrostate* refers to the whole of externally observable and measurable (macroscopic) properties of a system (typically temperature or pressure of a gas), whereas the *microstate* depicts the whole of microscopic properties of the constituent parts of the system (i.e., modules and particles). Generally, a particular macrostate (e.g., a certain temperature in a container) can be obtained by a myriad of different combinations of microstates (i.e., many different configurations of the molecules embedded into the container resulting in the same temperature). The higher the number of microstates consistent with that macrostate, the larger the degree of entropy becomes according to statistical thermodynamics. This relation can also be expressed in the following formula:

$$S = k_B log(W) \tag{1}$$

where S stands for the amount of entropy regarding a particular macrostate of a system, k_B equals the Boltzmann constant and W refers to the possible number of microstates consistent with the considered macrostate, given the assumption that each microstate is equally probable. According to this definition, entropy can then be seen as a measure of the information (or lack thereof) we have of the system, complying with the above mentioned interpretations of entropy as uncertainty or perceived disorder. In terms of the natural tendency of particles to interact, it can be seen as if they all “contribute” to the final resulting (observable) macrostate of the system through their mutual interactions, while it is unclear for the observer which exact configuration of particles (out of many possible configurations) brought it into being.

2.2 Entropy Applied in Business Topics

Several attempts have been made in the past to relate entropy concepts to business challenges and situations. Not claiming to be exhaustive or complete, we will illustrate some of them in this section.

For example, Trienekens, Kusters, Kriek & Siemons (2009) have elaborated the concept of entropy in the context of software development processes. First, they operationalized entropy as the amount of disorder, lack of structure and ‘instability’ [1] apparent in a system, being measured by the complexity (i.e., the number of interacting components) and change (i.e., the amount of changes over time) in the system. Also, they made the distinction between internal and external entropy: the former referring to the degree of ad-hoc organization inside the organization itself, the latter denoting the dynamism of its surrounding environment. As such, they conclude that both types of entropy are required to be in balance.

Janow (2004) studied the productivity and organizational decision making within firms based on Shannon’s entropy approach. In doing so, he found theoretical arguments to support the finding that organizations tend to become slower in their decision making

---

[1] The notion of stability according to Trienekens et al. (2009) was not formally defined in their article, although the meaning of invariability (absence of change) seems to be clearly suggested. This interpretation should not be confused with the definition as proposed by Mannaert et al. (2011, 2012) and employed in the remainder of this paper, as both interpretations clearly differ.

process as well as lose productivity when they grow. By analogy with the information theory developed for communication systems, an organization is considered as a network consisting of nodes (here: human beings) taking decisions and communicating them with each other. Each organization is then proven to reach "saturation" at a certain organizational size, resulting into organizational trashing and productivity implosion.

Next, entropy was also considered as being a measure for the degree of industry concentration (Horowitz, 1970) or corporate diversification (Jacquemin & Berry, 1979; Palepu, 1985). Again, the relation to the uncertainty interpretation of entropy is made is the sense that highly concentrated industries are considered to have a higher degree of entropy as it is more difficult is such situations to predict which of the several available companies will obtain the ultimate preference of a particular consumer.

Finally, the intent of Jung (2008) and Jung, Chin & Cardoso (2011) of measuring the degree of entropy present in business process models resulting in some uncertainty measures of process models, seems to be most closely related to our approach. Starting from Shannon's entropy as defined in information theory (Shannon, 1948), their aim is to measure the uncertainty or variability of workflow process models or the information gained by the process design. For example, it is concluded that the entropy of workflows consisting of purely serialized tasks or AND-splits have zero entropy as there is no uncertainty regarding which tasks are going be executed in process instantiations. On the other hand, the inclusion of XOR-plits, OR-splits and loops in a process increase entropy as one is not aware upfront which tasks are going to be executed (or even their frequency). Each of these uncertainties can then be derived, given an assumed probability of each branch or loop iteration. However, this approach differs from the approach we will take in Section 4 and onwards. First, Jung (2008) employs the entropy definition from information theory, whereas our approach will focus on the statistical thermodynamics perspective. Next, their measures are aimed at studying the design-time structure of business processes, whereas we will use entropy to focus on the run- or execution-time analysis of business processes.

3 NORMALIZED SYSTEMS

Normalized Systems theory (NS) is about the deterministic creation of evolvable modular structures based on a limited set of proven and unambiguous design theorems, primarily aimed at the design of evolvable software architectures. First, we will discuss the essence of NS in its initial form, i.e., starting from the stability point of view from systems theory. Next, the recent association and indications towards conformance with entropy concepts from thermodynamics will be highlighted.

3.1 NS and Stability

Normalized Systems theory initially started from the well-known maintenance problems in software applications, as was for instance already articulated earlier by Manny Lehman in his "*Law of Increasing Complexity*". This law states that software programs become ever more complex and badly structured as they are changed and evolve over time and hence become more and more difficult to adapt (Lehman, 1980). Based on the systems theoretic stability, this phenomenon was related to the concept of *combinatorial effects*: a change in modular structure of which the impact or effort to implement it, is related to the size of the system (on which the change is applied to) (Mannaert et al., 2011, 2012). Indeed, given the assumption that software applications keep on growing, this means that the same type of change requires more effort as time goes by. In contrast, systems which are free of such combinatorial effects (for a defined set of anticipated changes) are called *Normalized Systems*. These systems comply with stability as defined in systems theory as a bounded impact always results in a bounded output function (effort), even if time $t \rightarrow \infty$.

For this stability to be reached, the following design theorems were proposed and formally proven to be necessary conditions regarding the avoidance of combinatorial effects (Mannaert et al., 2011, 2012):

- *Separation of Concerns:* each concern (in terms of change drivers) should be separated in its own distinct action entity;
- *Separation of States:* the calling of action entities by other action entities should be performed in a stateful way;
- *Action Version Transparency:* the updating of action entities should not not have any impact on its calling action entities;
- *Data Version Transparency:* the updating of data entities should not have any impact on the action entities receiving the data entity as input or producing it as output.

As the construction of such stable software—strictly adhering to the above described principles—is not straightforward and current software constructs do not offer by themselves any mechanisms for software developers to obey them, a set of five elements was proposed: data elements, action elements, workflow elements, connector elements and trigger elements. As these elements offer recurring structures of constructs to facilitate the application of the previous principles, NS applications are traditionally build as an aggregation of instances of these elements.

3.2 NS and Entropy

Recently, efforts were made to explain the abovementioned in terms of entropy as defined in thermodynamics (Mannaert et al., 2012). First, the Boltzman definition in statistical thermodynamics was adopted considering entropy as the number of microstates consistent with a certain macrostate. As such, *microstates* were defined as binary values representing the correct or erroneous execution of a construct of a programming language. The *macrostate* is then to be seen in terms of loggings or database entries representing the correct or erroneous processing of the considered software system. In order to control the defined entropy, the earlier proposed theorems seem to be useful as well. Regarding the *Separation of States* principle, for instance, synchronous stateless pipeliness typically do not keep state when calling other action entities. As such, in case an error occurs, it not clear which particular action entity 'caused' the failure. In terms of the *Separation of Concerns* principle, each concern should again be isolated in its specific construct to avoid the creation of multiple microstates for one macrostates. This time however, concerns should be identified based on so-called uncertainty drivers instead of change drivers. The other two remaining principles, Data Version Transparency and Action Version Transparency, seem less applicable as they are related to compile-time and not for run-time analysis. However, two new theorems were suggested from this viewpoint:

- *Data Instance Traceability*, requiring each version and values of an instance of a data structure to be tracked;
- *Action Instance Traceability*, requiring each version and thread of an instance of a processing structure to be tracked.

Indeed, not exporting this information to an observable macrostate would lead to multiple possible microstates consistent with the same macrostate.

4 USING ENTROPY FOR BUSINESS PROCESS ANALYSIS

In order to extend the concept of entropy to business process analysis, we will first propose a definition of entropy, microstates and macrostates in a business process context. Next, we will illustrate the existence of such entropy by means of an example and discuss how NS principles can be helpful in reducing the amount of entropy in business process systems.

4.1 Defining Entropy Concepts in Business Process Systems

Our purpose is to apply the concept of entropy as defined in statistical thermodynamics (i.e., the number of microstates consistent with the same macrostate) to business processes. Consequently, a first effort should be directed towards interpreting macro- and microstates in such context. While the stability viewpoint (cf. Section 3) analyzes modular structures at design time, an entropy based analysis tends to investigate the modular structures during or after their execution (i.e., run time). Hence, regarding the *macrostate* (i.e., the whole of macroscopic properties of a system), typical externally observable properties of a business process might include:

- *throughput or cycle time* (how long did the process take to be executed?);
- *quality* and other output related measures (e.g., succesful or non-succesful completion of the process as a whole or the number of defects detected after the execution of the process);
- *costs* involved in the process;
- other *resources* consumed by the process (such as raw materials, electricity, human resources, etcetera).

Typical *microstates* (i.e., the whole of microscopic properties of a constituent modules or particles of the system) related to the above sketched macrostate might then comprise the throughput time of a single task in the process, the correct or erroneous outcome of a single task, the costs related to one activity or the resources consumed by one particular task of the considered business process. Analyzing instantiated business processes in terms of these defined macro- and microstates would then come down to management questions such as:

- which task or tasks in the business process was (were) reponsible for the extremely slow (fast) completion of this particular instance of the business process? ;

- which task or tasks in the business process was (were) responsible for the failure of the considered instantiated business process? ;
- which activities contributed substantially or only marginally to the overall cost or resource consumption of the considered business process (cf. cost-accounting and management approaches like Activity Based Costing)?

In case the answer to these questions is unambiguous and clear, the entropy in the system (here: business process) is low (or ideally zero) as a particular macrostate (e.g., the extremely long throughput time) can be related to only one or a few microstates (e.g., activity X took three times the normal duration to be carried out, whereas all other activities finished in their regular time span). On the other hand, when no direct answer to these questions can be found, entropy increases: multiple microstates (e.g., prolonged execution of activities X and/or Y and/or Z) could have resulted in the observed and possibly problematic macrostate (e.g., the lengthy execution of the overall process). This phenomenon seems to correlate well with the three basic entropy interpretations we listed in Section 2.1. First, business process analysis is more *complex* as the analyst has to consider the whole system at once, not being able to refine his problem analysis to certain clearly isolated parts of the system. Second, *uncertainty* is present when trying to perform remedial measures or optimizations. Indeed, it is simply not known where possible problems are situated, and the outcome or success of specific adaptations in the business process repository is uncertain as well. Finally, the *tendency of particles to dissipate* is reflected in the fact that the "traces" of one problematic activity are dispersed over the considered system as a whole. In essence, unless a consciously introduced separation is introduced, the information and outcome of all three possible problem causing activities (X, Y and Z) *interacts* before being exposed to the observer (in this case the measurements are aggregated). Hence, from an (enterprise) engineering viewpoint, it would seem appealing to control and reduce the entropy confronted with.

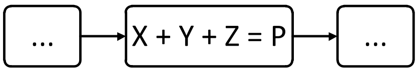
4.2 Illustrating Entropy in Business Process Models

In order to illustrate our conceptualization of the manifestation of entropy in business processes, consider the following example as depicted in the BPMN notation of Figures 1(a) and 1(b). Let us assume that both processes represent a part of a typical assembly line in which automobiles are finalized during their manufacturing. More specifically, we will claim that both process parts represent the manual attachment of the weels (X) and doors to the car (Y), as well as the spraying of a final liquid for the preservation of the color of the car's skeleton or "body" (Z). In Figure 1(a) all these activities are considered as one "finalization" activity P and only the throughput time for this entity of work is registered. A possible frequency graph of the throughput time of this way of modeling is represented in Figure 2(a) by means of a typical normal curve. Further, the sample mean $\overline{T_p}$ is given by the solid curve in this figure, while the "target mean"[2] set by the production manager is drawn by a dashed line. In case statistical hypothesis testing would point out that the observed mean actually significantly differs from the target mean, the production manager has no direct clue to determine which action is accountable for the prolonged production throughput time: both the attachment of wheels, doors or the preservation spray as well as any combination among them might be causing the delay. In entropy terms, the macrostate in this situation is the throughput time of process part P as a whole. The microstate is the combination of the throughput times of each of the constituting activities X, Y and Z. Consequently, the modeling as in Figure 1(a) exhibits a certain amount of entropy as it is not clear which activity causes the prolonged overall throughput time. We can call this confusion about the origin of a macrostate of a business process in terms of its microstate a *business process uncertainty effect*. Due to the inherent uncertainty and doubt in this situation, an imperative and profound study of the whole "finalization" activity should be performed in order to retrieve the cause and solve the problem identified.

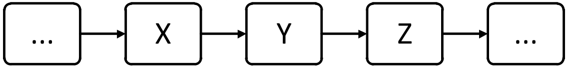
On the other hand, Figure 1(b) makes the obvious break up of P into the constituting tasks X (placement of the wheels), Y (placement of the doors) and Z (preservation spray) and their throughput measurements and means $\overline{T_X}$, $\overline{T_Y}$ and $\overline{T_Z}$. The corresponding normal curves of the frequency graphs are depicted in Figure 2(b). In case of the same problem of an enlarged overall average throughput time $\overline{T_p}$, the production manager is now able to refine his analysis towards the individual throughput times of X, Y and Z. Again, the real observed sample means are depicted in solid curves, the target means by the dashed line.

---

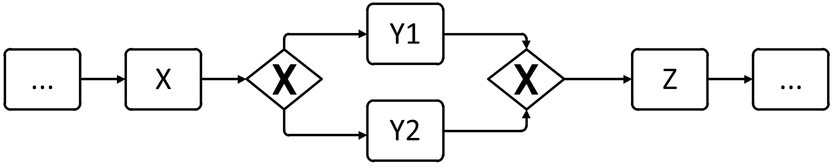
[2] In realistic production environments, typical upper control limits (UCL) and lower control limits (LCL) might be employed to determine significant deviations from the predefined goals. However, as this extra complexity does not add any further insight to our conceptual example (nor does it take the edge off our argument), we will leave this feature out of scope in this paper as its extention is straightforward.



(a) Process variant A



(b) Process variant B



(c) Process variant C

Figure 1: Three business process variants in BPMN, illustrating different degrees of entropy.

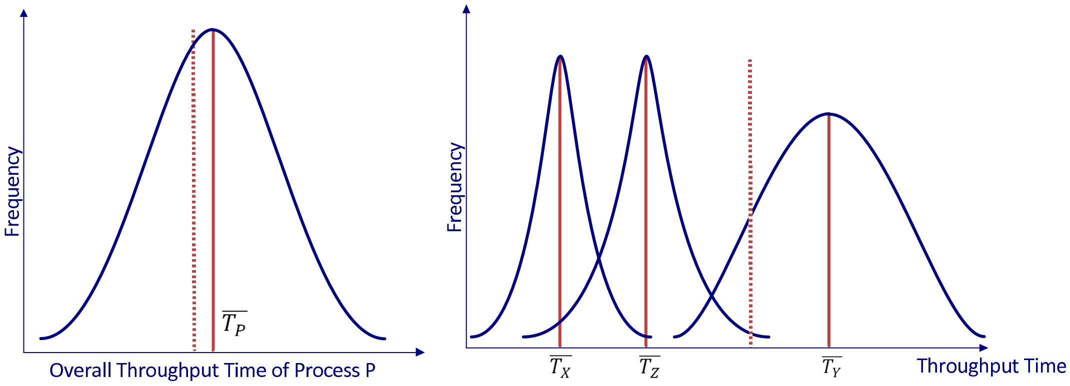
For tasks X and Z, the dashed lines are not visible as they coincide with the solide lines. Consequently, no issues regarding a prolonged execution time of these steps seem to be at hand. Regarding task Y however, a larger standard deviation and significant difference between the target mean and observed mean can be found. In order to improve the observed overall average throughput time $\overline{T_p}$, improvements regarding this specific business process task should be aimed for. Hence, entropy (and uncertainty) can be said to be reduced as the production manager can clearly see that the extended throughput time (i.e., the same macrostate as in our previous example) is caused by only one particular task, the other activities having regular throughput times (i.e., the macrostate is consistent with only one microstate configuration).

4.3 Illustrating the Need for Principles to Control Business Process Entropy

In the previous subsection, we illustrated how entropy generation and uncertainty effects can occur in business process models. Now we will illustrate how NS principles facilitate the control and reduction of entropy in the design of business processes.

Starting with the *Separation of States* principle, this theorem would call for the stateful executions of business processes. This would entail to include and keep a unique state after the execution of each business process activity. First, in terms of our previously discussed examples as visually represented in Figures 1(a) and 1(b), this implies that after each step a state should be pertained, registering the successful or unsuccessful completion of the step as well as the relevant observed system properties as costs, throughput time, resource consumption, etcetera. Not including these measure points or 'mashing them up' into one activity as in Figure 1(a) would after all lead to the uncertainty effect as described above. Second, these states need to be uniquely defined after each activity for maintaining transactional integrity and keeping record of exactly which activities have been executed. For example, consider the case of a business proces performing an application procedure for the entrance of potential future university students. The process contains multiple checks each resulting in a 'positive' state (after which the process can continue) or 'negative' state (in which the applicant is refused and the process terminated). Here, a unique state should be defined for each of the refusal situations in order to keep record of the precise reason why one person has been rejected. In case states are not uniquely defined (e.g., each negative outcome receives the same state 'refused') one will not be able to trace the obtained macrostate (i.e., a person has been refused) to the correct microstate configuration (i.e., which reason—which check—the person has been rejected for).

While the previous principle forces the usage of states in order to isolate activities in the system of which the macrostate is studied, the *Separation of Concerns* principle discusses the nature of the 'particles' that have to be separated by the states or measurement points. Stated otherwise, the division of the system into its constituent parts (and hence microstates) should not be done in an arbitrary way. Rather, concerns should be identified based on so-called *uncertainty drivers*: each separate part of the system of which the information should remain traceable for analysis purposes has to be isolated in its own construct (e.g., task or process). This implies amongst others that each task in a business process can contain only one single non-redundant concern. Suppose that in Figure 1(b) both tasks X and Y are combined with an electronic circuit general test inspection activity Q (or another registration, measurement or quality assessment activity) as both the wheel and door at-



(a) (b)

Figure 2: Corresponding frequency graphs of the throughput time, based on the various ways of modeling.

tachment are supposed to have a possible influence on the proper functioning of the electronic circuit of the car. In case Q is not properly isolated in a distinct task and not resulting in its own unique states, problem analysis of the overall troughput time pointing to X and/or Y is no longer unambiguous. First, the block $X + Q$ or $Y + Q$ should be further scrutinized to determine whether X, Y or Q was causing the throughput extension. Second, if investigation reveals a problem in the electronic circuit test Q, the problematic observation is escalated into several intermediate states and remediating actions should be taken at multiple process steps as well. As again multiple microstates can be associated with the same macrostate, entropy and uncertainty increase in such situations. Moreover, recurring task sequences with a clear business meaning of which information (e.g., its progress) is required to be recorded, should be isolated in their separate construct (here: business process) as well according to the same principle. Imagine again the application procedure in which the applicants have to pay an administration fee in order to be able to be assessed by the procedure and a registration fee later on if they have succesfully passed the whole procedure and decide to enroll. In case this redundant payment procedure (most likely repeated in many other business processes as well) is not properly separated, a bottleneck in this fixed task sequence might be much more difficult to be noticed. Consistent problems in the payment procedure would show up in many states (among many different processes) but hide its relation to the payment concern. Again, problems in such application procedure (macrostate) might be related to many interwoven subsequences (and hence microstates), resulting in a higher degree of entropy.

The principle of *Action Instance Traceability* would force us to keep track of the specific version of a task which is executed and relate every state (and measurement) to this specific task version. Also in a business process context, it is not unlikely to imagine situations in which a certain version of a task replaces a predecing task or multiple versions (variants) of one task exist concurrently. Consider for instance the situation in which our assembly line assembles both cars with two (Y_1) and four (Y_2) doors respectively. Elaborating on our investigation of througput time, it might seem reasonable to assume that (given the same technical equipment and resource availability) the attachment of doors for a two-door or four-door car variant might differ significantly. Consequently, the specific version should be recognized and traced individually as depicted in Figure 1(c) in order to allow further business process analysis and unambiguously relate particular macrostates to the correct microstate configuration (i.e., the specific versions) and avoid entropy generation.

Finally, *Data Instance Traceability* in a business process setting would prescribe us to keep track of the specificities of the information object processed by the business process in question. Applying the concept to our car assembly example, this means that characteristics of the car being assembled on a specific time slot should be tracked. Indeed, one can imagine that specific difficulties reflected in the states (e.g., extra costs, resource consumption, throughput time) can arise depending on the type (model) of car assembled on the same assembly line. Not tracking these specificities results in multiple possible ‘causes’ (microstates) consistent with the same ‘problem’ or ’fact’ (macrostate). Hence, registration of the relevant particularities of each processed information object can be considered as another ‘rule’ to control entropy.

5 REFLECTIONS

The concepts and principles discussed above might have several consequences for the design and analysis of business processes. We will first discuss some theoretical implications, followed by a few practical implications.

5.1 Theoretical Implications

Regarding the theoretical implications of employing an entropy viewpoint for analyzing business processes, several issues can be noticed. First, as was already mentioned in our analysis at the software level (Mannaert et al., 2012), the proposed analysis method includes the assumption that the introduced states are independent and decoupled. This means that they should only reflect the outcome of the activities performed in the module (here: task in a business process) they are attached to. Stated otherwise, the resulting state should not be dependent on the activities which have taken place earlier (e.g., in the business process) and with which the studied activity is coupled in a hidden (i.e., not explicitated) way. In a business interpretation, this could for instance be the case when the throughput time of task *B* is dependent on decisions taken earlier in task *A*. For example, it might be realistic to imagine a situation in which the employees responsible for the execution of task *A* choose to quickly (but poorly) finalize their task (in order to minimize their own throughput time), but having a pernicious consequence on the attainable throughput time of the execution of task *B* (for which the employees might then be forced to invest some extra time caused by the low-quality output of their predecessors). This obviously only leads to local optimizations, while preventing global optimization. Entropy reduction by imposing such a “structure” is limited and even misleading as the states here do not reflect the outcome of an isolated subsystem of the regarded overall system. Indeed, from an analysis viewpoint, it is no longer clear how the different subsystems (hence, microstate configuration) brought about the resulting macrostate. Hence, a clear interface between both activities should be defined, including (for example) unambiguous quality conditions as output criteria for activity *A* and preconditions for the execution of activity *B*. With the aim of preventing such phenomena, this assumption adds to our earlier call for the definition of completely and unambiguously defined interfaces for (organizational) modules in the first place (De Bruyn & Mannaert, 2012). Next, efforts should be directed towards avoiding such coupled modules.

Second, the initial application of the NS (entropy related) principles at the business process level demonstrated above, proved to be rather similar and parallel to the software level, and not to contradict with the guidelines by Van Nuffel based on the stability point of view (Van Nuffel, 2011). However, the work of Van Nuffel primarily concentrated on identifying business process instantiations of concerns to be separated in terms of *change drivers* based on the stability rationale of NS . Hence, it would be interesting to perform a similar study with the aim of identifying typical business process instantiations of concerns in terms of *uncertainty drivers* based on the entropy rationale of NS. This could lead to a parallel set of practical recommendations on how to design the modular structure of business processes and their constituting tasks. These guidelines would specifically allow for maximum entropy control and facilitate unambiguous analyses and tracking of outcomes obtained during execution time.

Finally, in the NS rationale at software level, the formulation of the principles (theorems) resulted in a limited set of elements or patterns (recurring structures of constructs) which are instantiated and aggregated consistently to build a software application. At the organizational level, such patterns (fixed structures of business processes) would be appealing as well when exhibiting both stability (i.e., proven absence of combinatorial effects towards a defined set of anticipated changes) and controlled entropy (i.e., ex-ante known measurements or metrics to allow for ex-post process analysis and optimization). The construction of such organizational patterns would facilitate pattern instantiation which is both stable and isentropic (i.e., having observable macrostates of the overall system which can be unambiguously traced to a microstate configuration). The formulation of such patterns is obviously a very challenging effort, and subject to future research. However, at the end of the following subsection, we will give an illustration of the usage of a recurring fixed pattern in reality for the execution of certain business functionalities, leading to entropy reduction.

5.2 Practical Implications

The reasoning in this paper holds irrespective of the implementation method. Therefore, concrete insights for supporting platforms for business processes such as Service-Oriented Architectures (SOA) can be made. In a SOA context, entropy is introduced in a business process when information regarding the micro-states of the service execution cannot be captured. However, the service concept specifically aims to hide the implementation of its functionality behind its interface. Consequently, micro-states which provide necessary knowledge during a service invocation cannot be captured. Consider the following definition of a service from an often-cited author: "*Services are self-describing, platform-agnostic computational elements that support rapid, low-cost composition of distributed applications. Services perform functions, which can be anything from simple requests to complicated business processes*" (Papazoglou, 2003, p. 3). This definition explicitly mentions the use of different distributed applications. In the context of the throughput example, the usage of such a service can result in delays caused by any application (e.g., because of the program logic, its hardware or its network connection). Consequently, the distributed nature of a service by itself increases the entropy when analyzing the business process.

Moreover, the design of services can introduce entropy as well. The definition mentions the service granularity. It seems that both fine-grained (i.e., "simple requests") and coarse-grained services (i.e., "complicated business processes") are considered valid by the definition. Nevertheless, our analysis indicates that information concerning fine-grained modular building blocks is required to lower the business process entropy. However, various authors discuss how currently a trend towards more coarse-grained services can be observed (e.g., Feuerlicht, 2006). These so-called "enterprise services" attempt to minimize the number of interactions needed to implement a given business function in order to reduce the complexity of the message interchange dialog (Feuerlicht, 2006). This example illustrates how considerations from a technical point of view (e.g., lowering the complexity of the message interchange dialog) may conflict with considerations from a business point of view (e.g., increasing the entropy during business process analysis). Dealing with such conflicts is an important issue for a paradigm which positions itself as the integration for business and IT perspectives. Nevertheless, the service definition by itself does not seem to provide guidance for entropy reduction. This task remains the responsibility of the service designer.

While no silver bullet is currently available to determine the "right" service granularity, certain interesting domain-specific solutions are emerging. These solutions apply the approach to controlling entropy as presented in Section 2 (i.e., enforcing a certain recurring *structure*) in order to ensure the availability of required knowledge. Consider the example of manufacturing organizations who are operating in a global supply chain. Governments of the national and international (e.g., the European Union) level define norms and regulations on these supply chains. The monitoring of these norms and regulations is executed by customs or quality inspections. In order to be able to perform these controls, certain information is required. Because of the heteroginity of processes used in these supply chains, any instance of these processes (e.g., a shipment) needed to be checked individually in order to ensure compliance. Consequently, these controls performed by the customs were very labor- and time-intensive, and organizations considered them to be a "necessary evil". The common goal of governments and organizations should be to ensure compliance to regulations, while having a minimal impact on the planning and execution of supply chains. Based on the introduction of structure in the organizational processes, results have already been achieved towards this goal. For example, member states of the European Union can grant the certificate of Authorized Economic Operator (AEO)[3] to organizations which follow, amongst others, customs compliance and appropriate record-keeping criteria. AEOs can benefit from simplified control procedures (e.g., fewer physical and document-based controls) during customs or quality controls. An AEO commits himself to structure his processes according to certain landmarks, which resemble the process states as discussed in the Separation of States theorem in Section 4.3. Governments which publish such landmarks can ensure that the information required for their controls can easily be gathered based on the registration of these process states. As a result, AOEs cannot use any SOA service with a granularity which spans multiple landmarks. Consequently, these landmarks effectively guide the selection of service granularity by imposing an appropriate structure on the processes, which is shown to be required to control the entropy in them.

6 CONCLUSIONS

In this paper, we explored the usage of the entropy concept for business process analysis. As such, the paper has multiple contributions, while suggesting several avenues for future research. First, we proposed a specific way for analyzing business processes from the typical engineering concept of entropy as defined in statistical thermodynamics. This approach seems to be contrasting with many ad-hoc or qualitative best practice approaches suggested in extant literature. By means of some pedagogical and conceptual examples, we showed that entropy has the tendency to show up in business processes which are arbitrarily

---

[3] See: http://ec.europa.eu/taxation_customs/customs/policy_issues/customs_security/aeo/

conceived, making it difficult to analyze them ex-post (e.g., in terms of quality, costs, resource consumption or throughput time). Consequently, in order to optimize and control such business processes, we argue that they should be purposefully *engineered* with the aim of controlling the entropy. Second, we proposed some initial principles for entropy control in business process systems (in analogy with previously defined principles at the software level). Third, our reflections demonstrated some of the implications of this entropy based reasoning for both theoretical and practical purposes, such as the design and usage of Service-Oriented Architectures. Obviously, the proposed principles might be further confined later on and business process instantiations of concerns from the entropy viewpoint (i.e., uncertainty drivers) could constitute an interesting path for future research, as would be the construction of organizational elements or patterns incorporating these issues.

ACKNOWLEDGEMENTS

P.D.B. is supported by a Research Grant of the Agency for Innovation by Science and Technology in Flanders (IWT).

REFERENCES

Anderson, G. (2005). *Thermodynamics of Natural Systems*. Cambridge University Press.

Boltzmann, L. (1995). *Lectures on Gas Theory*. Dover Publications.

De Bruyn, P. & Mannaert, H. (2012). Towards applying normalized systems concepts to modularity and the systems engineering process. In *Proceedings of the Seventh International Conference on Systems (ICONS)*.

Feuerlicht, G. (2006). Service granularity considerations based on data properties of interface parameters. *International Journal of Computer Systems science and Engineering*, *21*(4), 315–327.

Hammer, M. (1990). Reengineering work: Don't automate, obliterate. *Harvard Business Review*, *68*(4), 104 – 112.

Horowitz, I. (1970). Employment concentration in the common market: An entropy approach. *Journal of the Royal Statistical Society. Series A (General)*, *133*(3), 463–479.

Huysmans, P. (2011). *On the Feasibility of Normalized Enterprises: Applying Normalized Systems Theory to the High-Level Design of Enterprises*. PhD thesis, University of Antwerp.

Jacquemin, A. P. & Berry, C. H. (1979). Entropy measure of diversification and corporate growth. *The Journal of Industrial Economics*, *27*(4), 359–369.

Janow, R. (2004). Shannon entropy and productivity: Why big organizations can seem stupid. *Journal of the Washington Academy of Sciences*, *90*.

Jung, J.-Y. (2008). Measuring entropy in business process models. In *3rd International Conference on Innovative Computing Information and Control, 2008 (ICICIC '08)*.

Jung, J.-Y., Chin, C.-H., & Cardoso, J. (2011). An entropy-based uncertainty measure of process models. *Information Processing Letters*, *111*(3), 135 – 141.

Kettinger, W., Guha, S., & J.T.C., T. (1995). *Business process change: reengineering concepts, methods, and technologies*, chapter The Process Reengineering Life Cycle Methodology: A Case Study, (pp. 211–244). Idea Group Inc (IGI).

Leff, H. (1996). Thermodynamic entropy: The spreading and sharing of energy. *American Journal of Physics*, *64*(10), 1261–1271.

Lehman, M. (1980). Programs, life cycles, and laws of software evolution. *Proceedings of the IEEE*, *68*(9), 1060 – 1076.

Mannaert, H., De Bruyn, P., & Verelst, J. (2012). Exploring entropy in software systems: Towards a precise definition and design rules. In *Proceedings of the Seventh International Conference on Systems (ICONS)*.

Mannaert, H., Verelst, J., & Ven, K. (2011). The transformation of requirements into software primitives: Studying evolvability based on systems theoretic stability. *Science of Computer Programming*, *76*(12), 1210–1222. Special Issue on Software Evolution, Adaptability and Variability.

Mannaert, H., Verelst, J., & Ven, K. (2012). Towards evolvable software architectures based on systems theoretic stability. *Software: Practice and Experience*, *42*(1), 89–116.

O'Neill, P. & Sohal, A. S. (1999). Business process reengineering a review of recent literature. *Technovation*, *19*(9), 571–581.

Palepu, K. (1985). Diversification strategy, profit performance and the entropy measure. *Strategic Management Journal*, *6*(3), 239–255.

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

Shannon, C. (1948). A mathematical theory of communication. *Bell System Technical Journal*, *28*, 379–423.

Trienekens, J., Kusters, R., Kriek, D., & Siemons, P. (2009). Entropy based software processes improvement. *Software Quality Journal*, *17*, 231–243. 10.1007/s11219-008-9063-6.

Van Nuffel, D. (2011). *Towards Designing Modular and Evolvable Business Processes*. PhD thesis, University of Antwerp.