Заключение

В диссертационной работе рассмотрены основные проблемы, связанные с реализацией международного стандарта языка программирования Си++. На основе анализа лексической и синтаксической структуры языка предлагаются и обосновываются принципы и методы реализации соответствующих фаз компиляции. Обсуждается одно из базовых понятий языка - область действия и предлагается модель организации семантических таблиц компилятора, адекватно отражающая это понятие. Рассматриваются способы "погружения" в предложенную модель таблиц механизма типовой параметризации Си++ (шаблонов), а также методы поддержки раздельной компиляции и модульности на основе семантических таблиц.

Основной **методологический результат** диссертационной работы заключается в формулировании и обосновании концепции непрерывной компиляции. Эта концепция, отдельные положения которой рассматривались во введении, а также в главах 1, 2 и 4, позволяет преодолеть недостатки, свойственные традиционным подходам к компиляции, обеспечивает высокоуровневый диагностический сервис, адекватную поддержку новых языковых возможностей (в частности, шаблонов) и потенциально более высокую эффективность процесса разработки программ.

Концепция **непрерывной компиляции** образуется из следующих аспектов:

1. Фаза препроцессирования рассматривается как составная часть функциональности лексического анализатора, в противоположность традиционным подходам, реализующим препроцессирование в виде отдельной компоненты системы программирования. Данное решение повышает общую эффективность компилятора, исключая операции с промежуточным текстовым представлением программы, и делает лексический анализ "гладким". Исключение фазы препроцессирования, не ограничивая возможностей Си++ и потребностей программистов, создает предпосылки для поддержания полной семантической адекватности исходного текста и результирующих структур компиляции.

2. Внутренние структуры компилятора - семантические таблицы, дерево программы и представление системы типов - обычно используемые исключительно для внутренних целей генерации результирующего кода, логически выносятся вовне компилятора переднего плана и становятся результатом его работы. Указанные структуры играют роль промежуточного представления (ПП) исходной программы и интерфейса компилятора переднего плана с различными конечными процессорами, в частности, с генератором кода.

3. Процедуры идентификации имен переносятся на этап лексического анализа. В сочетании с практически полной однопроходностью это означает, что все компоненты компилятора участвуют в построении семантических таблиц и результирующего ПП в

целом. Результаты работы каждой компоненты, существенные для поддержания адекватности исходного текста и ПП, не теряются, а помещаются в ПП.

4. Концептуально компилятор переднего плана является однопроходным. Все его компоненты взаимодействуют по подпрограммному принципу, что исключает использование каких-либо громоздких промежуточных структур. Введение в реализацию дополнительного второго прохода не является принципиальным решением; оно продиктовано стремлением сделать общую структуру компилятора более обозримой и сопровождаемой и тем самым провести частичную "декомпозицию сложности" семантики Си++. Дополнительный проход компилятора не связан с порождением каких-либо новых структур и представляет собой однократный проход по уже построенному дереву программы с вычислением ряда семантических атрибутов и локальными преобразованиями поддеревьев.

5. Предложенная в работе и реализованная в компиляторе модель семантических таблиц дает возможность без значительных накладных расходов организовать компиляцию единицы трансляции в контексте откомпилированных ранее единиц. Реализация такой возможности существенно упрощает компонентную структуру среды программирования и преодолевает ряд проблем, связанных с межмодульными интерфейсами Си++, многие из которых практически не имеют решения в традиционных архитектурах систем программирования. Естественным следствием такой модели компиляции может служить исключение понятия единицы трансляции из практического использования и трактовка программы на Си++ как линейной или иерархической совокупности описаний сущностей. Тем самым создание программы с точки зрения разработчика приобретает вид серии непрерывных манипуляций с описаниями; результат каждой операции с описанием компилируется в общем контексте программы и немедленно помещается в ПП. "Гладкость" и непрерывность этого процесса обеспечивается сравнительно небольшим объемом вносимых изменений (так как квантом изменений служит отдельное описание, а не единица трансляции) и, как следствие, весьма незначительными накладными расходами на компиляцию.

Также в процессе исследований получены следующие **результаты**:

- Предложена модель промежуточного представления программ на языке Си++, образуемая системой семантических таблиц, деревом программы и представлением системы типов программы.

  Основной компонентой ПП является система семантических таблиц. Выполнена реализация этой модели в действующем компиляторе переднего плана Си++. Показано, что устройство модели семантических таблиц компилятора обладает необходимой общностью, что позволяет без серьезных затрат реализовать в ее рамках концепцию непрерывной компиляции.
- Разработана общая архитектура компилятора переднего плана Стандарта Си++, а также компонентная структура компилятора.

Предлагаемый способ построения компилятора предусматривает сохранение и последующее использование результирующего ПП, что создает предпосылки для реализации концепции непрерывной компиляции.

Разработаны и реализованы компоненты компилятора переднего плана, выполняющие лексический, синтаксический и семантический анализ.

- Создан первый в России промышленный компилятор языка Си++ и один из первых компиляторов, реализующих полный стандарт Си++.