Vivisection of a non-executable, domain-specific language
— Understanding (the usage of) the P3P language —

Ralf Lämmel and Ekaterina Pek
Software Languages Team & ADAPT Lab
Universität Koblenz-Landau, Germany

Abstract

P3P is the policy language with which websites declare the intended use of data that is collected about users of the site. We have systematically collected P3P-based privacy policies from websites listed in the Google directory, and analysed the resulting corpus with regard to different levels of validity, size or complexity metrics, different cloning levels, coverage of language constructs, and the use of the language's extension mechanism. In this manner, we have found interesting characteristics of P3P in the wild. For instance, cloning is exceptionally common in this domain, and encountered language extensions exceed the base language in terms of grammar complexity. Overall, this effort helps understanding the de-facto usage of the non-executable, domain-specific language P3P. Some elements of our methodology may be useful for other software languages as well.

I. Introduction

The present paper is a vivisection of W3C's P3P language [1], [2] informed by methods from the fields of program comprehension, reverse engineering, programming languages, and empirical software engineering.

P3P is a non-executable, domain-specific, XML-based language for privacy policies to be declared by the websites for the benefit of web-site users. "*The Platform for Privacy Preferences Project (P3P) enables Web sites to express their privacy practices in a standard format that can be retrieved automatically and interpreted easily by user agents. P3P user agents will allow users to be informed of site practices (in both machine- and human-readable formats) and to automate decision-making.*" [1].

The P3P language has a trivial syntax that is based on just a few concepts (purpose, recipient, retention, and *data*) whose domains are essentially enumeration types. That is, P3P involves vocabulary for data references (such as user data or browsing-related dynamic data), a retention level (to control storage of data, e.g., 'no retention' or 'indefinite retention'), recipients (such as 'ours' or 'public'), purposes (such as administration or tele-marketing). We refer to Fig. 1–Fig. 2 for two typical P3P policies.

```
<POLICY>
  ...
  <STATEMENT>
    <NON-IDENTIFIABLE/>
  </STATEMENT>
  ...
</POLICY>
```

Fig. 1. P3P sample 'private site' (The policy, with its special NON-IDENTIFIABLE element, signifies that the website does not collect any data, or that it does collect data, but anonymizes it upon collection.)

```
<POLICY>
  ...
  <STATEMENT>
    <PURPOSE><admin/><current/><develop/></PURPOSE>
    <RECIPIENT><ours/></RECIPIENT>
    <RETENTION><indefinitely/></RETENTION>
    <DATA-GROUP>
      <DATA ref="#dynamic.clickstream"/>
      <DATA ref="#dynamic.http"/>
    </DATA-GROUP>
  </STATEMENT>
  ...
</POLICY>
```

Fig. 2. P3P sample 'logging only' (The policy, with the two specific data references in the `#dynamic` branch, signifies that the website collects navigation and click-stream data as well as computer information. The purposes `admin`, `current`, and `develop` mean that the data is collected for the website's benefit—the user will not be contacted or affected in any way. The only recipient of the data is '`ours`'—the website. The data may be stored `indefinitely`.)

The present paper reports on an *empirical language analysis* of P3P on the grounds of a corpus obtained from the Internet. We realized the need for such empirical work as we were looking into methods of checking

policy compliance for information systems with P3P as an important policy language in the wild. Our analysis is concerned with a broad scale of validity, different kinds of size or complexity metrics, different levels of cloning, coverage of the language's vocabulary, and the use of P3P's language-extension mechanism. In contrast, previous P3P studies [3], [4], [5], [6] have focused on P3P adoption (e.g., geographically), schema-based validity, and compliance with legislation for P3P.

We hope our research to be of value for researchers and practitioners deeply interested in privacy policies. However, we also submit this work to the broader software language engineering community. After all, P3P is yet another domain-specific language. We have learned that a multi-dimensional exploration of language usage is an effective and organized way of understanding a language in ways that are not achievable through classical sources such as the language documentation.

Contributions

1. We study *syntactically and semantically bounded metrics for P3P policies*. This effort helps understanding the distance between the syntactical level of P3P (which is familiar to the typical P3P language user) versus the more fine-grained semantical meaning.

2. We study *cloning for P3P* at the textual, syntactical, and semantical level. This effort helps understanding the authoring habits for P3P and makes us realize that there are common policies which are widely used.

3. We study important properties of *P3P language usage*, e.g., adherence to constraints, coverage of the policy language or its data schema, and the use of P3P's extension mechanism.

4. Our work is based on a substantial *P3P corpus* which we obtained by systematically scraping URLs for candidate sites from Google directory. We are making available the corpus and all retrieved data at the paper's website.[1]

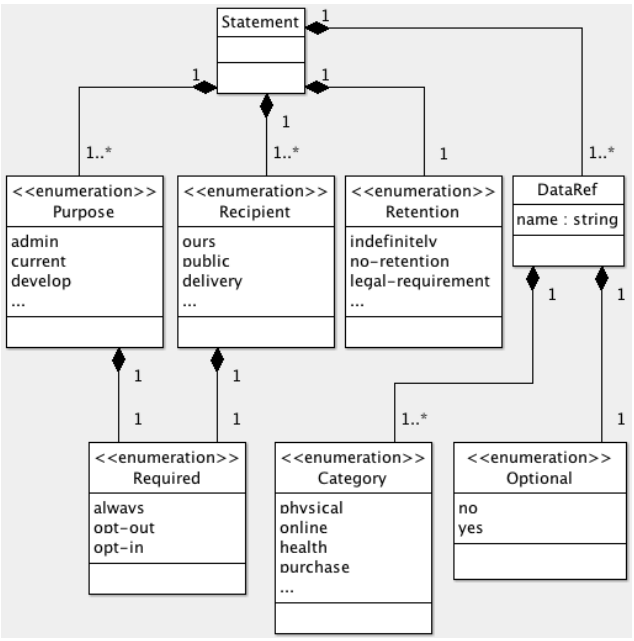
Road-map

- §II briefly recalls P3P's *syntax*.
- §III sets up a suitable *semantics* for P3P.
- §IV describes the retrieval of the P3P *corpus*.
- §V assesses the validity of the P3P *corpus*.
- §VI presents our take on P3P *metrics*.
- §VII presents our take on P3P *cloning*.
- §VIII is concerned with other aspects of *language usage*.
- §IX discusses related work.
- §X concludes the paper.

II. Language syntax

A policy consists of any number of *statements* each of which identifies one or more *purposes* of collecting data, one or more *recipients* for the data, one *retention* level (say, a 'persistence' level), one or more *data references*; see the introductory examples. Fig. 3 describe the simplified, abstract P3P syntax in UML notation. We only cover statements with identifiable data (as opposed to the special

[1] http://userpages.uni-koblenz.de/~laemmel/p3p



More comprehensive syntax definitions:
- XSD: http://www.w3.org/2002/01/P3Pv1.xsd
- Relax NG: http://yupotan.sppd.ne.jp/relax-ng/p3pv1.rng
- RDF: http://www.w3.org/TR/p3p-rdfschema/

Fig. 3. Simplified, abstract syntax of P3P

form of Fig. 1). We elide P3P elements that are irrelevant with regard to the logical meaning of policies. In particular, we have elided the 'entity' that issues the policy, and a policy's 'consequences' that are given in natural language.

Purposes and recipients may be qualified by an attribute *required* with the following possible values: 'always'—the purpose or recipient cannot be deselected by the user of the website; 'opt-in' versus 'opt-out'—selection or deselection may be expressed. Data references may be qualified by an attribute *optional*; the values are 'yes' versus 'no'—the user of the web site may or may not omit the data, respectively. Data references are also implicitly or explicitly qualified with so-called *categories* such as 'physical' (i.e., physical contact information).

III. Language semantics

A software language engineer's approach to language understanding would typically involve a language semantics. P3P's semantics—in fact, its *nature*—is non-obvious. The P3P standard does not directly define anything of a semantics by programming language standards. Of course, P3P is a non-executable language. Hence, we have to be careful with our expectations.

A. Comparison-based semantics

The P3P architecture suggests a semantics—implicitly: *the comparison of a website's policy with user preferences for privacy*. That is, the meaning of a policy would need

- r-purpose(data,purpose,required)
- r-recipient(data,recipient,required)
- r-retention(data,retention)
- r-data(data,optional)
- r-category(data,category)

Fig. 4. Data-centric semantics of P3P [11]

to be understood relative to queries (predicates) over the policy. In practice, user agents carry out 'comparison' when a website is visited—before any data is submitted to the site; see [7] for W3C's reference implementation.

The P3P standard does not insist on any particular style of user preferences and queries, but it recommends *APPEL* [8] (another W3C language). APPEL's definition has been criticized widely [7], [9], [10], [11]. For instance, comparison is not stable under 'obviously' meaning-preserving variation of policies. Alternative languages for preferences have been proposed [9], [10], but ultimately P3P is not fully standardized in this respect.

B. Yu et al.'s data-centric semantics

Our empirical analysis of P3P usage does not involve any sort of comparison-based semantics. However, we realized that we could benefit from a normal form for P3P policies. The issue is that P3P's statement form allows one to group purposes, recipients, retentions and data references in different, semantically equivalent ways. When we examine complexity of policies or cloning, it would be favorable to use a normal form so that semantically equivalent variations are not distinguished. Yu et al.'s proposal for a relational semantics for P3P [11] provides such a normal form, indeed. There are different relations to describe precisely—in a point-wise manner—what purposes, recipients, retention levels and data references are associated by a policy; c.f., Fig. 4 for the relational schema. There are also appropriate key constraints; see the (underlined) primary keys in the figure.

The relations clarify that P3P may be interpreted in a 'data centric' manner such that a policy describes different 'properties' of data references to be collected: purposes, recipients, and retention levels.

IV. Language corpus

We commit to a single source for website URLs: the Google directory.[2] This source is well respected in terms of its coverage. It is important to us that the retrieval of the corpus should be easily reproducible by others (and ourselves). Thus, the corpus is made publicly available by us on the paper's website.[3]

[2]http://www.google.com/dirhp?hl=en

[3]Our efforts of scraping, crawling, and downloading were performed over December 2009-January 2010.

A. URL scraping from the Google directory

The Google directory contains these top level categories: *Arts, Business, Computers, Games, Health, Home, News, Recreation, Reference, Science, Shopping, Society, Sports, and Regional*. Our assumption is that sites reachable through (sub)categories of the directory cover a broad range of companies, consortia, and other entities with a web presence that may be reasonably expected to declare a privacy policy.

Google's categories may be structured through multiple levels of subcategories. We applied a depth-first traversal over the categories. On the page of a given category (or subcategory), relative 'href' attributes are interpreted as pointers to subcategories, and the non-relative ones correspond to the URLs of sites to be searched for policies.[4]

# URLs	1472090
# Distinct URLs	1450660
# Domains	991948

TABLE I. Results of URL scraping

The slightly lower number of *distinct* URLs shows that some URLs occur multiple times in the directory. We list the number of distinct domains as an indicator of diversity.

B. Policy scraping per website

The mechanism of finding out whether a site uses P3P and locating the actual policy is defined in the P3P specification [1]. There are two steps. Firstly, it must be checked whether a *policy reference file* exists; if so, secondly, the referenced policies must be fetched.

The policy reference file is searched as follows. (i) The file may be located in a predefined "well-known" location, which is essentially [WebSiteURL]`/w3c/p3p.xml`. (ii) the website may contain the HTML/XHTML tag `<link>` indicating the location of the file. (iii) an HTTP response from the server may contain the reference.

	URLs	**Domains**
# Located policy reference files	29449	23778
# Referenced policy files	7047	4514
# Downloaded policy files	4575	4565

TABLE II. Raw corpus before any validation

When policies are embedded into reference files, then we count the reference file as a policy file, too. The above numbers show that only few entries of the Google directory have an associated policy reference file. There are even less policies that we could retrieve; this is due to two effects: unresolvable references (i.e., failing downloads) and aliasing (i.e., several policy reference files point to the same policy).

[4]In order to prevent cycles during such traversal, only *direct* relative references were followed. Further, the special category 'Regional' was excluded.

V. Analysis of validity

Starting from the total number of downloaded policies, we gradually increase the level of validity; see the following table.

Level	Criterion	# Files	# Policies	!5
0	All downloads	4575	—	—
1	XML ('no noise')	4158	—	—
2	Well-formed XML	4075	—	—
3	XSD-validated XML	3086	3227	1603
4	Key constraints	2741	2868	1359
5	Additional constraints	2661	2780	1301

TABLE III. # Policies per validity level

First, we exclude all 'noise', i.e., files that obviously do not contain XML markup for a policy. Next, we exclude non-well-formed XML files. Then, we exclude files that do not validate with the P3P XSD, which can be viewed as 'syntax checking'. Then, we exclude files by consistency checking with regard to (the key constraints of) the relational semantics of §III-B. Finally, we impose a few additional constraints, which we discuss below.

While this paper will not address policies at levels 0–2 due to their malformed-ness, it is noted that some P3P tools may tolerate specific malformed-ness problems.

A. Key-constraint violations

These violations are based on the relational semantics of §III-B, which is not endorsed by W3C and may be potentially in conflict with established intuitions. However, in our opinion, the key constraints are correct and useful from a privacy point of view. For instance, a violation of the key constraint for r-retention would mean that a policy makes different retention promises (such as 'no retention' versus 'indefinite retention') for the same data, which is hardly sensible.

Relation	# Policies	!
r-purpose	33	27
r-recipient	4	4
r-retention	98	79
r-data	307	199
any of them	359	244

TABLE IV. Violations of key constraints

The high number of issues with r-data indicates a discrepancy between the user's perception of the language and the likely semantics (in terms of constraints). Our reading of the relevant paragraph in the P3P standard[6] confirms the key constraint for r-data. It appears that many users perceive the attribute *optional* to be restricted to a specific purpose or recipient; they may be misguided by the statement-based grouping in the syntax. This observation may suggest that such fine-grained optionality should be added to the P3P language since users seem to require it. Also, P3P tools should perform stronger validation to enforce the current optionality rules of P3P.

B. Additional constraints

The revised P3P standard mentions a few examples of sanity-checking rules for user agents.[7]. However, we are not aware of any comprehensive discussion of logically consistent use of P3P's vocabulary. As an experiment, we extracted a few constraints from [11], and used them for level 5 in Table III: *'ours mandatory'*—a policy that does not involve any rules for 'ours' as recipient (i.e., the website's entity) is arguably illogical; *'public recipient ⇒ indefinite retention'*—a policy that associates the *public* recipient with a data reference, should accordingly admit indefinite retention for this data reference; *'historical purpose ⇒ some retention'*—a policy that associates the *historical* purpose to a data reference, should accordingly admit at least some level of non-indefinite retention.

Constraint	# Policies	!
'ours ...'	64	42
'public ...'	36	25
'historical ...'	3	3
any of them	80	58

TABLE V. Violations of other constraints

VI. Language metrics

For any software language, the question of complexity or size arises inevitably. What metrics usefully order policies on a scale 'trivial, small, medium, large'? P3P is a simple language, but the provision of metrics is challenged by a difficulty of validation. One could reasonably expect that P3P metrics should reflect effort for development, maintenance or comprehension of the policies and the corresponding information system. We do not have access to any data about effort of any kind.

Hence, we will opt for a safe route here, and only measure *syntactical versus semantical size* of the policies themselves in a straightforward manner. We hypothesize that semantical size, in particular, provides a lower bound for the effort of understanding whether or not a information system is in compliance with a given policy.

A. Syntactical size

The well-known LOC ('lines of code') metric is not applicable to P3P—as it is an XML-based language. Hence, a simple intuition is to define the size of a policy as the recursive node count of its representing XML file.

[5] **"!" is used as an indication of diversity** in some tables. In those columns, we give numbers of *syntactically distinct* policies, thereby providing an additional indication of diversity in the corpus. We study such cloning characteristics in detail in §VII.

[6] http://www.w3.org/TR/P3P11/#DATA

[7] http://www.w3.org/TR/P3P11/#ua_sanity

This intuition can be refined into a representation-ally independent and appropriately narrow definition of the syntactical size of a policy p, denoted as $\mathcal{SYN}(p)$. We count the following: *purposes, recipients, retention levels, data references, required/optional attributes, (explicitly defined) categories, grouping nodes for statements and data groups, nodes for non-identifiable data.* Specifically, we do not count such nodes as the description of the entity and consequences.

B. McCabe and Halstead effort

For imperative and OO programs, there are different metrics to describe the effort needed for understanding the program, notably McCabe's cyclomatic complexity [12] and Halstead effort [13]. Let us briefly discuss these two metrics, and use them for inspiration.

The Halstead effort is defined relative to (numbers of) operators and operands. In an imperative language, operands would be essentially all kinds of abstractions (types, top-level functions, etc.), and operators would be all kinds of statement and expression forms. Abstractions are not really present in P3P, unless we count data references in a certain way. It appears that Halstead effort would provide another syntactical size for P3P.

McCabe complexity is meant to measure the number of linearly independent paths through a flow graph, or it can be interpreted as a measure for the number of decisions in a program where a decision corresponds to a Boolean-valued expression in conditional or iteration statements. Flow and decision nodes are not present in P3P, but instead, one could try counting certain decision nodes in an information system, namely all those decision or verification nodes that serve compliance, specifically. Alas, such counting is not feasible because actual information systems do not reveal the nodes in question.

C. Semantical size

The following metric $\mathcal{SEM}(\cdot)$ for the semantical size of a policy is notably different from syntactical size; it shares the intention of McCabe to count "decision nodes" of some kind. That is, based on the relational semantics of §III-B, we can view a policy as multiple Boolean ('decision') matrices. All the matrices use the universe of data references as the first dimension. The second dimension is either of these universes: purposes, recipients, retention levels, requiredness, and optionality. (Arguably, we may also consider explicitly or implicitly defined categories in the second dimension, but this is not done in the paper.) We can define the semantical size as the number of all the *checked cells* in these matrices. In fact, we do not count cells for requiredness and optionality that use defaults (required="always", optional="no"). Here, we assume that these defaults map to 'no-ops' in the system.

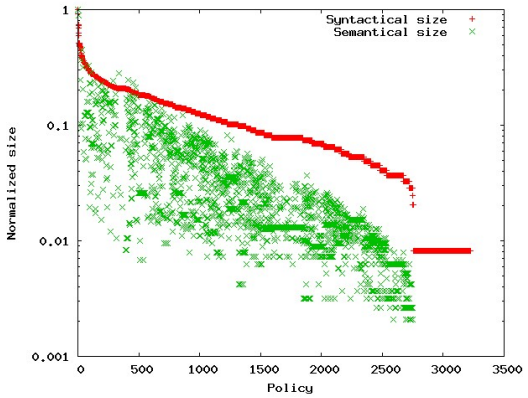


Fig. 5. Syntactical versus semantical size

We use the intuition that understanding whether or not a (well-structured) information system is in compliance with a policy may correlate with the semantical size in so far that the decision matrices would need to be present in some form in the system and would need to be understood. Again, we offer this as an intuition as opposed to a hypothesis that could be reasonably validated.

D. Discussion

The distributions of the two size metrics are shown in Fig. 5. Policies are ordered by syntactical size. Both metrics are normalized to the $[0, 1]$ range. The policies at the tail with low syntactical size have a semantical size of 0—the logarithmic y-axis starts only at 0.001. There are many policies without assigned semantical size because of the violation of key constraints, as we showed in Table III. However, this status is not visible in Fig. 5 because the invalid policies have scattered syntactical sizes.

Size	25th perc.	Median	75th perc.	Max	Avg
Syntactical	11.00	19.00	35.00	245	25.57
Semantical	18.00	25.00	107.00	1907	48.09

TABLE VI. $\mathcal{SYN}(\cdot)$ **and** $\mathcal{SEM}(\cdot)$

The cells for percentiles in Table VI could be used to classify policies as 'trivial', 'small', 'medium', and 'large'. We make two important observations: i) Most policies in our corpus are trivial by all standards. (In particular, half of all policies has a node count of no more than 19.00.) ii) Policies of about the same syntactical size may vary in semantical size by a maximum factor of 29.13. We interpret this variation as to mean that *P3P's syntax masks semantical complexity.*

In an attempt to find other complexity metrics (say, indicators thereof), we realized that the numbers of (distinct) purposes, recipients, and (explicitly stated) categories reveal architectural or legal complexity of the associated website; c.f., Fig. 6. For instance, any declared recipient other than 'ours' reveals another party being involved in

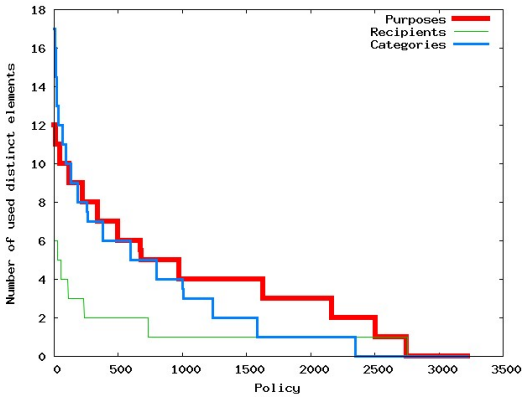


Fig. 6. Distribution of complexity indicators

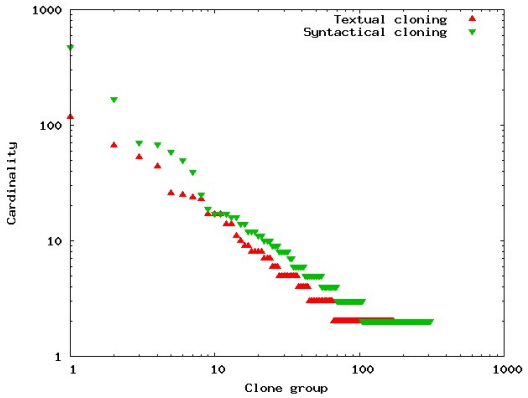


Fig. 7. Distribution of clone groups

the service. Most sites do not involve any such additional recipient.

VII. Cloning

Why would we want to perform clone detection [14], [15], [16] in the context of understanding language usage? Foremost, we were driven by the hypothesis that there may be some common policies that are used by many websites.

A. Levels of clone detection

For P3P, it is straightforward to consider these levels of clone detection: *textual level*: two policies (downloaded from different URL) are textually identical; *syntactical level*: the abstract, syntactical structures (as defined in Fig. 3) are identical; *semantical level*: the relational semantics (in terms of the relations of Fig. 4) are equal.

Level	# Clone groups	Delta # Clone groups	Delta # Cloned pol.	Enlarged clone groups	Merged clone groups	% Clones
Textual	172	172	774	–	–	23.99
Syntactical	309	175	812	29	60	25.16
Semantical	282	14	36	0	7	1.12
Total	–	361	1622	–	–	50.26

TABLE VII. Clone detection by level

Table VII gives a first idea of the cloning situation for our corpus. It is important to remember that clone groups at higher levels assimilate those at lower levels. We make the following observations. i) There are many textual clones, which may be surprising because, one could expect all policies from different sites to differ at least in the 'issuing entity'. ii) We take the large number of syntactical clones as support for our hypothesis that many websites reuse certain proven, generic policies. iii) Semantical clone detection, as it stands, adds very little.

The distribution of textual and syntactical clone groups in terms of cardinality is shown in Fig. 7. We observe that there are about 25 clone groups of each kind with cardinality ≥ 10, but the majority of the clone groups is trivial in the sense that the cardinality is 2.

B. Discussion of textual clones

Some parameters of the 'Top 10' clone groups (in terms of cardinality) are collected in Table VIII. We list the issuing entity, as well as syntactical and semantical size. The latter size is 0 for all encounters of the 'private site' policy; c.f., Fig. 1. (This policy has syntactical size 2 because it consists of a statement node and a non-identifiable node underneath.)

	Entity	Card.	Avg. dist.	$\mathcal{SYN}(\cdot)$	$\mathcal{SEM}(\cdot)$
1	CybrHost	118	1	2	0
2	PhotoBiz	67	1	2	0
3	NASA	53	0	13	29
4	Bravenet	44	0	17	17
5	I-net Broadcast.	26	1	45	49
6	WebRing Inc	25	0.04	25	68
7	Wetpaint	24	0.167	19	25
8	FeedBurner	23	0.957	2	0
9	CNET Networks	17	0.647	76	244
10	BioMed Central	17	0.941	25	36
	Average	–	0.58	22.60	46.80

TABLE VIII. Top 10 textual clone groups

Indication of copy&paste The table also shows the *average URI distance*. This measure is supposed to help with understanding the copy&paste habits in the P3P domain. The URI distance for a single policy is the 'distance' between site URI and the entity URI (the so-called discuri). We set this distance to 0 if both URIs appear to be about the same entity—subject to advanced matching of domain names (such as in the case of www.microsoft.com and server1.microsoft.de), and to 1 otherwise.[8]

Let us inspect a few of the clone groups. No. 1 is the 'private site' policy of a web-site hosting service. An

[8]Consider the following data point. The site p3pedit.com presents a software product of the company codeinfusion.com. The site of the product has a P3P policy embedded into the reference file http://p3pedit.com/w3c/p3p.xml, but the "discuri" attribute points to the site of the company. This is not against the P3P specification (see discuri; http://www.w3.org/TR/P3P11/#POLICY), but shows a good example of cross-domain copy&paste. Incidentally, P3PEdit is a P3P tool!

average URI distance of 1 indicates that *all* clones appear on domains apparently unrelated to the hosting service. We hypothesize that many hosted web sites reuse (perhaps implicitly) a default policy without customizing it for their own domains. Several of the other top-10 are about web hosting, too—with similar URI distances. In contrast, the NASA policy is never cloned outside NASA's web space. We face yet another situation for No. 9. This is the policy with the largest size among the top 10. The policy deals with data collection for website navigation/interaction, cookies, and user registration. The URI distance is relatively high; cursory inspection suggests that we face a corporation with several, de-facto independent sites which however loop back eventually to the corporation for the legal matter of a privacy policy—as far as the issuing entity is concerned.

C. Discussion of syntactical clones

Let us emphasize that textual clones are most likely the result of exact copy&paste. In contrast, additional options arise for syntactical clones due to the performed abstraction: clones could be syntactically equal out of coincidence (in particular, when they are of trivial syntactical size), or policies may have been reused and customized without affecting our abstract syntax.

	Sample entity	Card.	Avg. dist.	$\mathcal{SYN}(\cdot)$	$\mathcal{SEM}(\cdot)$
1	Jaclind Corp.	200	0.095	2	0
2	BestRestaurants	131	0.122	19	25
3	allsafepool.com	68	0.118	9	12
4	Res.Reading	45	0.156	51	–
5	Art of War Central	19	0.105	12	17
6	Bass Centre	14	0	24	15
7	Casino Enterprise	12	0	14	21
8	Johnston Press	9	0.111	20	24
9	Acoustic Concerts	8	0	34	32
10	Benefit News	7	0	68	132
	Average	–	0.07	25.30	27.80

TABLE IX. Top 10 syntactical clone groups

In the 'Top 10' syntactical clone groups in Table IX, we do not count textual clones (so that we compensate for the effect that several syntactical clone groups are extensions of textual clone groups). It turns out that there is one semantically invalid clone group among the 'Top 10': No. 4 has conflicting optionality attributes for one data reference.

No. 1 shows us that there are even more instances of the 'private site' policy in the corpus. If we combine textual and syntactical cloning, then there are 472 occurrences of this policy in the corpus. This is 14.63 % of *all* policies. Incidentally, the policy is not in the suite of online samples of the primary textbook on P3P [2].[9]

No. 2 declares that data related to web-site navigation and interaction may be stored indefinitely, and cookies are optionally stored for some time. We coin the name '*logging and cookies*'. Given the size of the policy, the frequency of its occurrence (say, the cardinality of the group) cannot be explained as a coincidence. Hence, we face a prominent policy for reuse. The low URI distance should mean that the copied&pasted clones are genuine in the sense that different issuing entities are used.

No. 3 is the '*logging only*' policy of Fig. 2. Assuming a privacy ordering on policies, we have that No. 2 $\geq$ No. 3, i.e., No. 2 signifies more collection (storage and sharing) of data than No. 3. That is, No. 2 adds cookie-based tracking to No. 3.

Let us refer to No. 4 as '*web shop*': there are elements to deal with user data, delivery, and data for a purchase. We have that No. 4 $\geq$ No. 2.

[9] http://p3pbook.com/examples.html

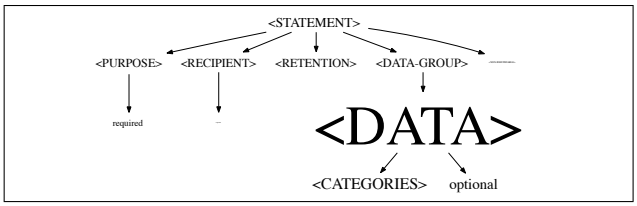


Fig. 8. Frequency of P3P constructs

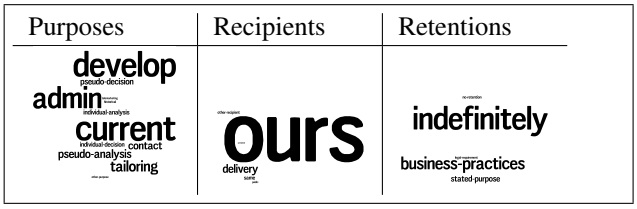


Fig. 9. Tag clouds for P3P constants

VIII. Language usage

First, we will discuss coverage analysis of the P3P language elements and its data model. Then, we will discuss a grammar inference to understand the use of P3P's mechanism for language extension.

A. Coverage analysis for P3P constructs

As a starting point, let us examine the relative frequency of (usage of) the major P3P constructs. For ease of understanding, Fig. 8 shows frequency by means of the *size* of each element while arranging the elements according to the syntax definition of the language in a tree. We observe: i) statements are indeed used to group several data references; ii) there are obviously many data references with associated category elements (clearly more than one category per statement, by average); iii) the default "always" for the attribute "required" is kept mostly, i.e., website users are not often provided with opting in or out.

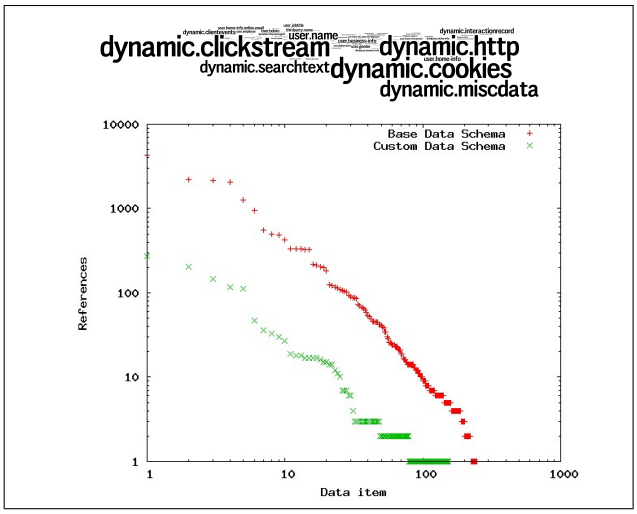


Fig. 10. Usage analysis for P3P data

Fig. 9 shows the frequency of the various values for purposes, recipients and retention. Only a subset is substantially used: 3 (out of 11) for purposes; 1-2 (out of 6) for recipients, namely 'ours', and perhaps 'delivery'; 1-2 (out of 5) for retention levels, namely 'indefinitely', and perhaps 'business-practices'. The dominance of 'ours' was to be expected because each logically consistent policy must involve 'ours'. The dominance of indefinite retention may be explainable through web logs, as our discussion of clone groups illustrated.

B. Coverage analysis for P3P data

Fig. 10 shows the frequency of referencing to all the data items from both the base data schema (BDS) and any custom schema that may get imported. As an indication, BDS has 324 nodes. Overall, 247 (i.e., 76.23 %) of all nodes from the schema are referenced at least once. From the tag cloud in the picture, we can see that 5-6 data references dominate; the first 5 belong to the *dynamic* branch of the schema; the 6th item is *user name*; the other branches (*thirdparty* and *business*) are relatively negligible. The particular *dynamic* data references imply that policies in the corpus declare collection of data such that the highest frequency is about web navigation and interaction, the second highest frequency is about cookies, and the third is *dynamic.miscdata*. The latter is a wild-card for collecting data—subject to the obligatory specification of P3P categories. That is, the wild-card is used in favor of more specific data references.

C. Grammar inference for P3P extensions

P3P readily provides an extension mechanism that allows P3P tool providers and policy authors to add and use extra constructs. For instance, one might want to add a more detailed specification of retention, say in terms of a specific duration.

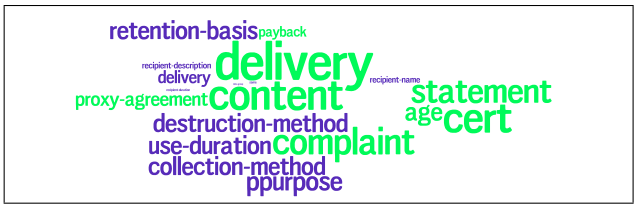


Fig. 11. Tag cloud of extensions w/o group-info

We wanted to learn how this extension mechanism is used. Such insight would be useful in deciding whether P3P is considered sufficient for privacy concerns, or whether any important extensions are used frequently, and hence, may need to be added to the language in the future.

Fig. 11 gives an overview of all top-level extension tags (omitting one dominant tag, as discussed below) that we found in the corpus. Fig. 12 summarizes all extensions of the corpus as XML-element declarations (using EBNF notation for conciseness' sake). These declarations were obtained by a form of XML-schema inference from all extension elements in the corpus.

# extensions	3715
# extension tags	21
# affected policies	1529
# syntactically distinct policies	807

TABLE X. Basic numbers about extension usage

The P3P standard illustrates a few proposals for extensions. It turns out that one of these proposals, 'group-info', accounts for most extension usage in the corpus. The group-info tag lives in an XML namespace by IBM, and it serves for structured comments. There are only 70 syntactically distinct policies that use extensions other than group-info. Most of these extensions 'invade' the standard XML namespace of P3P—as illustrated in Fig. 12.

The complexity of the inferred grammar suggests that *there are more proposed extensions than standardized language elements*. In numbers:

- # distinct extension elements = 81
- # distinct XML elements in P3P statements = 50
- Ratio = 1.62

This situation suggests that P3P may be considered as insufficiently expressive in some cases. There is a natural resistance to using extensions because it challenges the W3C standard-based interpretation of the policy, but our measurements show that there is clearly a need for using extensions.

IX. Related work

A. Empirical program analysis

There is early work on simple static and dynamic program analysis for FORTRAN, Cobol, and Pascal with

statement : (***collection-method*** | ***destruction-method*** | group-info)*
collection-method = (***other-method*** | **delivery** | **document** | **purnish** | **qnaboard** | **subscription** | TELEPHONE)*
destruction-method = (***other-method*** | **format** | **shatter**)*
other-method ≐ ε

purpose : (***ppurpose*** | ***purpose*** | **age** | **cert** | **complaint** | **content** | **payback** | **proxy-agreement** | **statement**)*
ppurpose ≐ (account | browsing | **delivery** | FEEDBACK | **finmgt** | **government** | **login** | **marketing** | **news** | **payment** | **sales** | **session**)*
purpose = (FEEDBACK | **government**)*

recipient : (···)* % Production elided for brevity

retention : (***retention-basis*** | ***use-duration***)*
retention-basis ≐ (***credit-privacy-law*** | ***e-trade-law*** | ***internal-company-regulation*** | ***other-basis***)*
use-duration = (***other-duration*** | **five-year** | **instance** | **one-month** | **one-year** | **six-month** | **three-month** | **three-year**)*
credit-privacy-law = (***other-duration***)*
e-trade-law = (**five-year**)*
internal-company-regulation ≐ ε
other-basis ≐ ε
other-duration ≐ ε

data-group : (···)* % Production elided for brevity

P3P admits extension elements immediately below P3P elements for statements, purposes, recipient, retention levels, and data references (in fact, data groups) as resembled by the different blocks of productions above; see the first rule in each block. Top-level tags of extensions are listed next to the P3P element; extra extension elements, as they occur in nested positions, are listed below the first production in each block. Underlined tags correspond to elements without content. Attributes of extensions are not shown in the grammar. We simply aggregate all encountered extension elements as an iterated choice. (We do not try to locate schemas for the extensions.) When '≐' is used as a rule separator, then text content instead of structured content was encountered for the element in question. All tags that use P3P's XML namespace are shown in bold face. All tags that are used with different XML namespaces are shown in capitals and bold face. All the other tags use a unique XML name space (different from P3P's).

Fig. 12. A fragment of the extension grammar

the objective of informing compiler implementors, and possibly language designers [17], [18], [19], [20]. Static analysis is typically based on simple structural program properties. Dynamic analysis may examine aspects such as depth of recursion.

We also measure structural properties. Dynamic analysis is not directly relevant for P3P. We pick up the extra opportunity of semantics-based measurements. Our work may be of interest for P3P tool providers and language designers, and we provide insights into P3P's use cases.

In [21], the use of APL language features is found to obey an 80-20 rule. We have checked that the use of P3P's base data schema obeys such a rule, too. That is, about 80 % of all P3P policies in the corpus (in fact, 81.54 %, i.e., 2631 policies) use only 20 % of all data references.

In recent years, Java programs (or byte-code programs) have been empirically analysed [22], [23], [24]. The work of [23] makes a laudable effort to deeply study the mathematical distribution of simple structural properties; we have taken a naive approach here.

The idea to empirically analyze policies (as opposed to programs) was also inspired by our earlier work on understanding the usage of XML schemas [25]. In this context, we also had to come up with non-classical forms of metrics and structural properties as well as styles.

B. Studies of P3P adoption

Several studies have analyzed P3P's adoption; we consider the following list as representative of this line of work: [3], [4], [5], [6]. The studies differ in objective, the method of obtaining a corpus, and analytical techniques. Table XI provides a summary.[10]

Property	**[3]**	**[4]**	**[5]**	**[6]**
Time stamp	12/2006?	11/2005	12/2006	11/2006
Size of corpus	3846?	1482?	3846?	3282
Number of sources	11	6	5	1
Syntax error rates	○	○	●	○
Boundary breakdown	–	●	–	●
Website evolution	–	●	●	–
P3P language coverage	–	–	○	–
Privacy classification	●	–	○	–
Legislation	–	–	–	●
HRPP interpretation	–	–	●	–

TABLE XI. Related work on P3P adoption

Legend: 'boundary breakdown' means that the analysis distinguishes countries, continents or language boundaries; 'website evolution' means that the addition of policies to websites, policy changes, and the removal were tracked; 'privacy classification' means that site policies are classified according to perceived privacy profiles; 'legislation' means that available legislation (per jurisdiction) is mapped to privacy preferences, and then compared with site policies; 'HRPP' (Human Readable Privacy Policy) means that natural-language-based policies are compared with P3P machine-readable parts (requiring human effort to this end).

In 3 out of 4 cases, these earlier studies involved multiple sources such as ranking lists for websites, and they also used search engines, e.g., by locating websites through ranked lists of search terms. Our reliance on a single source is a threat to validity, but given the size of

[10]Question marks in the cells mean that we may have misinterpreted corresponding data; '●' means that the study focuses on this issue; '○' means that the study touches upon the issue.

the corpus and complete leverage of the Google directory, we should have covered many important websites. Still our distributions may be biased. In particular, we may look at more unpopular sites than earlier studies.

Such previous work has not analyzed complexity (size), cloning, semantical issues, P3P extensions, and language coverage (except for some basic considerations in [5]). This is largely a consequence of a focus on adoption as opposed to our focus on language-usage analysis, and our background in software-language engineering.

C. Clone detection

Clone detection is typically used to locate duplicated code (as evidence of ad-hoc reuse, say copy&paste) [16]. Our textual clone detection is indeed meant to find instances of copy&paste. In fact, we test for identical text (as opposed to type-1 clones [16]) because we are specifically interested in exact copy&paste. Our syntactical clone detection, which is AST-based [14], addresses the additional objective of finding 'common policies'. Here, we leverage aggressive abstraction in the mapping from the concrete XML representation to ASTs. Trying to find 'common' Java or C++ programs would be a fruitless undertaking, but it is feasible for P3P because of the simplicity of the P3P language and policies—especially after abstraction.

Overall, it is simple to accommodate clone detection for P3P because the P3P syntax is trivial (there are not even identifiers to be considered for type-2 clones), typical policies are small (compared to mainstream languages), typical P3P corpora are small, too. Hence, implementation is straightforward, and no scalability challenge must be tackled. It is hard to detect semantical clones for general programming languages, but it is straightforward for P3P with its finite semantical domains.

X. Conclusion

One can try to understand P3P as defined through W3C, textbooks, and online samples. A different language can be observed on the grounds of actual usage. The language 'in the wild' is affected by people's understanding, copy&paste, legislation, tool support, and others. By inquiring language usage, one can deepen a language's understanding in an exploratory manner. We have provided and interpreted data about the size of P3P policies, cloning habits, the coverage of the language constructs and data items as well as usage of P3P's extension mechanism.

There are several aspects of P3P language usage that we could not discuss in this short paper but plan to present in a forthcoming publication or report, e.g., policy-style analysis, similarity analysis (in generalization of clone detection), and additional forms of visualizing and interpreting coverage for P3P constructs and its data schema. We also plan to apply the developed notion of language-usage analysis to other languages.

References

[1] W3C, "The Platform for Privacy Preferences 1.1 (P3P1.1) Specification," 2006, http://www.w3.org/TR/P3P11/.

[2] L. F. Cranor, *Web Privacy with P3P*. O'Reilly & Associates, 2002.

[3] S. Egelman, L. F. Cranor, and A. Chowdhury, "An Analysis of P3P-enabled Web Sites among Top-20 Search Results," in *ICEC*, ser. ACM International Conference Proceeding Series, M. S. Fox and B. Spencer, Eds., vol. 156. ACM, 2006, pp. 197–207.

[4] I. Reay, P. Beatty, S. Dick, and J. Miller, "A Survey and Analysis of the P3P Protocol's Agents, Adoption, Maintenance, and Future," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 2, pp. 151–164, April-June 2007.

[5] L. F. Cranor, S. Egelman, S. Sheng, A. M. McDonald, and A. Chowdhury, "P3P Deployment on Websites," *Electronic Commerce Research and Applications*, vol. 7, no. 3, pp. 274–293, 2008.

[6] I. Reay, S. Dick, and J. Miller, "A Large-scale Empirical Study of P3P Privacy Policies: Stated Actions vs. Legal Obligations," *ACM Transactions on the Web (TWEB)*, vol. 3, no. 2, 2009.

[7] G. Hogben, T. Jackson, and M. Wilikens, "A Fully Compliant Research Implementation of the P3P Standard for Privacy Protection: Experiences and Recommendations," in *Proceedings of ESORICS 2002*, ser. LNCS, vol. 2502. Springer, 2002, pp. 104–125.

[8] W3C, "A P3P Preference Exchange Language 1.0 (APPEL1.0), W3C Working Draft," 2002, http://www.w3.org/TR/P3P-preferences/.

[9] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "An XPath-based preference language for P3P," in *Proceedings of WWW 2003*. ACM, 2003, pp. 629–639.

[10] ——, "XPref: a preference language for P3P," *Computer Networks*, vol. 48, no. 5, pp. 809–827, 2005.

[11] T. Yu, N. Li, and A. I. Antón, "A formal semantics for P3P," in *Proceedings of SWS 2004*. ACM, 2004, pp. 1–8.

[12] T. J. McCabe, "A Complexity Measure," *IEEE Trans. Software Eng.*, vol. 2, no. 4, pp. 308–320, 1976.

[13] M. H. Halstead, *Elements of Software Science*. Elsevier Science Inc., New York, NY, 1977.

[14] I. D. Baxter, A. Yahin, L. Moura, M. Sant'Anna, and L. Bier, "Clone detection using abstract syntax trees," in *Proceedings of ICSM 1998*. IEEE Computer Society, 1998, p. 368.

[15] S. Ducasse, M. Rieger, and S. Demeyer, "A Language Independent Approach for Detecting Duplicated Code," in *Proceedings of ICSM 1999*. IEEE Computer Society, 1999, pp. 109–118.

[16] R. Koschke, "Identifying and removing software clones," in *Software Evolution*. Springer, 2008, pp. 15–36.

[17] D. E. Knuth, "An Empirical Study of FORTRAN Programs," *Software, Practice & Experience*, vol. 1, no. 2, pp. 105–133, 1971.

[18] S. K. Robinson and I. S. Torsun, "An Empirical Analysis of FORTRAN Programs," *The Computer Journal*, vol. 19, no. 1, pp. 56–62, 1976.

[19] R. J. Chevance and T. Heidet, "Static profile and dynamic behavior of COBOL programs," *SIGPLAN Notices*, vol. 13, no. 4, pp. 44–57, 1978.

[20] R. P. Cook and I. Lee, "A Contextual Analysis of Pascal Programs," *Software, Practice & Experience*, vol. 12, no. 2, pp. 195–203, 1982.

[21] H. J. Saal and Z. Weiss, "An empirical study of APL programs," *Computer Languages*, vol. 2, pp. 47–59, 1977.

[22] J. Y. Gil and I. Maman, "Micro patterns in Java code," in *Proceedings of OOPSLA 2005*. ACM, 2005, pp. 97–116.

[23] G. Baxter, M. Frean, J. Noble, M. Rickerby, H. Smith, M. Visser, H. Melton, and E. Tempero, "Understanding the shape of Java software," in *Proceedings of OOPSLA 2006*. ACM, 2006, pp. 397–412.

[24] C. S. Collberg, G. Myles, and M. Stepp, "An empirical study of Java bytecode programs," *Software, Practice & Experience*, vol. 37, no. 6, pp. 581–641, 2007.

[25] R. Lämmel, S. Kitsis, and D. Remy, "Analysis of XML schema usage," in *Conference Proceedings XML 2005*, Nov. 2005.