А. С. Тощев
Казанский (Приволжский) федеральный университет,
sanchis.no@gmail.com

НОВАЯ КОНЦЕПЦИЯ АВТОМАТИЗАЦИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Существует множество инструментов для автоматизации разработки, например, современные интегрированные среды разработки, - Microsoft Visual Studio 2010 [1], Microsoft Visual Studio 2008 [1], IntelliJ IDEA [2],- включают технологии типа IntelliSense [3] (технология автодополнения ввода пользователя). Также стоит отметить богатые средства моделирования UML [4] включают в себя множество готовых архитектурных шаблонов, средства управления циклом разработки, такие как MAVEN [5] (сборщик приложений), включают множество шаблонов для генерации скелета приложения. Основной минус этих систем состоит в том, что они не "понимают"àрхитектуры приложения как целостной системы. Например, инструмент сгенерировал код приложения, разработчик заполнил недостающие тела методов. Понадобилось добавить поле в Базу Данных, и инструмент опять перегенерировал, все приложение и вся работа разработчика была потеряна. В целом ни один из инструментов не автоматизирует весь цикл разработки, только лишь его части. В 2005 году в Массачусетский технологический институте было опубликовано исследование по исследованию генерации классов Python на основе сокращенного английского языка [6].Подход заключался в разборе текста на естественном языке и переводе его в семантическую модель концепций языка программирования.

Наше представление об эффективной автоматизации разработки программного обеспечения заключается в следующем: Входными параметрами системы являются бизнес требования созданные аналитиком. Выходными параметрами является готовый код приложения Бизнес требования должны отвечать следующим параметрам:

1. Отсутствие неоднозначности;

2. Грамматическая верность;

3. Отсутствие противоречий (это должно быть выявлено системой и решено с помощью эксперта);

Изначально идея была в том чтобы улучшить прототип Массачусетского технологического института - Metafor. Вначале мы решили создать "понимательный"механизм архитектуры приложения Выяснилось что создать прямолинейное сопоставление объектов из текста на классы в приложении не вариант решения проблемы. Поэтому мы добавили некоторую промежуточную, метамодель требований. Нами был использован уже готовый лексический обработчик "Stanford Parser"[7]. Отталкиваясь от него и была формализована архитектура генератора: Лексический обработчик ? Понимательный модуль ? Модуль генерации решения Все модули используют Базу знаний для хранения информации. В дополнения может быть создан модуль общения, через который остальные модули смогут общаться с экспертом.

Данная система является Open Source проектом Menta [8]. Мы считаем, что в конечном итоге нам удастся автоматизировать как минимум 10 % всех задач разработки, то есть убрать тривиальные задачи и дать возможность разработчику работать над сложными и интересными задачами.

Л И Т Е Р А Т У Р А

1. *Visual Studio* //Wikipedia –
http://ru.wikipedia.org/wiki/Visual_Studio
2. *IntelliSense* //Wikipedia –
http://ru.wikipedia.org/wiki/IntelliJ_IDEA
3. *IntelliJ IDEA* //Wikipedia –
http://ru.wikipedia.org/wiki/IntelliSense
4. *UML* //Wikipedia –
http://ru.wikipedia.org/wiki/UML
5. *Introduction to the Build Lifecycle* //Apache Maven Project, Apache Software foundation, –
http://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html.

6. . Lieberman H., Liu H. *Metafor: Visualizing Stories as Code.* // Proceedings of the ACM International Conference on Intelligent User Interfaces, IUI 2005, January 9-12, 2005, San Diego, CA, USA ACM Press. – 2005. – P. 305-307 –
http://larifari.org/writing/IUI2005-Metafor.pdf.

7. Klein D., Manning C. *The Stanford Parser: A statistical parser* // The Stanford Natural Language Processing Group –
http://nlp.stanford.edu/software/lex-parser.shtml

8. *The Menta Project*, Menta-project Web Site –
http://menta-project.org

9. Рассел С., Норвиг П. *Искусственный интеллект: современный подход.* – М.: Изд-во "Вильямс 2007. – 1408 с.