# An Observational Approach to Practical Process Non-Conformance Detection

Sean Thompson
*La Trobe University, Bundoora, Australia*
*sean@nostin.com*

Torab Torabi
*La Trobe University, Bundoora, Australia*
*T.Torabi@latrobe.edu.au*

## Abstract

*A key challenge facing process non-conformance detection is to develop approaches not only effective across different domains and for different kinds of processes, but also to provide a practical and easy to implement solution. Since this is a process improvement area, the commercial realities of today dictate that if a solution cannot be implemented easily, quickly and cost-effectively then it is probably not worth implementing at all regardless of its effectiveness. The focus of this paper is on presenting a simple and practical approach to process non-conformance detection. We propose an approach based on specifying non-conformance rules to enhance an existing process specification without modifying it. The rules specified are used to check for conformance of process enactment data. Our methodology is flexible in its instantiation and because it is observatory, can be implemented with minimal interference with the existing process. We evaluate this approach using a case study.*

## 1. Introduction

There have been varied attempts at detecting process non-conformance presented in the literature. One of the main issues in this area is the lack of a practical approach that can be applied easily and generically with minimal impact on the process itself. We see a need for a solution that can be applied easily and quickly to a process irrespective of how the process has been initially specified. The solution should also be non-invasive, that is to say the process specification should not need to be changed to accommodate the methodology.

A process is a structured set of activities carried out to achieve a business goal. The activities that make up the process may be carried out in a number of different ways, including sequentially, concurrently, simultaneously, overlapping or in parallel [1].

Non-conformance is an instance of difference between a process specification and an enactment of that process. There are two types of non-conformance we recognize in this approach: deviations and inconsistencies. An *inconsistency* is a concept relating to the state of a process or its associated activities whereas a *deviation* is a concept relating to non-conformant transition between process activities. Cugola et al. first introduced this distinction in their paper on formalizing inconsistencies and deviations in human-centred systems [2].

The proposed approach is concerned with the enhancement of a process specification by defining *rule-sets* to complement it. The rule-sets contain rules that specify values, constraints and conditions the process enactment data must comply with. Non-conformance is detected by comparing enactment data with these rules.

### 1.1. Related Work

Research in this area first began to evolve in its current state following the 1999 paper by Cook and Wolf [3] on mining process event data in order to construct a real process specification from it. This reverse approach at modelling the process from data already recorded from existing enactments spawned an experiment by Huo et al. [4] that used this method to compare the discovered specification to further process enactments. The major problems with this idea is first the sheer amount of data needed to discover the initial process model and second, the fact that many iterations of process enactments need to take place before enough data can be gathered for the discovery. This inevitably means that the process may already be in a relatively mature state before the approach can be implemented. Some types of processes are also simply not enacted often enough to be suitable for this. Taking this into consideration, we devised our own approach to be applicable to a wide range of processes, from the beginning

and regardless of how often they are enacted in practice.

In 2000, Cîmpan, and Oquendo [5], published a fuzzy logic approach to non-conformance detection. The idea was to model a "perfect" simulation of how a process should be enacted and set the simulated model as the expectation for further enactments. Actual process enactments were then mapped to the expected behaviour to look for variations with the use of fuzzy sets theory. The authors conceded that the issue with this kind of idea was that the likelihood of a perfect process enactment was realistically close to zero, which virtually guarantees non-conformance of some kind. It also did not seem to address the possibility that processes could be enacted in a number of different yet equally valid ways. Nevertheless, to accommodate the shortcomings of expected perfection, the authors introduced the "tolerable" deviation concept in which they define certain situations in which certain deviations may be overlooked or tolerated by the approach. This concept is one we take a similar viewpoint in our own research with the inclusion of "exception types" regarding our own definition of non-conformance.

More recently, Mohammed et al. [6] presented a similar approach to our own, in which two co-existing process models are utilized. The approach works by using a guidance driven process model to dictate how the process should be carried out, similarly to a process specification. The other model is implemented only to observe the actions of human actors involved in the process. Comparisons are made between the two as the observations are made and recorded and "deviations" are detected as a result of these comparisons. Rule specification is also introduced in this approach but only in the form of determining the relative severity of a detected deviation and whether or not it should be tolerated and the process should be allowed to continue. This approach is therefore not strictly observational in its implementation.

In this approach, the authors consider the concepts of "deviation" and "inconsistency" differently to the way we consider it in our own research and the way it has been previously presented in the literature, such as in [2]. They do not consider the tern "non-conformance", rather they consider a "deviation" to constitute what we would otherwise consider to be non-conformance and that an "inconsistency" refers to abnormal process data resulting from a deviation that has occurred. Furthermore, the authors do not consider the possibility that non-conformance (or deviations in this case) may be instigated for a good reason – the actor has discovered what he or she believes to be a better way of doing things. It is important to recognize this possibility as an opportunity for process improvement and is one of the main reasons why in our own approach a strictly observational implementation is sought.

Finally in 2008, van der Aalst et al. [7] presented an approach in which the conformance of software services is checked in the context of its behaviour as expected by other software services within its environment. Behavioural conformance is tested via an implemented Petri net system from BPEL, which works by checking event data logs from the services against their custom specification. Two dimensions are defined in this context: *fitness* being the observed behaviour conforming to the specification and *appropriateness* being the resulting evaluation of whether the specification is an appropriate description of how the service should behave. This transitional type approach, as in non-conformance being tested through the expectation of related entities is in a similar vein to how we implement deviation detection pre and post conditions, which are applied to process activities. The transition between process activities can be thought of in this respect, with pre and post conditions being defined to govern how related activities should have been performed and when to constitute a legal transition into the respective activity.

The rest of this paper is comprised of two main sections. In section 2, we present our methodology and describe how our approach can be generically implemented with minimal intrusion into an already existing process. We detail our own definition of non-conformance and how it can be detected through the comparison of a process enactment to its related specification. Section 3 presents a simple implementation in which we applied our methodology to a case study over two distinct phases. The results are then analyzed and discussed before concluding the paper and stating our intention for related future research.

## 2. Methodology

### 2.1. Overview

The methodology we describe in this section aims to detect instances of non-conformance in process enactments via a method we call *process specification enhancement*. This approach is intended to be observational rather than a process support system, which have been criticized in the past for their rigidity and lack of flexibility [11, 12]. The proposed approach is

rule based and essentially complements an existing process specification without requiring any modifications to it, and without interfering with the way the process is enacted. We achieve this by defining specific *rule-sets* that stipulate how certain parts of the process should be performed. We then check the validity of these rules by comparing them against process enactment data.

The assumptions we make in this context are that the process enactment data is in such a form that it is possible to use it in rule comparisons and also that the process is broken into specific activities to which we can apply the rule-sets.

## 2.2. Non-conformance

Non-conformance is any instance where an enacted process has not conformed to its associated specification. In the context of this research, an instance of non-conformance is any recorded enactment data that fails to comply with the rules defining the constraints on how the process should have been performed.

As mentioned in the introduction, an instance of non-conformance can be one of two possible types: deviations or inconsistencies. Inconsistencies we define as part of a process enactment that has failed to comply with rules governing either general process constraints as a whole or with the values relating to a specific process activity. A deviation we define as when a process enactment fails to comply with the rules defining a process activities preconditions and post-conditions, representative of the legal transition between activities.

## 2.2. Rules

In order to enhance the process specification, we define rules that describe how parts of the process should and should not be legally carried out. Rules are used to check whether process enactment data complies with the specification and will return a Boolean value indicating whether or not the data has passed the comparison check. To illustrate, we use an example from our case study. Suppose there is an existing specification for a process activity called “Fill in survey”, in which a user answers some survey questions. One of the questions is “what is your current character level?”. We can “enhance” this specification by defining a specific non-conformance rule such as:

**Example rule:**
Character level must be of numeric value

The rule will check to see if the response data submitted by the user is numeric, and if so will return true to indicate that the check passed. If not, then the rule has not been complied with and non-conformance is detected because the false return will cause the rules rule-set to fail (covered in next section). The application and checking of this rule has no bearing on the existing specification for the process activity.

## 2.3. Rule-sets

Rule-sets are used to group related rules together to give non-conformance instances more meaning and also to make the application of this methodology more scalable and easier to maintain. Rule-sets consist of one or more rules and return a Boolean value in their own right. The Boolean value returned by a rule-set is indicative of the result of the validation of its associated rules.

For each rule-set, every rule associated with it must return true for the rule-set to pass. If one or more rules inside the rule-set return false, the rule-set itself will return false and indicate an instance of detected non-conformance. Rule-sets also have descriptions associated with them, which tell us more about the nature of the non-conformance it is checking for as opposed to solitary rules. We can continue from the previous example on rules to show an entire rule-set for checking the conformance of the specified character level:

**Example rule-set:**
**Rule 1:** Level must be numeric
**Rule 2:** Level must be greater than 0
**Rule 3:** Level must be less than 80

When we consider the concept of detected non-conformance in this methodology, it is an instance of where a rule-set has returned false.

## 2.4. Applying the rule-sets

Zero or more rule-sets can be applied to different parts of the process specification to detect non-conformance, meaning that we can easily check for as much or as little non-conformance as we need to simply by the definition and application of more and new rule sets. We can apply rule-sets to a process in four separate ways:

- General Process
- Activity
- Activity Pre-conditions
- Activity Post-conditions

With respect to activities, we can define general consistency rule-sets for the activity as an entity and also deviation rule-sets that apply to its pre and post conditions. Figure 1 shows the three ways we apply rule-sets to activities.

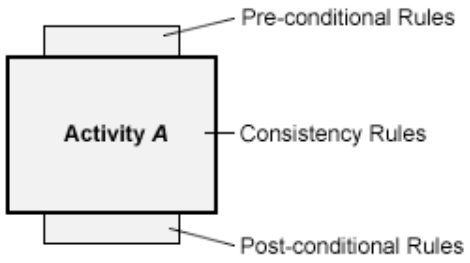


**Figure 1: Activity rule-sets**

General process rule-sets may also be defined which relate to some of the emergent properties that can only be measured by viewing the process as a whole. These could be rule-sets to govern things such as total activity counts, the absence of any required activities from the process or even just a minimum and/or maximum boundary on the total time duration of the process as a whole.

## 2.5. Activity sequencing

Since deviations refer to the transition between process activities, we also need to address the way a process may have specified the sequencing of its activities. This regards the conformance of the actual transition between activities rather than a comparison of sequencing patterns such as in [14]. Fortunately, the way this is best covered is through the application of pre and post-conditions governing the legal begin and end conditions of each activity. We can achieve this by simply defining rule-sets for each activity and assigning them as either pre or post-conditions. Instead of defining a situation where activity A has some sort of tag specifying that when it has concluded, activity B should be commenced, it works much better to assign a pre-condition to activity B stipulating that activity A should be successfully completed before activity B should begin.

Park et al. illustrate how pre-requisites can be applied to eLearning activities that govern when each activity can legally begin [8]. The pre-requisite approach stipulates certain conditions that must be complied with before a certain activity can legally commence. This tactic is employed in our own methodology with the use of pre-conditional rule-sets, being rules that govern when the activity can commence without causing non-conformance and also post-condition rule-sets being rules governing when the activity can legally conclude.

## 2.6. Formalization

As previously stated, we expect that an existing process specification would be apparent in some form, and that the process would be split into activities before we can begin applying this non-conformance methodology to the process specification. This is depicted in figure 2 in the grey area, which represents an existing process specification. Figure 2 also depicts the application of rule-sets to both the process as a whole entity along with its associated activities.

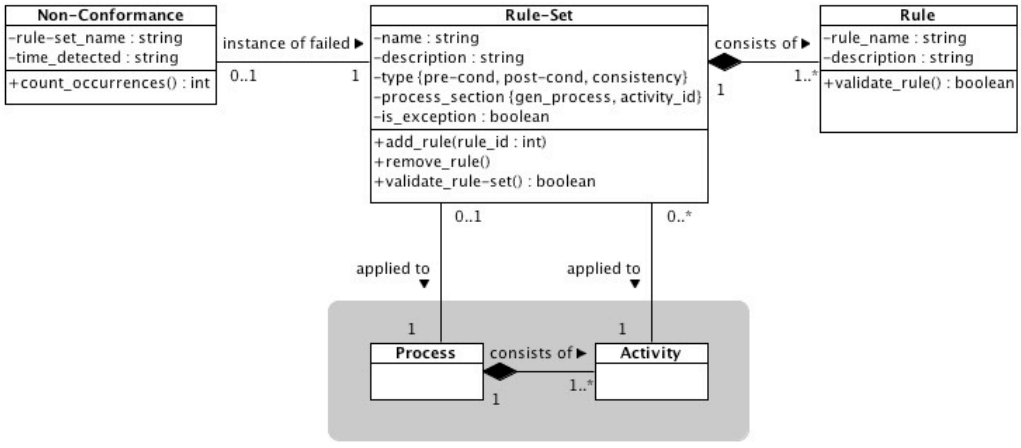


**Figure 2: UML representation of the methodologies application**

When a rule-set returns false, indicating that an instance of non-conformance has been detected, this is recorded according to the "non-conformance" class related to the rule-set. The rule-set provides general information about the non-conformance such as a description of what was looked for and found, the type and the section of the process it was found in. The detected instances represented by the "non-

conformance" class provide more detail about the actual detection characteristics, such as a timestamp or the occurrence count of the detections from the same rule-set. The attributes associated with a detected instance of non-conformance are therefore available through the relationship between the rule-set and the non-conformance classes.

## 3. Case Study

### 3.1 Overview

To test this methodology we built an online survey conducted in two phases based on the immensely popular online game World of Warcraft. An implementation of this nature served two important purposes. The first is that the observing and recording of process event data is required for a non-conformance methodology to be implemented, despite it not being related to the actual research at hand and this can be tricky to accomplish. The LAMP based online environment of the survey enabled us to achieve this without too much hassle and record exactly how participants went about completing it. Second, the large World of Warcraft community provided us with a steady amount of data to work with throughout both phases of the implementation and facilitated an adequate evaluation of our approach, as active players could be enticed to participate through the games official forums [10].

Survey participants were not aware that their responses were not of direct interest to us as researchers, rather the way that the survey was filled out, or rather how the process of filling in the survey was conducted. In the first implementation phase, the survey was presented to participants to fill out and their responses and behaviour were recorded. The results were then analysed and our non-conformance methodology applied. Using the non-conformance results only from what our approach detected over the first phase, some subtle changes were made to the survey in order to see if and how the survey could be improved using our approach. We then presented the survey again to a new set of World of Warcraft players to complete and again analysed the results.

### 3.2 Survey Basics

The survey was completed by a total of 410 different participants, 213 in the first phase and 197 in the second. No answers were forced in this survey by the use of radio buttons or drop down menus. Instead the default options had "not-specified" values, so every answer recorded was manually submitted by the participant. A few subtle changes were made between survey phases based on the non-conformance instances that were detected. Apart from these few changes, the survey remained the same in every other aspect. The modifications from phase 1 to phase 2 were as follows:

- Reduction of the number of captcha code characters from 6 to 4.
- Elimination of all lowercase characters, leaving only uppercase and numeric.
- Elimination of the O and the 0 from the list of possible code characters – they might look too similar for users to distinguish.
- A cap on the character level text field of 2 characters.
- A cap on the age text field of 2 characters.

A screenshot was taken of the phase 2 version of the survey and is depicted in figure 3.

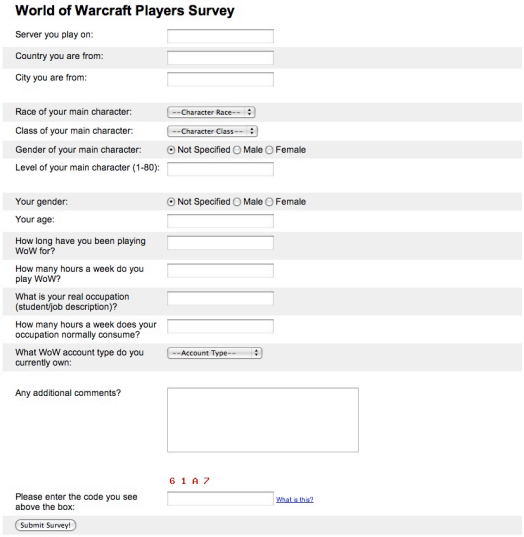


**Figure 3: Screenshot of the second phase survey**

The survey had three specified activities, which are shown in figure 4. Rule-sets were defined to complement the specification of all three activities as well as some general process rule-sets and these were built into the implementation in PHP code.

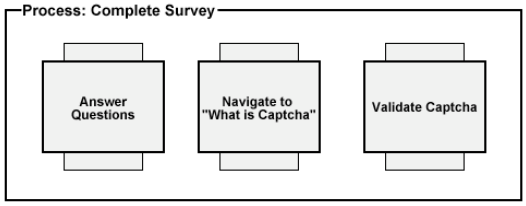


**Figure 4: Process Activity Breakdown**

### 3.3 Evaluation

From all of the non-conformance instances detected in the first phase, the most significant was the captcha code failures. The primary purpose of a

captcha on a website is to differentiate between human and non-human users [9] thus automatically removing any entries deemed to have been submitted by bots or crawlers. It seemed that people were having a lot of trouble filling in the code correctly, because the non-conformance detected related to records that were otherwise indicative of a genuine human response. By taking the corrective measures described in section 3.2 we were able to reduce the number of captcha failure non-conformance instances from 7.4% of participants to 0.5%.

We were pleasantly surprised to discover that a lot of the value in non-conformance detection is that it tends to point you to look in a direction you may not have otherwise, prompting you to spot issues with the process specification itself and not necessarily from the enactments.

Another useful feature was noticed upon discovering that when a record has a high number of non-conformance instances detected, it usually meant that a participant was deliberately messing with our survey. If we actually cared about collating the survey question data for research purposes, this would aid greatly in weeding out the invalid responses without having to manually iterate such a large dataset.

## 4. Conclusion and Future Work

The work presented in this research yielded some interesting results. We believe that the application of this methodology achieves the research goal of being simple and easy to implement however this is going to be dependent on the process in question and the accessibility of its enactment data. The effectiveness of our proposed approach is also going to be heavily dependent upon the competence of the administrator to identify and define appropriate non-conformance rules and rule-sets.

In this vein, we believe there is still an opportunity to further develop this approach through the forming of more generic rules which could be developed ready for implementation without the need of the developer to identify and specify them. These could be pre-defined rule-sets to govern things like a generic implementation of activity time duration boundary values or even domain specific rule-sets that could be relevant to a particular process types environment or domain. The consistent nature of resource usage in processes also lends itself well to this idea, and our existing research on resource based non-conformance detection [13] is a good candidate for exploring this in our future research.

## References


[1] Y. Rezgui, F. Marir, G. Cooper, J. Yip, and P. Brandon, "A Case-Based Approach to Construction Process Activity Specification", *International Conference on Intelligent Information Systems (IIS),* 8-10 Dec, 1997, pp. 293 – 297.

[2] G. Cugola, E. Di Nitto, A. Fuggetta, and C. Ghezzi, "A framework for formalizing inconsistencies and deviations in human-centered systems", *ACM Transactions on Software Engineering and Methodology (TOSEM),* Volume 5 Issue 3, ACM Press, July 1996.

[3] J.E. Cook and A.L. Wolf, "Discovering models of software processes from event-based data", *ACM Transactions on Software Engineering and Methodology (TOSEM),* Volume 7 Issue 3, ACM Press, July 1998.

[4] M. Huo, H. Zhang, and R. Jeffery, "An Exploratory Study of Process Enactment as Input to Software Process Improvement", *International Conference on Software Engineering,* Shanghai, China, 2006.

[5] S.Cîmpan, and F. Oquendo, "Dealing with software process deviations using fuzzy logic based monitoring", *ACM SIGAPP Applied Computing Review*, Volume 8 Issue 2, ACM Press, December 2000.

[6] M. Kabbaj, R. Lbath, and B. Coulette Bernard, "A Deviation-tolerant Approach to Software Process Evolution", *Ninth international workshop on Principles of software evolution (IWPSE'07)*, Dubrovnik, Croatia, September 2007.

[7] W.M.P. van der Aalst, M. Dumas, C. Ouyang, A. Rozinat, and E. Verbeek, "Conformance checking of service behavior", *Transactions on Internet Technology (TOIT)*, Volume 8 Issue 3, ACM, May 2008.

[8] N. Park, H. Kim, K. Kim, H. Park, J. Chun, and Y. Hwang; "An eLearning Activity Control Model for SCORM's Sequencing Prerequisites", *Fourth International Conference on Semantics, Knowledge and Grid*, IEEE, 3-5 Dec. 2008, pp. 322 – 329.

[9] Official CAPTCHA site, http://www.captcha.net.

[10] World of Warcraft official game forums, http://forums.worldofwarcraft.com/index.html.

[11] M.Ramage, "Engineering a smooth flow? The links between workflow and business process reengineering", MSc thesis 1994, University of Sussex, England.

[12] L.Aversano, and G. Canfora, "Introducing eservices in business process models", *Proceedings of the 14th international conference on Software engineering and knowledge engineering SEKE '02*, ACM Press, July 2002.

[13] S. Thompson, and T. Torabi, "Towards Formalizing Resource Based Non-Conformance in Business Processes", *19th Australian Software Engineering Conference ASWEC*, Perth, Australia, March 2008.

[14] R. Hamid, A. Johnson, S. Batta, A. Bobick, C. Isbell, and G. Coleman, "Detection and explanation of anomalous activities: representing activities as bags of event n-grams", *Conference on Computer Vision and Pattern Recognition*, IEEE, Volume 1, 20-25 June 2005, pp. 1031 – 1038.