Guidelines to Design Evolvable Multiple GAAP Accounting Information Systems

Els Vanhoof
Department of Accountancy and Finance, University of Antwerp, Antwerp, Belgium
Els.vanhoof@uantwerp.be

Walter Aerts
Department of Accountancy and Finance, University of Antwerp, Antwerp, Belgium and Tilburg University, School of Economics and Management, Tilburg, The Netherlands
Walter.aerts@uantwerp.be

Abstract

Companies need to apply different sets of accounting rules, hence they need evolvable Accounting Information System (AIS) to be able to comply with regulation. This paper uses a mixed methods approach of case studies and design science to develop guidelines for designing an evolvable AIS that supports reporting in multiple Generally Accepted Accounting Principles (GAAP). In Vanhoof et al. (2014) combinatorial effects were identified in existing AIS structures, using Normalized Systems Theory (NST). NST provides guidelines to design a modular structure with an ex-ante proven degree of evolvability (Mannaert & Verelst 2009). We use these combinatorial effects to derive five guidelines to design evolvable AIS that support multiple GAAP. This is the first set of domain-specific evolvability criteria, using NST and a first application of evolvability in AIS. The matter is subject to further research, since the list of combinatorial effects is incomplete and we evaluate our guidelines theoretically.

Keywords

Normalized Systems Theory, Multiple GAAP, Design Science, Mixed Methods

Introduction

Companies need to provide various kinds of information to their stakeholders such as regulators, investors, customers, suppliers, or shareholders, including financial information. However, how recorded events should be processed into usable financial information and how this information should be presented in financial statements, is not the same in all institutional environments. Different regulators require different generally accepted accounting principles (GAAP) to be applied (Sinnett and Willis 2009). A company may be obliged to comply with information requirements of different regulators concurrently and hence might need to process the same underlying information using multiple GAAP resulting in different reports. For example, in the Netherlands a different GAAP exists for filing purposes and for tax purposes. Another example are listed companies: in the EU they need to use International Financial Reporting Standards (IFRS) to file their consolidated financial statements, while belonging to an international group can be an additional reason to apply multiple GAAP: when your parent company needs additional financial information in another GAAP.

Since companies use Accounting Information Systems (AIS) to support the processing and reporting of financial information, these AIS should be able to handle reporting in multiple GAAP. However, the AIS literature does not provide guidance as to how this issue should be tackled: the REA model of McCarthy (1982) describes how to design an accounting database so accounting conclusions can be derived from this data (Poels et al. 2005), without specifying how to process recordings in the database to derive financial information and eventually financial statements (Geerts and McCarthy 2002). In Vanhoof et al. (2014) the structures of AIS (able to report in multiple GAAP) of four companies are evaluated with respect to evolvability, using Normalized Systems Theory (NST). They provide a list of combinatorial effects (violations of NST) present in those AIS structures. As such, a theoretically sound explanation of
the lack of evolvability in AIS is demonstrated. Evolvability of AIS is important, because a set of anticipated changes to AIS can be implemented in such a way that the impact of the change has a bounded impact on the system in place.

This paper aims to contribute to the literature by not only explaining the lack of evolvability, but providing guidelines on how to build evolvable AIS which report in multiple GAAP. To design these guidelines we use a mixed methods approach of design science methodology and case studies, which is described in the methodology section. In the NST section, we elaborate on the theoretical framework of NST, which allows us to design evolvable AIS. Following design science research methodology, we distinguish three iterative phases in our work: problem statement, design and evaluation. In the problem statement section we describe the case studies (Vanhoof et al. 2014). The guidelines section treats the design phase, which consists of a description of the combinatorial effects (Vanhoof et al. 2014) and the development of guidelines. In the evaluation section we relate the guidelines to observations from the case studies and to relevant literature. We discuss the results in the conclusion section.

**Methodology**

Since our goal is to propose a new set of guidelines for designing AIS, a design science methodology, consisting of a problem phase, a design phase and an evaluation phase, is appropriate for our research. The guidelines are an initial step towards a method artifact, as defined by March and Smith (1995). In a first design iteration, the current state of the art of AIS design is evaluated. A set of case studies are conducted and describe the design of AIS in practice. These designs are evaluated on containing combinatorial effects. The identification of combinatorial effects is the end result of the first design iteration and is reported in Vanhoof et al. (2014). Since the empirical observation of cases is an important aspect of that first iteration, a mixed methods research methodology as proposed in Huysmans and De Bruyn (2013) is used.

In the second iteration we also use a mixed methods methodology where the point of interaction between the two methods (design science and case studies) is situated in the problem statement phase of the design science method (Huysmans and De Bruyn 2013). In that problem statement, we elaborate on the combinatorial effects identified in Vanhoof et al. (2014). In our design phase we use general principles proposed by NST to formulate guidelines on how to prevent the occurrence of these combinatorial effects. We demonstrate the applicability of these guidelines by revisiting the design observed in the first design iteration and propose improvements in terms of evolvability. Moreover, we evaluate the resulting designs to identify remaining combinatorial effects. This approach satisfies the evaluation criteria using logical proof as proposed by Hevner et al. (2004). In a next design iteration, we will then apply the identified guidelines to design a prototype AIS which is evolvable with regards to a set of anticipated changes.

The four cases (one insurance company, one pharmaceutical manufacturer and two transportation companies) are extensively discussed in Vanhoof et al. (2014). All companies apply two different GAAP. We have access to the notes taken during the original interviews. We use these case studies in our problem statement: the combinatorial effects identified in Vanhoof et al. (2014) are the rationale to develop our guidelines. In the evaluation section, we relate our findings to the case studies.

**Normalized Systems Theory**

NST uses the concepts of stability and entropy to derive theorems for the design of evolvable information systems (IS). The stability concept requires that a bounded input function results in bounded output values, even when an infinite time horizon is considered (an unlimited evolution of a system). Violations of this stability concept are called combinatorial effects: the impact of the change imposed on a system is not only dependent on the change itself, but also on the size of the system in place (Mannaert and Verelst 2009; Mannaert et al. 2011; Mannaert et al. 2012b). For example, if every employee in a company has its own stack of envelopes with the company’s logo on it and that logo needs to change because of an update of the company’s house style, the stack of envelopes of every employee needs to be replaced. This is a combinatorial effect, because the change is dependent on the size of the system i.e. the number of employees in the company.
NST consists of four principles that designers need to strictly adhere to, to create a modular structure that does not contain combinatorial effects (Mannaert and Verelst 2009; Mannaert et al. 2012b).

- **Separation of Concerns:** every change driver and concern has to be separated (in a module or action entity) from other change drivers or concerns in the system.
- **Data Version Transparency:** every data entity needs to be able to change without causing incompatibility with unchanged action entities.
- **Action Version Transparency:** every action element needs to be able to change without causing incompatibility with other unchanged action entities.
- **Separation of States:** when one action entity calls another action entity, state keeping needs to be used.

Boltzmann (1995) argues that entropy increases when the number of possible microstates that corresponds with one single macrostate increases. Hence as entropy increases, more details are lost. Applied to a business process, when entropy increases, detailed information about the process is lost: there is one piece of information (the macrostate) which is the result of multiple microstates on which you have no more detailed information (De Bruyn et al. 2013). Starting from the concept of entropy, Mannaert et al. (2012a) explore design principles that help with the development of evolvable information systems. These are compared to the existing NST theorems and two of the theorems are enforced by the concept of entropy in software i.e. separation of concerns and separation of states. The two other theorems (data version and action version transparency) are not relevant from a run-time entropy viewpoint, but the entropy concept does reveal additional requirements to make data and action elements evolvable: data instance and action instance traceability. These design rules require that certain additional information about the element is exported into an observable macrostate. This makes it possible to study every microstate of a processing function in isolation from the rest of the system since there are no multiple microstates possible that correspond with the same macrostate (Mannaert et al. 2012a):

- **Data instance traceability:** tracking of the version and values of every instance of a data structure that serves as an argument.
- **Action instance traceability:** tracking of the version of every instance of a processing structure and the thread in which it is embedded.

We use both concepts of stability and entropy to evaluate the guidelines we describe in the guidelines section. Van Nuffel (2011) proposes guidelines for the development of evolvable and modular business processes. Since they are generally applicable, they should also be applicable in the accounting domain. In our evaluation section we relate some of our guidelines to those of Van Nuffel (2011).

**Problem Statement**

In Vanhoof et al. (2014) the modular structure of AIS designs within cases are studied. We will describe this structure to the extent needed to be able to understand our guidelines. The combinatorial effects in these designs (Vanhoof et al. 2014) are described in the guidelines section.

The modular structure consists of a ledger, accounts, a chart of accounts, transactions and postings. A ledger is the collection of postings to different accounts. Those accounts are hierarchically structured and divided into logical categories in a chart of accounts. Transactions are recorded by postings to accounts in ledgers, those postings consist of the name and number of the accounts that increase/decrease and the respective amounts.

Vanhoof et al. (2014) identify three possible accounts designs and three possible posting designs in the modular structure of their cases. The accounts designs are: parallel accounts, parallel ledgers and separate company codes. In the parallel accounts design (accounts design 1) there are two sub-designs: duplicate accounts design (accounts design 1a) and areas design (accounts design 1b). In both cases there is only one chart of accounts which contains all accounts for all GAAP. For accounts design 1a all accounts are duplicated for each GAAP, whereas in the areas design (in case of two GAAP), three subsets of accounts exist: a common set of accounts, a set for the primary GAAP and a set for the additional GAAP. In the parallel accounts design, GAAP are separated in ledgers, based on the same chart of accounts. The company code design is described in two ways: accounts design 3a requires consolidation to add accounts from different company codes, which is not necessary in accounts design 3b. In accounts design 3a
separate company codes could use a different chart of accounts, which is not possible in accounts design 3b.

The three posting designs are difference posting (posting design 1), complete posting (posting design 2) and a special setup (posting design 3) (Vanhoof et al. 2014). Difference posting means that all postings resulting from transactions are made to the accounts of the primary GAAP and only the difference between the primary GAAP and the additional GAAP (in case of two GAAP) is posted to the accounts of the additional GAAP. When reports need to be made for the additional GAAP, difference posting requires the accounts of the primary GAAP to be added to the accounts of the additional GAAP. In the case of complete posting, transactions result in a posting to both the primary GAAP and the secondary GAAP. A special setup is needed when both types of postings are mixed, e.g. posting design 3: difference posting is used by default, except for one category of postings (financial instruments) complete postings are used. To be able to make reports in a correct way (for reporting the ledgers of the two GAAP are added), an additional posting is made to the additional GAAP ledger, which is the reverse the posting to the primary GAAP ledger.

Guidelines

In Vanhoof et al. (2014) combinatorial effects in each design (accounts design and posting design, as described in the problem statement section) are described, which allows us to contrast designs, see which designs cause combinatorial effects and investigate the manifestation of the combinatorial effects. After this analysis, we formulate a design guideline that is generally applicable and avoids the combinatorial effect. When a new guideline is proposed, we first reevaluate previous guidelines to check whether they are still applicable (do not conflict with new guidelines) or whether they should be defined more stringently. This iterative design allows us to adapt and/or refine guidelines as we gain more insight into the problem of multiple GAAP reporting. We structure our guidelines in the way combinatorial effects are structured in Vanhoof et al. (2014): introducing changes (four changes are proposed: creating an additional account, new revenue recognition criteria (effect on postings), new revenue recognition criteria (effect on posting module) and new measurement criteria for all GAAP) to the system and evaluating the impact.

Creating an additional account induces two combinatorial effects in a parallel accounts design (duplicated accounts), which do not exist in a parallel ledger design. Firstly, since in a (duplicated) parallel accounts design accounts are duplicated within the same ledger, creation of an additional account requires creating a new account in all duplicated sets. This is a combinatorial effect because the impact of the change is equal to the number of GAAP used. In a parallel ledger design, all ledgers are based on the same chart of accounts, so a new account only needs to be added once. Hence, no combinatorial effect arises from adding an additional account. Secondly in the parallel accounts design, to make a report, the right set of accounts needs to be selected from the chart of accounts. Hence, when an additional account is created, the selection criteria need to be altered, which imposes a combinatorial effect because the impact is equal to the number of GAAP used. In case of parallel ledger this does not impose a combinatorial effect: for reporting all accounts from the GAAP specific ledger (in case of complete postings) or all accounts from the GAAP specific ledger and the primary ledger (in case of difference posting) need to be selected.

Since the combinatorial effects exist in one design and not in another, we can state that regarding evolvability and with respect to these combinatorial effects the parallel ledger design is a better design than a parallel accounts design. From this conclusion we derive our first guideline for evolvable AIS: postings to different GAAP should be made in separate ledgers.

Creating an additional account can also create a combinatorial effect in accounts design 3, using separate company codes for different GAAP. When the separate company codes do not use the same chart of accounts, creating an additional account requires the account to be added in all charts of accounts. Hence, this is a combinatorial effect since the impact is equal to the number of charts of accounts. When separate company codes use the same chart of accounts, the combinatorial effect does not exist. Our second guideline hence states: all GAAP should use the same chart of accounts. The separate company code design is compatible with our first guideline, since the use of separate company codes implies the use of separate ledgers: it does not matter whether they are contained within the same or another company code.
The second proposed change only considers the effect of changing revenue recognition criteria on postings. Since our first guideline states that separate ledgers should be used, the consequences of changing revenue recognition criteria on the parallel accounts areas design are irrelevant. In case of difference posting, a combinatorial effect arises when the revenue recognition criteria require the posting to the primary GAAP to be changed: since postings to all additional GAAP depend on the difference between the primary and the additional GAAP, all postings to additional GAAP need to be changed. This is a combinatorial effect, because the impact of the change depends on the number of GAAP used. When using the complete posting design, the combinatorial effect does not exist, since the postings are made independently. We derive the following guideline: postings to different GAAP should be made independently. If we consider this guideline posting design 3 becomes irrelevant.

The third change identifies the effect of new recognition criteria on the posting module, consisting of several tasks. New recognition criteria impose a combinatorial effect when the new recognition criteria are encapsulated within the same task as for example, measurement criteria: a new task needs to be created containing the new revenue recognition criteria and the same measurement criteria. This will result in a structure in which all combinations of recognition and measurement criteria are contained in a separate task. If later on, certain measurement or recognition criteria change, they need to be altered in all tasks containing those criteria. This is not only the case for combining measurement and recognition criteria, but also for the concepts, presentation and disclosure criteria. In a design where all these ways in which GAAP can differ are separated from each other, only one version of every concept, recognition criteria, measurement criteria, presentation and disclosure exists and the combinatorial effect does not exist. Our fourth guideline states that every transaction should at least pass through five separate tasks (versions of the following tasks: concepts, recognition, measurement, presentation and disclosure) before any posting is made.

The last change is that the measurement criteria change for every GAAP. When all GAAP have separate measurement criteria, this change causes a combinatorial effect with an impact equal to the number of GAAP that use those measurement criteria. This combinatorial effect calls for reuse of measurement criteria across GAAP when they are identical: if the measurement criteria are separated into a task, independent of the GAAP they are used for, the combinatorial effect does not exist. Note that this is also valid for concepts, recognition, presentation and disclosure changes. Our fifth guideline hence states: every measurement method, definition of a concept, recognition criterion, presentation requirement and disclosure requirement that has a separate change driver should be separated in a distinct task, independent from the GAAP.

**Evaluation**

From the guidelines section we can derive a first set of five guidelines that should be followed for the design of an evolvable AIS that supports multiple GAAP. The guidelines are the following:

- **Guideline 1**: Postings to different GAAP should be made in separate ledgers.
- **Guideline 2**: All GAAP should use the same chart of accounts.
- **Guideline 3**: Postings to different GAAP should be made independently of each other.
- **Guideline 4**: Every transaction that could have an accounting impact should pass through at least five tasks (versions of the following tasks: definitions of concepts, recognition, measurement, presentation and disclosure) before any posting is made.
- **Guideline 5**: Every measurement method, definition of a concept, recognition criterion, presentation requirement and disclosure requirement that has a separate change driver should be separated in a distinct task, independent of the GAAP.

The first guideline is already a good practice used in 3 of our 4 case studies. Two of them use a separate company code, which is also compliant with the guideline: separate company codes imply separate ledgers. Moreover, the use of a separate ledger is integrated in the more recent versions of SAP and can as such be considered a good practice. Additional ledgers can be added, without impact on the existing ledgers, which is consistent with NST. Each GAAP represents a separate change driver, so in line with the first NST theorem (separation of concerns), the ledgers represent the boundary of the GAAP and hence should be separated from each other. The use of a separate ledger prevents that creating additional accounts leads to combinatorial effects in the selection rules for accounts for reporting purposes.
The second guideline to use the same chart of accounts for all GAAP, does not prevent the need for selection criteria: using one chart of accounts does not mean that all accounts should be used for all ledgers. It is possible that accounts need to be added that are only needed in one GAAP. This does not create a combinatorial effect, since when adding an additional account it can be specified in which ledger the account should be used. Determining which accounts are used in which ledger should however be done in such a way that the ledgers stay independent of each other. For example, you cannot define that ledger 3 has all accounts from ledger 2 + some additional accounts, rather you should define that ledger 3 has all accounts of the chart of accounts minus a certain set of accounts. This implies that selection criteria can be adjusted over time, but the impact of the change in selection criteria is limited to the number of ledgers that are affected. All accounts of a ledger should be used for reports, to prevent the need for other selection criteria within a ledger and to prevent a violation of the double-entry bookkeeping principle by not selecting all accounts that were each other’s counterpart in the double-entry recording of events. This all comes down to the same issue: how to define selection criteria. Selection criteria should be defined as generically as possible, to prevent combinatorial effects. How to define selection criteria in an evolvable way is a subject for further research.

The second guideline reduces the need for selection criteria and reduces the number of accounts that need to be maintained in the system. Therefore the guideline adheres to the separation of concerns principle. We explain this from the entropy viewpoint: when the same account exists twice (one for every GAAP), the same macrostate (the name of a certain account) can refer to multiple microstates (the account for GAAP 1 or for GAAP 2). Moreover, the separation of concerns principles demands that all accounts should only exhibit one change driver. This means that accounts should be defined on a detailed, fine-grained level. Accounts should be carefully defined to be able to reuse them in all GAAP. How the accounts should be arranged for reporting purposes can differ between GAAP. How to solve these further issues with regard to the chart of accounts is a subject for further research.

In the third guideline it is required to treat different GAAP independently of each other. Because they have a different change driver, they can also change independently of each other. We already saw a primitive separation of GAAP in one of our case studies, where postings to the different GAAP concerning financial instruments are almost made independently. The reason for this is that the difference between the two GAAP is so large that difference postings would become complicated. This is an indication that separating GAAP is a good practice, especially when GAAP differ much from each other.

Following the fourth guideline we can give a first impression of what an accounting business process should look like: we can identify five different tasks, that all represent separate change drivers. This is in line with the aggregated business process guidelines proposed by Van Nuffel (2011). The aggregated business process in this case is the posting of a transaction to a ledger. That aggregated process consists of at least five tasks at the aggregated level: definition of concepts, recognition, measurement, presentation and disclosure. All these tasks refer to separate elementary business processes, the aggregated business process only coordinates the tasks (Van Nuffel 2011). In that way whatever changes into one of the tasks, does not have an impact on the other tasks or the process as a whole. Each task should also be version transparent to prevent that a change within one specific task results in a combinatorial effect in the other tasks or in the aggregated process. For example, when a recognition criterion needs to be altered one should create a new version of the old recognition task so the other tasks do not need to be changed.

The reason many of our case companies do not separate GAAP from each other is because the differences between the GAAP are small and it would result in a lot of additional work, without having any better result. Therefore the fifth guideline requires that all concepts, recognition criteria, measurement methods, presentation guidelines and disclosure requirements that can be used in more than one GAAP are taken together. This relates to the business rule task of Van Nuffel (2011): he denotes that every business rule should be separated in a separate task. If we apply that in on our findings, every rule/guideline/criterion that relates to the concepts, recognition criteria, measurement methods, presentation guidelines or disclosure requirements, should be separated in a separate task. These tasks should then be assembled into versions of higher-level tasks. For example, all the rules about calculating fair value should be separated into separate task, but then for a particular type of transaction the task should be assembled in a way that it can be used to calculate the fair value of that specific type of transaction. This requires an intermediate level of calculating fair value can be done in different ways and within these different ways different tasks are needed that apply different business rules. Hence multiple layers of tasks are necessary.
to separate all concerns. This is consistent with the notion of entropy and the need to have knowledge of all the microstates (the specific combination of fine-grained tasks that are used) that cause a certain macrostate (the result of e.g. a certain calculation). Again, the version transparency theorems should be applied to prevent new versions of tasks to cause a combinatorial effect. From the notion of entropy the instance traceability theorems are relevant, since it is necessary to know which versions of which tasks are used for a certain posting.

By relating our guidelines to our case studies and existing literature, we find that parts of the combinatorial effects are resolved in existing software or that there are general guidelines i.e. Van Nuffel (2011) that relate to the issue. Although this literature and evidence is present, it is not evident to derive guidelines that are specific and detailed enough to resolve all combinatorial effects. At this point the guidelines are still general and more research is needed to refine and complement them.

Conclusion

In literature there is a lack of specific guidance on to process transactions in AIS that are able to report in multiple GAAP. Vanhoof et al. (2014) conduct case studies to identify combinatorial effects in existing AIS structures. These combinatorial effects show the lack of evolvability of the investigated AIS structures. Using a mixed method (Huysmans and De Bruyn 2013) consisting of design science and the case studies from Vanhoof et al. (2014), we develop a set of five guidelines to prevent combinatorial effects in the design of AIS that support multiple GAAP reporting. To design these guidelines we apply NST in the accounting domain. NST has shown its relevance for the design of software (Mannaert and Verelst 2009), business processes (Van Nuffel 2011) and enterprise architectures (Huysmans 2011) and now shows its relevance in the development of guidelines specific for the design of AIS that support multiple GAAP reporting. The guidelines have implications on both the design of the artifacts of the modular system (guideline 2: the design of the chart of accounts) as on the design accounting business processes (guidelines 1, 3, 4 and 5: the way postings are made to the ledger).

We evaluate the developed guidelines by relating them to the case studies and existing literature. We can conclude that although our guidelines provide a first guidance for designing evolvable AIS, they are not sufficient to avoid all combinatorial effects. Firstly, Vanhoof et al. (2014) did not provide an exhaustive list of combinatorial effects for multiple GAAP AIS. Secondly, our evaluation is merely theoretical, practical evaluation of these guidelines is a subject for further research. Our research contributes to the literature, since these guidelines are the first guidelines to develop evolvable multiple GAAP AIS. The use of the evolvability criterion (and the use of NST) is new in AIS research, moreover this is the first set of domain specific guidelines based on NST.

In further research additional case studies can reveal additional combinatorial effects that provide more insight to develop additional guidelines and go more in depth into the matter. Further research will also focus on the design of other issues within evolvable AIS like e.g. direct calculation of cash flows and reporting with XBRL. Another line of research will explore the design of evolvable information systems within other application domains like e.g. logistics, production, etc. Moreover, research efforts are being made to develop additional general principles at the business level e.g. Van Nuffel (2011) and the enterprise level e.g. Huysmans (2011). Another identified research aim is how to define selection criteria in a way that they are generic enough to be able to anticipate future changes.

REFERENCES


