Finding fault: Detecting issues in a versioned ontology

Maria Copeland, Rafael S. Gonçalves, Bijan Parsia, Uli Sattler and Robert Stevens

School of Computer Science, University of Manchester, Manchester, UK

Abstract. Understanding ontology evolution is becoming an active topic of interest to ontology engineers, e.g., we have large collaborative developed ontologies but, unlike software engineering, comparatively little is understood about the dynamics of historical changes, especially at a fine level of granularity. Only recently has there been a systematic analysis of changes across ontology versions, but still at a coarse-grained level. The National Cancer Institute (NCI) Thesaurus (NCIt) is a large, collaboratively-developed ontology, used for various Web and research-related purposes, e.g., as a medical research controlled vocabulary. The NCI has published ten years worth of monthly versions of the NCIt as Web Ontology Language (OWL) documents, and has also published reports on the content of, development methodology for, and applications of the NCIt. In this paper, we carry out a fine-grained analysis of the asserted axiom dynamics throughout the evolution of the NCIt from 2003 to 2012. From this, we are able to identify axiomatic editing patterns that suggest significant regression editing events in the development history of the NCIt.

1 Introduction

This paper is part of a series of analyses of the NCIt corpus [1,2], the earlier of which focus on changes to the asserted and inferred axioms. The current analysis extends previous work by tracing editing events at the individual axiom level, as opposed to the ontology level. That is, instead of analysing the total number of axioms added or removed between versions, we also track the appearance and disappearance of individual axioms across the corpus. As a result, we are able to positively identify a number of regressions (i.e., inadvertent introduction of an error) which occur over the last ten years of the development of the NCIt ontology, as well as a number of event sequences that, while not necessarily introducing errors, indicate issues with the editing process. We are able to do this analytically from the editing patterns alone.

2 Preliminaries

We assume that the reader is familiar with OWL 2 [3], at least from a modeller perspective. An ontology $\mathcal{O}$ is a set of axioms, containing logical and non-logical (e.g., annotation) axioms. The latter are analogous to comments in conventional programming languages, while the former describe entities (classes or individuals) and the relations between these entities via properties. The *signature* of an ontology $\mathcal{O}$ (the set of individuals, class and property names in $\mathcal{O}$) is denoted $\widetilde{\mathcal{O}}$.

We use the standard notion of *entailment*, an axiom α entailed by an ontology $\mathcal{O}$ is denoted by $\mathcal{O} \models \alpha$. We look at entailments of the form $A \sqsubseteq B$ where A and B are class names i.e., atomic subsumptions. This is the part of the type of entailment generated by the classification reasoning task, a standard reasoning task that forms the basis of the 'inferred subsumption hierarchy' .

Finally, we use the notions of effectual and ineffectual changes as follows:

Definition 1. *Let $\mathcal{O}_i$ and $\mathcal{O}_{i+1}$ be two consecutive versions of an ontology $\mathcal{O}$.*
An axiom α is an addition (removal) *if $\alpha \in \mathcal{O}_{i+1} \backslash \mathcal{O}_i$ ($\alpha \notin \mathcal{O}_i \backslash \mathcal{O}_{i+1}$).*
An addition α is effectual *if $\mathcal{O}_i \not\models \alpha$ (written as $EffAdd(O_i, O_{i+1})$), and* ineffectual *otherwise (written as $InEffAdd(O_i, O_{i+1})$).*
A removal α is effectual *if $\mathcal{O}_{i+1} \not\models \alpha$ (written as $EffRem(O_i, O_{i+1})$), and* ineffectual *otherwise (written as $InEffRem(O_i, O_{i+1})$) [1].*

3 Conceptual Foundations

Prior to the study of fault detection techniques, we establish a clear notion of the type of faults we are trying to isolate. In all cases, we define a *fault* as deviation from the required behaviour. In Software Engineering, software faults are commonly divided into functional and non-functional depending on whether the fault is in the required functional behaviour (e.g., whether the system is acting correctly in respect to its inputs, behaviour, and outputs) or whether the fault is in the expected service the system needs to provide (i.e., whether the (correct) behaviour is performed *well*). Functional and non-functional faults can be further subdivided based on the impact to the system and/or to the requirements specifications. For example, functional faults can be divided into fatal and non-fatal errors depending on whether the fault crashes the system. Generally, crashing behaviour is always a fatal fault, however it might be preferable to encounter a system crash instead of a non-fatal fault manifested in some other, harder to detect, manner. Faults that impact the requirements may be implicit, indeterminate (i.e., the behaviour might be *underspecified*), or shifting. A shifting specification can render previously correct behaviour faulty (or the reverse), as faults are defined as deviations from the "governing" specification. For convenience, we presume throughout this study that the specification is stable over the lifetime of the examined ontology, i.e., we expect the notation of 'acceptable model' or 'acceptable entailment' to be stable throughout the lifetime of the ontology.

We also restrict our attention to the *logical* behaviour of the ontology, and we approximate this by sets of desired entailments. This restriction might not reflect the full behaviour of an ontology in some application as 1) many entailments might be irrelevant to the application (e.g., non-atomic subsumptions for a terminologically oriented application) or 2) the application might be highly sensitive to other aspects of the ontology, including, but not limited to, annotations, axiom shape, and naming patterns. However, these other aspects are less standardised from application to application, so are rather more difficult to study externally to a given project. Furthermore, faults in the logical portion of an ontology both can be rather difficult to deal with and affect these other aspects. With this in mind, we define a logical bug as follows:

Definition 2. *An ontology $\mathcal{O}$ contains a (logical) bug if $\mathcal{O} \models \alpha$ and α is not a desired entailment or $O \not\models \alpha$ and α is a desired entailment.*

Of course, whether a (non)entailment is desired or not is not determinable by a reasoner — a reasoner can only confirm that some axiom is or is not an entailment. Generally, certain classes of (non)entailments are always regarded as errors. In analogy to crashing bugs in Software Engineering, in particular, the following are all standard errors:

1. $\mathcal{O}$ is inconsistent i.e., $\mathcal{O} \models \top \sqsubseteq \bot$
2. $A \in \widetilde{\mathcal{O}}$ is unsatisfiable in $\mathcal{O}$ i.e., $\mathcal{O} \models A \sqsubseteq \bot$
3. $A \in \widetilde{\mathcal{O}}$ is tautological in $\mathcal{O}$ i.e., $\mathcal{O} \models \top \sqsubseteq A$

In each of these cases, the "worthlessness" of the entailment is straightforward[1] and we will not justify it further here. That these entailments are bugs in and of themselves makes it easy to detect them, so the entire challenge of coping with such is in explaining and repairing them.

Of course, not all errors will be of these forms. For example, in most cases, the subsumption, $Tree \sqsubseteq Animal$ would be an undesired entailment. Detecting this requires domain knowledge, specifically, the denotation of `Tree` and `Animal`, the relation between them, and the intent of the ontology. If there is an explicit specification such e.g. a finite list of desired entailments, then checking for correctness of the ontology would be straightforward. Typically, however, the specification is implicit and, indeed, may be inchoate, only emerging via the ontology development process. Consequently, it would seem that automatic detection of such faults is impossible.

This is certainly true when considering a single version of an ontology. The case is different when multiple versions are compared. Crucially, if an entailment *fluctuates* between versions, that is, if it is the case that $\mathcal{O}_i \models \alpha$ and $\mathcal{O}_j \not\models \alpha$ where $i < j$, then we can conclude that *one* of those cases is erroneous. However, it is evident that $\mathcal{O}_i \not\models \alpha$ but $\mathcal{O}_j \models \alpha$ may not be as the fact that $\mathcal{O}_i \not\models \alpha$ as it might just indicate that the "functionality" has not been introduced yet. In what follows, we consider a sequence of $\mathcal{O}_i$, ... , $\mathcal{O}_m$ of ontologies, and use *i*, *j*, *k*, ... , as indices for these ontologies with *i*<*j*<*k*<... With this in mind, we can conservatively determine whether there are logical faults in the corpus using the following definition.

Definition 3. *Given two ontologies, $\mathcal{O}_i, \mathcal{O}_j$ where i<j, then the set of changes such that α is $\{EffAdd(O_i, O_{i+1}) \cap EffRem(O_j, O_{j+1})\}$is a* fault **indicating** set of changes *written as* FiSoC(i, j).

Note that if $\alpha \in FiSoC(i, j)$ either the entailment $\mathcal{O}_i \models \alpha$, thus $\alpha \in \mathcal{O}_i$, or the non-entailment $\mathcal{O}_i \not\models \alpha$ may be the bug in question and *FiSoC(i, j)* does not identify which is the bug. Instead the fault indicating set tells us that *one* of the changes introduces a bug. As mentioned earlier, the set shows the existence of a bug assuming a stable specification. Any subsequent findings of the same $\alpha \in FiSoC(i, j)$ is a fault indicate content regression. It is not surprising to find reoccurring content regressions due to the absence of content regression testing.

We can have a similar set of changes wherein the removal is *ineffectual* i.e., $\alpha \in \mathcal{O}_i$, $\alpha \notin \mathcal{O}_{i+1}$, but $\mathcal{O}_{i+1} \models \alpha$. Since the functionality of the ontology is not changed by

[1] There is, at least in the OWL community, reasonable consensus that these are all bugs in the sort of ontologies we build for the infrastructure we use.

an ineffectual removal, such a set does not indicate regression in the ontology. Indeed, such a set is consistent with a *refactoring* of the axiom, that is syntactic changes to the axiom that result in the axiom being strengthened or weakened based on the effectuality of the change [1]. Of course, if the added axiom is the bug, then the ineffectual removal from $\mathcal{O}_i$ to $\mathcal{O}_{i+1}$ would be a failed attempt to remove the bug. Without access to developer intentions or other external information, we cannot distinguish between these two situations. However, we can conclude that an iterated pattern of ineffectual changes is problematic. That is, even if the set of changes $EffAdd(O_i, O_{i+1}) \cap InEffRem(O_j, O_{j+1})$ is a refactoring, a subsequent ineffectual addition, $InEffAdd(O_k, O_{k+1})$, would indicate a sort of thrashing. Meaning, if the original refactoring was correct, then "refactoring back" is a mistake (and if the "refactoring back" is correct, then the original refactoring is a mistake).

Definition 4. *Given two ontologies, $\mathcal{O}_i, \mathcal{O}_j$ where $i{<}j$, then any of the following sets of changes for α*

F1SSoC. $\{EffAdd(O_i, O_{i+1}) \cap InEffRem(O_j, O_{j+1})\}$
F2SSoC. $\{InEffAdd(O_i, O_{i+1}) \cap InEffRem(O_j, O_{j+1})\}$
F3SSoC. $\{InEffRem(O_i, O_{i+1}) \cap InEffAdd(O_j, O_{j+1})\}$

are fault **suggesting** set of changes *written as* FSSoC(i, j).

There is a large gap in the strength of the suggestiveness between sets of kind F1SSoC and the sets of kind F2SSoC and F3SSoC. Sets of kind F1SSoC can be completely benign, indicating only that additional information has been added to the axiom (e.g., that the axiom was strengthened), whereas there is no sensible scenario for the occurrence of sets of kind F2SSoC and F3SSoC. In all cases, much depends on whether the ineffectuality of the change is known to the ontology modeller. For instance, if a set of type F1SSoC(i, j) was an attempt to repair α, then α is a logical bug if α is an undesired entailment that was meant to have been repaired in O_j, then this repair failed.

All these suggestive sets may be embedded in larger sets. Consider the set where α is *(1)* $EffAdd(O_i, O_{i+1})$, *(2)* $InEffRem(O_j, O_{j+1})$, *(3)* $InEffAdd(O_k, O_{k+1})$, *(4)* $EffRem(O_l, O_{l+1})$. From this we have an indicative fault in the set <(1),(4)> and two suggestive faults in the sets, <(1),(2)> and <(2),(3)>. The latter two seem to be subsumed by the encompassing former. The analysis presented here does not, at this time, cover all paired possibilities. This is partly due to the fact that some are impossible on their own (e.g., two additions or two removals in a row) and partly due to the fact that some are subsumed by others.

Of course, as we noted, all these observations only hold if the requirements have been stable over the examined period. If requirements fluctuate over a set of changes, then the changes might just track the requirements and the ontology might never be in a pathological state.

4 Methods and Materials

The verification of the concepts and definitions proposed in Section 3 is carried out by conducting a detailed analysis of The National Cancer Institute Thesaurus (NCIt) ontology. The National Cancer Institute (NCI) is a U.S. government funded organisation for the research of causes, treatment, and prevention of cancer [4]. The NCIt is an ontology written in the Web Ontology Language (OWL) which supports the development and maintenance of a controlled vocabulary about cancer research. Reports on the collaboration process between the NCIt and its contributors have been published in 2005 and 2009 (see [5,6,7]), which provide a view of the procedural practices adopted to support domain experts and users in the introduction of new concepts into the ontology. These publications together with the publicly available monthly releases and concept change logs are the basis for the corpus used in this study.

We gathered 105 versions of the NCIt (release 02.00 (October 2003) through to 12.08d (August 2012)) from the public website.[2] Two versions are unparseable using the OWL API [8], and were discarded, leaving 103 versions. The ontologies were parsed and individual axioms and terms were extracted and inserted into a MySQL v5.1.63 database. The database stores the following data for each NCIt release, $\mathcal{O}_i$ (where i is the version identifier):

1. Ontology $\mathcal{O}_i$: Each ontology's NCI identifier $\mathcal{O}_i$ is stored in a table "Ontology" with a generated integer identifier i.
2. Axioms $\alpha_j \in \mathcal{O}_i$: Each structurally distinct axiom α_j is stored in an "Axioms" table with identifier j, and a tuple (j, i) is stored in a table "Is In" (that is, axiom j is asserted in ontology i).
3. Classes $C_j \in \mathcal{O}_i$: Each class name C_j is stored in a table "Classes" with an identifier j, followed by the tuple (j, i) into table "Class In".
4. Usage of class C_j in $\mathcal{O}_i$: Each class C_j that is used (mentioned) in axiom $\alpha_k \in \mathcal{O}_i$ is stored in table "Used In" as a triple (j,k,i).
5. Effectual changes: Each added (removed) axiom $\alpha_j \in EffAdd(O_i, O_{i+1})$ ($\alpha_j \in EffRem(O_i, O_{i+1})$), with identifier j, is stored in table "Effectual Additions" ("Effectual Removals") as a tuple (j, $i+1$).
6. Ineffectual changes: Each added (removed) axiom $\alpha_j \in InEffAdd(O_i, O_{i+1})$ ($\alpha_j \in InEffRem(O_i, O_{i+1})$), with identifier j, is stored in table "Ineffectual Additions" ("Ineffectual Removals") as a tuple (j, i).

The data and SQL queries to produced this study are available online.[3]

All subsequent analysis are performed by means of SQL queries against this database to determine suitable test areas and fault detection analysis. For test area identification, we select test sets based on the outcome of 1) Frequency Distribution Analysis of the set of asserted axioms (i.e., in how many versions each axiom appears or follows), and 2) asserted axioms Consecutivity Analysis (whether an axiom's occurrence pattern has "gaps"). For fault detection, we conduct SQL driven data count analysis between the selected test cases and the Effectual and Ineffectual database tables to categorise logical bugs as FiSoCs or FSSoCs.

[2] `ftp://ftp1.nci.nih.gov/pub/cacore/EVS/NCI_Thesaurus/archive/`.
[3] `http://owl.cs.manchester.ac.uk/research/topics/ncit/regression-analysis/`

5 Results

5.1 Test Areas Selection

The test area selection for this study is determined by conducting analyses on axioms' frequency distribution and consecutivity evaluation. Frequency distribution analysis calculates the number of versions an axiom is present in the NCIt. From this sequence analysis we identify their consecutivity based on the type of occurrence in the corpus, such as: continual occurrence, interrupted occurrence, and single occurrence. The analysis of axioms with continual occurrence provides knowledge about the stability of the ontology, since it helps with the identification of axioms that, due to their consistent presence throughout the ontology's versions, can be associated with the 'core' of the represented knowledge. As described in Section 3, axioms' presence can be successfully correlated with FiSoCs or FSSoCs depending on the effectuality of their changes.

In the analysis, we found that the highest number of 20,520 asserted axioms correspond to frequency 11. This means that 20,520 axioms appear in the NCIt ontology for exactly 11 versions. Of these asserted axioms, 20,453 asserted axioms (99.67%), appear in 11 consecutive versions. The distribution of these axioms across the corpus is concentrated between version 6 to 16 with 19,384 asserted axioms (the majority of these additions took place in version 6 with 13,715 added axioms), between versions 1 to 52 with 593 asserted axioms, and 187 asserted axioms for the remaining versions. These numbers do not account for the 358 new asserted axioms added in version 93 that are still in the corpus for version 103 with 11 occurrences but have the potential of remaining in the corpus in the future versions.

The next highest frequency is 5 with 14,586 asserted axioms and 14,585 occurring consecutively. Only the axiom `Extravasation` $\sqsubseteq$ `BiologicalProcess` is present from version 20 to 23, it is removed in version 24 and re-enters in version 45 before being removed in version 46.

The next two rows in Table 1 show the results for frequency distribution 2 and 3 with 13,680 and 12,806 asserted axioms respectively. For frequency distribution 2, there are 10,506 asserted axioms with consecutive occurrence. Of these axioms, 445 entered the corpus in version 102 and remain in the corpus until version 103. The total number of axioms with non-consecutive occurrences is 3,174 asserted axioms. However, only 8 axioms are not included in the set of axioms that are part of the modification event taking place between versions 93 and 94. In this event 3,166 axioms with non-consecutive occurrences were added in version 93, removed (or possibly refactored) in version 94, and re-entered the ontology in version 103. This editing event is discussed in Section 6. Of the 12,806 asserted axioms with frequency distribution 3, 12,804 asserted axioms occur in consecutive versions (99.98%) and 644 asserted axioms are present in the last studied version of the corpus.

Our results show that three high frequency distributions are observed in the top ten distributions with axioms occurring in 87, 79 and 103 versions. There are 12,689 asserted axioms present in 87 versions with 99.86% of asserted axioms occurring consecutively. From these axioms, 12,669 asserted axioms appear in the last version of the ontology with 12,651 asserted axioms added in version 17 and remaining consecutively until version 103. For frequency distribution 79, there exist 10,910 asserted axioms

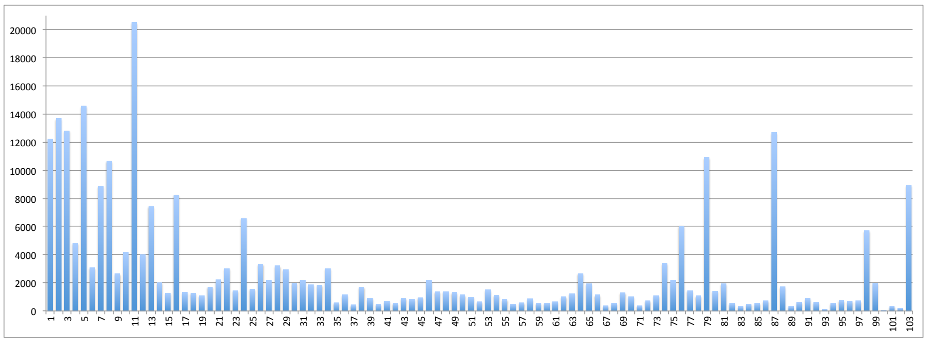

Fig. 1. Distribution of asserted axioms based on the number of versions they are present in (x-axis: frequency, y-axis: number of asserted axioms).

that appear in 79 versions with 10,866 still present in version 103. From these 10,866 asserted axioms, 10,861 asserted axioms were added in version 25 and remain until version 103 consecutively. Finally, there are 8,933 asserted axioms that appear in 103 versions of the NCIt. This means that 8,933 axioms were added in the first studied version of the NCIt and remain until the last studied version. That is, of the 132,784 asserted axioms present in version 103, 6.73% of the axioms were present from version 1. From this information it can be inferred that 6.73% of the asserted axioms population found in the last version of the NCIt represent a stable backbone of asserted axioms present in all versions of the NCIt.

Frequency	Axiom Count	Occurring in Version 103	Consecutive Occurrence	Non-consecutive Occurrence
11	20,520	358	99.67%	0.33%
5	14,586	831	99.99%	0.01%
2	13,680	445	76.80%	23.20%
3	12,806	664	99.98%	0.02%
87	12,689	12,669	99.86%	0.14%
1	12,219	47 in v102 and 2,084 in v103	–	–
79	10,910	10,866	99.93%	0.07%
8	10,662	599	99.93%	0.07%
103	8,933	8,933	100.00%	0.00%

Table 1. Frequency distribution trends.

As seen in Table 1, 12,219 asserted axioms occur in only 1 version of the NCIt. Of these asserted axioms, 2,084 axioms appear in version 103 and may remain in future versions. When taking this fact into account, we observe that a total of 10,135 asserted axioms with single occurrences are present in the remaining 102 versions. From the 103 studied versions, 98 versions have asserted axioms that only appear in those versions; and versions 45, 54, 88, 92, and 100 do not. A detailed representation of this distribution across the NCIt corpus demonstrates that the first three years of the studied NCIt versions show the highest rate of single occurrences in the corpus with three identifiable high periods of single occurrences around version 1 to 5, versions 16 to 18, and versions 21 to 25.

5.2 Fault Detection Analysis

In this study, we limit Fault Detection Analysis to the finite set of asserted axioms with non-consecutive occurrence for the top ten frequencies identified in the previous section. It is important to note at this point that this study does not examine the set of all *FiSoC* and *FSSoC* for the collected versions of NCIt. Instead we focus our attention on the identified 53 asserted axioms that occur in non-consecutive versions for the top ten distributions, excluding all axioms that were part of the renaming events identified between versions 91 to 103 of the NCIt. Of these 53 examined axioms, 32 asserted axioms have logical bugs of type *FiSoC*. Further examination of the change sets of these *FiSoCs* indicate that 27 axioms conform directly with Definition 3 because all of their additions and removals are effectual; that is, the set of changes is $(EffAdd(O_i, O_{i+1}) \cap EffRem(O_j, O_{j+1}))$. The remaining 5 axioms have change sets of type $(EffAdd(O_i, O_{i+1}) \cap InEffRem(O_j, O_{j+1}) \cap EffRem(O_k, O_{k+1}))$. Although in this set there is an ineffectual removal prior to the effectual removal, from this change set we may conclude that ineffectual removal is “fixed” when the effectual removal takes place.

We also identified the asserted axiom `Benign_Peritoneal_Neoplasm` $\sqsubseteq$ `Disease_Has_Primary_Anatomic_Site only Peritoneum` with *axiom_id* 159025 as a logical bug of type *FiSoC* for the first removal $(EffAdd(O_{20}, O_{21}) \cap EffRem(O_{21}, O_{22}))$, and a second logical bug type *FSSoC* for the second removal $(EffAdd(O_{26}, O_{27}) \cap InEffRem(O_{28}, O_{29}))$. The presence of both logical bugs, *FiSoC* and *FSSoC*, in this axiom suggests that the re-introduction of the axiom to the ontology in version 27 after being removed in version 22 may correspond to content regression, and the second ineffectual removal in version 29 to refactoring.

The remaining 21 asserted axioms have logical bugs of type *FSSoC* . Seventeen of these axioms conform with *F1SSoC* set, thus suggesting possible refactoring. To confirm these refactoring events, additional axiomatic difference analysis needs to be carried out on these axioms, as suggested in [9]. Four axioms (*axiom_ids* 110594, 153578, 157661, and 127241) have the change sets identified for *F2SSoC*. Two of these axioms (*axiom_ids* 157661, and 127241) suggest refactoring for the first change set (the set is of type *F1SSoC*), and are later re-introduced in the ontology with logical bugs of type *FiSoC*.

As mentioned earlier, the analysis conducted in this section excludes fault detection for the set of axioms affected by the renaming event that took place between versions 91 and 103. We provide more information about this renaming event and the impact to our results in Section 6. However, it is important to mentioned that our analysis is sensitive to cosmetic changes to axioms, e.g., axiom renaming, and does not treat them as logical bugs due to the superfluous nature of these changes.

Frequency Rate	Axiom ID	Versions for <Eff. Add., Eff. Re.>	Versions for <Eff. Add.>	Versions for <Eff. Add., Ineff. Re., Eff. Re.>	Versions for <Ineff. Add.>	First NCIt Version	Last NCIt Version
11	57506	<4,5>, <7,17>				4	16
	58364	<4,5>, <7,17>				4	16
	103206			<7,17,26>		7	25
	105069			<7,17,26>		7	25
	210295	<40,47>, <51,55>				40	54
2	49544	<2,3>, <4,5>				2	4
	50602	<2,3>, <4,5>				2	4
	50858	<2,3>, <18,19>				2	18
	120551	<12,13>, <16,17>				12	16
	172613	<25,26>, <62,63>				25	62
	172917	<25,26>, <62,63>				25	62
3	159025	<21,22>				21	28
	257839	<83,84>, <93,94>	<103>			83	103
87	30433	<1,12>, <14,75>	<89>			1	103
	39267	<1,2>	<18>			1	103
	68617	<5,6>	<18>			1	103
	118516	<12,74>	<79>			12	103
	119326	<12,74>	<79>			12	103
	121919	<13,47>	<51>			13	103
	122832	<13,47>	<51>			13	103
79	6838			<1,17,86>	<23>	1	85
	8905	<1,6>	<30>			1	103
	44135			<1,17,86>	<23>	1	85
	125718	<15,19>	<29>			15	103
	125895	<15,19>	<29>			15	103
	162303	<23,93>, <94,103>				23	103
	162304	<23,34>	<34>			23	103
8	22465			<1,2,52>	<45>	1	51
	67505	<5,6>, <10, 17>				5	16
	238416	<72,79>	<103>			72	103
	238488	<72,79>	<103>			72	103
	262226	<87,93>, <94,96>				87	95

Table 2. Indicating fault in sequence of changes (Effectual Addition abbrv. to "Eff. Add.", Effectual Removal abbrv. to "Eff. Re.", Ineffectual Addition abbrv. to "Ineff. Add.", and Ineffectual Removal abbrv. to "Ineff. Re.").

6 Discussion

In general, the historical analysis of the NCIt, as recorded in their monthly releases from 2003 to 2012, show that the ontology is consistently active and the evolution management process in place for NCIt 's maintenance (as described in [10] and [6]) may be positive contributors to the overall steady growth of the NCIt ontology.

The growth of the ontology is mostly driven by the asserted ontology where high levels of editing activity took place in the first three years of the analysed population. The change dynamics observed in this period suggest a trial and error phase, where editing and modelling activities are taking place until reaching a level of stability, possibly related to reaching maturity, for the remainder of the observed versions.

Although the chronological analysis primarily points to the first three years as a phase of rapid change, a more in-depth study of the diachronic data set revealed that content regression takes place throughout all versions of the NCIt. A detailed study of the 'life of axioms' in the ontology from the Frequency Distribution Analysis shows that the evolution of the NCIt is marked by logical bugs of either *FiSoC* and/or *FSSoC* types.

Frequency Rate	Axiom ID	Versions for <Eff. Add., Ineff. Re.>	Versions for <Ineff. Add., Ineff. Re.>	Versions for <Ineff. Add.>	Versions for <Ineff. Add., Eff. Re.>	First NCIt Version	Last NCIt Version	Refactoring
11	110594		<10, 20>, <31, 32>			10	31	
	215592	<50, 55>		<98>		50	103	Refactoring
	215897	<50, 55>		<98>		50	103	Refactoring
5	157661	<20, 24>	<45, 46>			20	45	
2	99659	<6, 7>			<16, 17>	6	16	Refactoring
	127241	<16, 17>	<21, 22>			16	21	
3	159025	<27, 29>				21	28	Refactoring
87	3241	<1, 7>		<23>		1	103	Refactoring
	12085	<1, 17>		<33>		1	103	Refactoring
	106537	<9, 17>		<25>		9	103	Refactoring
	106569	<9, 17>		<25>		9	103	Refactoring
	106878	<9, 17>		<25>		9	103	Refactoring
	107407	<9, 17>		<25>		9	103	Refactoring
	107860	<9, 17>		<25>		9	103	Refactoring
	107952	<9, 17>		<25>		9	103	Refactoring
	108468	<9, 17>		<25>		9	103	Refactoring
	111380	<10, 17>		<24>		10	103	Refactoring
	114579	<10, 17>		<24>		10	103	Refactoring
79	42533	<1, 17>		<41>		1	103	Refactoring
8	153578		<17, 18>, <20, 27>			17	26	
	215709	<50, 53>		<99>		50	103	Refactoring

Table 3. Suggesting fault in sequence of changes (Effectual Addition abbrv. to "Eff. Add.", Effectual Removal abbrv. to "Eff. Re.", Ineffectual Addition abbrv. to "Ineff. Add.", and Ineffectual Removal abbrv. to "Ineff. Re.").

As a result, we found that asserted axioms with logical bugs enter the ontology in a version, are removed in a different version, and later re-entered the ontology unchanged. Only 6.73% of the asserted axioms in version 103 correspond to axioms that have been present unchanged from the first version analysed until this last version.

Our study revealed that most asserted axioms appear in two versions of the ontology. However, in this finding we identified 125,294 axioms are affected by the renaming event that took place between versions 93 and 94. In a preliminary study conducted for this paper, we found these asserted axioms first appear in version 93, are removed in version 94, and then re-enter the NCIt unchanged in version 103 . We have confirmed with the NCI that this editing event corresponds to the renaming of terms that took place in version 93, where every term name was replaced from its natural language name to its NCIt code. This renaming event also affects the set of asserted axioms with frequency distribution 11. The non-consecutive version occurrences for 1,186 axioms show that they first occur consecutively from versions 91 and 92, are removed in version 93, and then re-enter the ontology in version 94. These axioms remain consecutively until version 102 before they are removed again in version 103. The identification of this renaming event does not affect the information content dynamics of the ontology; however, it does affect the overall change dynamics. This renaming event is important to our analysis because it shows major editing periods are still part of the NCIt.

Taking into account these renaming events, the study found that the NCIt overall 'survival' rate for asserted axioms is 5 versions. Axioms with non-consecutive presence in the ontology are directly linked to logical bugs that either indicate content regressions or suggest axiom refactoring. Information content is not as permanent as the managerial and maintenance processes indicate, but logical bugs for unmodified axioms are more predominant than expected. The analysis conducted in this paper identifies specific sets of axioms that are part of this group of regression cycles, and it is able to provide in detail the type of faulty editing patterns for these axioms and the location of these errors. We argue that the identification axioms with re-occurring logical bugs is a crucial step towards the identification of test cases and test areas that can be used systematically in Ontology Regression Testing.

7 Limitations

This study has taken under consideration the following limitations: *(i)* The NCIt evolution analysis and asserted axiom dynamics correspond to the publicly available OWL versions of the NCIt from release 02.00 (October 2003) to 12.08d (August 2012). Historical records of NCIt prior to OWL are not taken into consideration in this study. *(ii)* The presented results and analysis is limited in scope to the set of asserted axioms only. The inclusion of entailment analysis is only conducted in regards to the computation of logical differences to categorise the asserted axioms ' regression events into logical bugs of types *FiSoC* or *FSSoC*. *(iii)* Test area selection for the set of axioms with presence in non-consecutive versions is derived by selecting all axioms with non-consecutive presence based on their ranking in the high frequency analysis for all asserted axioms. The selected test area should be viewed as a snapshot of the whole population of axioms with non-consecutive presence, since the set of 53 analysed axioms correspond only to the top 10 high frequency distribution as described in Section 5.1. Analysis of the whole corpus is planned for future research. *(iv)* This study primarily corresponds to Functional Requirement Test Impact Analysis since it deals directly with the ontology. Non-functional Requirements are linked to entailment analysis such as subsumption hierarchy study, which is excluded in this work.

8 Conclusion

Large collaborative ontologies such as the NCIt need robust change analysis in conjunction with maintenance processes in order to continue to effectively support the ontology. The work presented in this paper shows that a detailed study of axioms with logical bugs need to be part of ontology evaluation and evolution analysis techniques due to its significant contribution to regression testing in ontologies. Although the study presented here is limited in that it is only evaluating unchanged asserted axioms, it still shows that a great portion of the editing efforts taking place in the NCIt is in the unmodified content. Regression analysis of this unmodified content can target specific changes in the modelling and representation approaches which can potential safe effort and increase productivity in the maintenance of the ontology.

Regression testing in Ontology Engineering is still a growing area of research, and the work presented here shows that a step towards achieving regression analysis in ontologies is by providing quantitative measurements of axiom change dynamics, identification of logical bugs, and the study of ontology evolutionary trends, all of which can be extracted efficiently by looking at versions of an ontology.

References

1. Gonçalves, R.S., Parsia, B., Sattler, U.: Analysing the evolution of the NCI thesaurus. In: Proc. of CBMS-11. (2011)
2. Gonçalves, R.S., Parsia, B., Sattler, U.: Analysing multiple versions of an ontology: A study of the NCI Thesaurus. In: Proc. of DL-11. (2011)
3. Cuenca Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P.F., Sattler, U.: OWL 2: The next step for OWL. J. of Web Semantics (2008)
4. de Coronado, S., Haber, M.W., Sioutos, N., Tuttle, M.S., Wright, L.W.: NCI Thesaurus: Using science-based terminology to integrate cancer research results. Studies in Health Technology and Informatics **107**(1) (2004)
5. Hartel, F.W., de Coronado, S., Dionne, R., Fragoso, G., Golbeck, J.: Modeling a description logic vocabulary for cancer research. J. of Biomedical Informatics **38**(2) (2005) 114–129
6. Thomas, N.: NCI Thesaurus - Apelon TDE Editing Procedures and Style Guide. National Cancer Institute. (2007)
7. Noy, N.F., de Coronado, S., Solbrig, H., Fragoso, G., Hartel, F.W., Musen, M.A.: Representing the NCI Thesaurus in OWL: Modeling tools help modeling languages. Applied Ontology **3**(3) (2008) 173–190
8. Horridge, M., Bechhofer, S.: The OWL API: A Java API for working with OWL 2 ontologies. In: Proc. of OWLED-09. (2009)
9. Gonçalves, R.S., Parsia, B., Sattler, U.: Categorising logical differences between OWL ontologies. In: Proc. of CIKM-11. (2011)
10. de Coronado, S., Wright, L.W., Fragoso, G., Haber, M.W., Hahn-Dantona, E.A., Hartel, F.W., Quan, S.L., Safran, T., Thomas, N., Whiteman, L.: The NCI Thesaurus quality assurance life cycle. Journal of Biomedical Informatics **42**(3) (2009)