Reducing the Complexity of Semantic Data Translation

Owen Gilson*, Phil W. Grant, and Min Chen

Department of Computer Science, University of Wales Swansea, Singleton Park, Swansea SA2 8PP, United Kingdom

Abstract. In order to address the restrictions of existing program translation techniques an alternative method is proposed. The technique is demonstrated in a software tool which is used to translate between simple graphics languages. The tool uses the output errors from a schema validator to iteratively evolve from a source to a target program.

Keywords: XML, schema validator, language translation, semantic equivalence.

1 Introduction

There is a wide variety of semantically marked up languages. Translating between any two of these languages is a laborious task which requires syntactic and semantic mappings to be created. Given a system with n languages, building a direct translator between every pair of languages would require n(n-1) translators, a large amount of work. Also, adding a new language to the system will require 2(n-1) additional translators to be built.

2 Related Work

Much work on translation between computer languages was done in the 1960s. Andreyev[1] classified two approaches to language mapping, "binary translation" and "translation via an intermediate language". Binary translation is the most accurate form, but has the complexity overheads described above. Intermediate translation has a reduced complexity (2n translators), but is susceptible to the constraints of the intermediate language.

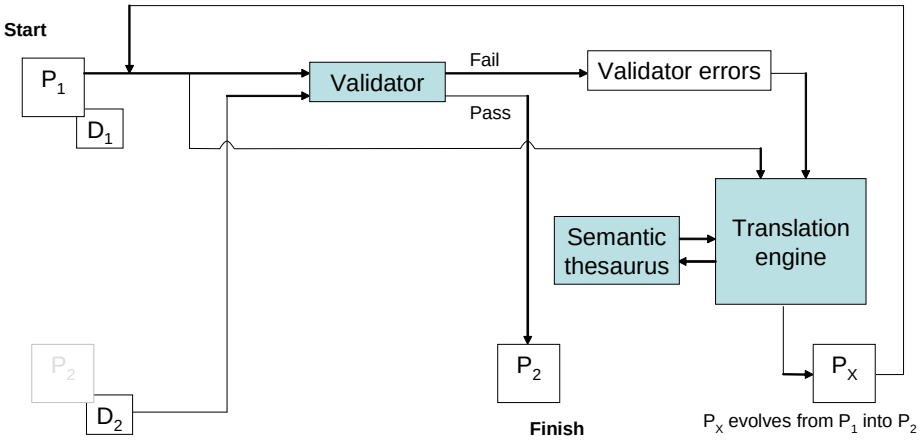
3 Method

We have investigated a technique which enables the complexity of this process to be reduced. For each language in the system there is an associated syntactic descriptor. To translate between any two languages, a generic translator is used to mould a source program into the target program using the target's syntactic

* Contact author: csowen@swan.ac.uk

descriptor to guide this process. In this way, the complexity of the system is reduced to n descriptors and a single general purpose translator. Adding a new language to the system will only require one additional descriptor to be built.

4 Technique



The system converts a source program, P_1 into the target program P_2. Both P_1 and P_2 have associated syntactic descriptors known as D_1 and D_2. The Validator takes P_1 and validates it against D_2 to produce a set of validator errors. The Transalation Engine uses these errors to produce a new version of P_1 (known as P_X) where the errors have been resolved. The Semantic Thesaurus provides synonyms for language elements found within the program being translated.

The validator is based on the conformant schema-aware processor, XSV(XML Schema Validator)[2]. This validator takes an XML document together with an XML Schema (XSD file). If the document fails validation, the validator produces a set of errors. These errors are used in the translation engine to produce a new version of the source program where these errors are fixed. The new updated version of the program (P_X) is then returned to the validator.

The translation engine attempts to resolve the validator errors by referring to a semantic thesaurus which consists of a large collection of synonyms for language elements. The thesaurus may be specific to the domain of languages being translated (e.g. graphics languages), or could be general purpose.

When a version of P_X correctly validates against D_2, this version of P_X is deemed to be a successful translation to P_2 and the process is complete.

5 Software Proof of Concept

A software tool has been developed to demonstrate the technique. The translation domain has been limited to simple graphical scene description languages for which the tool successfully demonstrated the feasibility of this technique.

6 Future Work

We have demonstrated the concept of the technique. However, in order to make the technique more applicable to real-life languages and applications further work is needed.

XML is capable of expressing rich constructs as defined by Clover[3]. It is intended to extend the tool to handle these fully.

As the complexity of languages in the system increases, situations will arise where multiple paths of translation will be required. It is proposed to investigate resolution of multiple paths both programmatically and by the user.

The output from XSV is a human-readable error message rather than marked-up XML. A more elegant solution for interfacing XSV with the translator will be investigated.

There are additional ways of determining semantic equivalence which we would like to investigate. Equivalence is currently derived from a list of synonyms. An ontological representation of the knowledge would allow a greater level of refinement. Also we believe that comparing the similarity of construct features may prove a better metric of equivalence.

7 Conclusions

1. Existing methods of language translation lead to purpose-built translators and a potentially huge number of translators.
2. We have described an alternative technique for language translation.
3. The technique has been demonstrated in a software tool for translating between a simple subset of graphics languages.
4. The software tool relies on a feedback loop from the output of a schema validator to a translation engine which continues until the validator reports no errors.
5. We have proposed work which could be carried out to investigate the validity of this technique in real-life scenarios.

References

1. N. Andreyev. Machine Translation, chapter The intermediary language as the focal point of machine translation. John Wiley Sons, Inc. New York., 1967.
2. Thompson, Henry S., and Tobin, Richard. XSV (XML Schema Validator). 22 April 2005. http://www.ltg.ed.ac.uk/~ht/xsv-status.html
3. Clover, R. XML and Semantic Transparency. 23 October 1998. http://www.oasis-open.org/cover/xmlAndSemantics.html