инженерия программного обеспечения

Фролов Денис Андреевич

УДК 004.424

РАЗРАБОТКА УНИВЕРСАЛЬНОЙ БИБЛИОТЕКИ АСПЕКТОВ ДЛЯ НАДЁЖНЫХ И БЕЗОПАСНЫХ ВЫЧИСЛЕНИЙ В ПРИЛОЖЕНИЯХ .NET

Аннотация

В связи со стремительным развитием информационных технологий в наше время информация приобретает всё большую значимость. Утечка важных данных может привести к катастрофическим последствиям. Посредством интернета, локальной или беспроводной сети злоумышленники могут получить доступ к секретной информации, которая им совсем не предназначена. Это может привести как к финансовым, так и к информационным потерям. Таким образом, на данный момент IT-безопасность становится необходимым элементом при построении любой IT-инфраструктуры и напрямую влияет на целостность и конфиденциальность всей цифровой информации в целом. Учитывая скорость развития информационных технологий и повседневного взаимодействия с ними в жизни каждого отдельно взятого человека, понятия надёжности и безопасности хранения, обработки и передачи частной информации приобретают новую значимость. С учетом всех реалий можно выделить основную идею: необходимо иметь в виду постоянное требование надёжности и безопасности вычислений (TWC, от Trustworthy Computing) при разработке и сопровождении продуктов в сфере ИТ. В данной работе предлагается реализовать методы надёжности и безопасности программирования с помощью аспектно-ориентированного программирования.

Ключевые слова: аспектно-ориентированное программирование, Aspect.NET, надежность и безопасность вычислений, IT безопасность.

В связи со стремительным развитием информационных технологий в наше время информация приобретает всё большую значимость и стоимость. Обладание эксклюзивной информацией зачастую является источником благополучия не только на уровне отдельных компаний, но и на государственном уровне, что приводит к увеличению ее ценности. При этом утечка важных данных может привести к катастрофическим последствиям. Посредством интернета, локальной или беспроводной сети злоумышленники могут получить доступ к секретной информации, которая им совсем не предназначена. Это может привести как к финансовым, так и к информационным потерям. Понятия «кибер-атака», «кибер-оружие» уже не представляется чем-то фантастическим, а является реальностью, с которой может столкнуться любой пользователь. Таким образом, на данный момент IT-безопасность становится необходимым элементом при постро-

ении любой IT-инфраструктуры и напрямую влияет на целостность и конфиденциальность всей цифровой информации в целом.

Согласно результатам ежегодного исследования об утечках конфиденциальной информации, проведённого аналитическим центром компании Zecurion, в 2011 г., было зарегистрировано всего 819 инцидентов, а суммарный ущерб оценивается более чем в $20 млрд., из которых более $1 млрд. пришлось на российские компании. По словам руководителя аналитического центра Zecurion Владимира Ульянова, в условиях ограниченного бюджета на информационную безопасность, что справедливо в среднем для 19 из 20 компаний, приоритет должен отдаваться наиболее эффективным инструментам [1].

Учитывая скорость развития информационных технологий и повседневного взаимодействия с ними в жизни каждого отдельно взятого человека, понятия надёжности и безопасности хранения, обработки и передачи частной информации приобретают новую значимость. В ближайшем будущем ИТ-отрасль будет продолжать бороться против растущих атак и угроз в отношении данных. Но нынешняя стратегия, применяемая многими компаниями, в связи с ограниченными финансовыми ресурсами направлена на краткосрочное разрешение проблем и не приносит должного эффекта. Необходимо в корне изменить методы и подходы в сфере безопасности информационных технологий и постараться реализовать следующее:

– разработать новые подходы для уничтожения широко распространённых атак и угроз в киберпространстве. Упор при этом делать на качественные программные продукты;

– удостовериться, что вновь создаваемые системы при применении этих же методов и подходов остаются неуязвимыми;

– предоставить разрабатываемые инструменты принимающим решения лицам в правительстве и ИТ-отрасли для возможности будущего инвестирования в информационную безопасность (создание библиотеки аспектов, специфичных для данного приложения);

– разработать новые вычислительные системы так, чтобы безопасность и надёжность этих систем были понятны и управляемы обычным пользователем, при этом гарантировалась конфиденциальность его данных.

С учетом всех реалий можно выделить основную идею: необходимо иметь в виду постоянное требование надёжности и безопасности вычислений (TWC, от Trustworthy Computing) при разработке и сопровождении продуктов в сфере ИТ.

Одним из подходов к реализации TWC, рассматриваемым в данной статье, является аспектно-ориентированное программирование (АОП). Другим подходом, применимым для TWC, является инженерия знаний, более точно – разработка и практическое использование баз знаний о надёжности программ и о методах разработки надёжного и безопасного кода. Более простой подход – применение шаблонов кода (code patterns), позволяющее в некоторых наиболее простых случаях избежать ошибок, связанных с безопасностью [2].

Такие новые интересные и современные направления, как АОП и TWC, очень тесно связаны между собой. Суть реализации TWC в большинстве случаев состоит в систематических групповых «сквозных» добавлениях в существующий код каких либо действий или проверок, для чего и предназначено АОП [2].

Аспектно-ориентированное программирование – активно развивающаяся технология разработки программного обеспечения для модуляризации и использования сквозной функциональности (cross-cutting concerns). Во-первых, использование АОП позволяет определить спецификацию TWC в виде отдельного модуля (аспекта), обеспечивая ясность, наглядность и легкость сопровождения исходного кода программы. Во-вторых, АОП предоставляет возможность автоматического добавления новой сквозной функциональности в код целевых приложений, в отличие от традиционных сред разработки, в которых эти операции приходится выполнять вручную. Наконец, АОП широко поддерживается различными средами разработки программного обеспечения, что спо-

собствует распространению концепции TWC на практике [2]. Функциональность АОП базируется на основной единице – аспекте, что следует из самого названия данного подхода. Аспект является модулем, реализующим сквозную функциональность. Данный модуль вызывается в программе в специальных точках внедрения аспекта – join points. В свою очередь, аспектно-ориентированный подход рассматривается как возможность обеспечения локализации сквозной функциональности в аспектах. Главной особенностью использования аспектного подхода является его интеграция в уже существующие системы программирования. АОП не отрицает процедурного программирования или объектно-ориентированного подхода, а дополняет их, представляя собой новую ветку в эволюции технологий программирования, предлагая большие перспективы и новые решения [5].

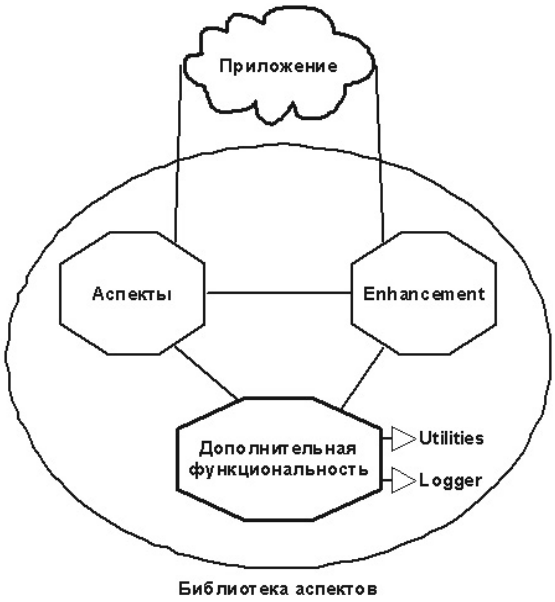

Рис. 1. Структура библиотеки аспектов

Принцип надёжных и безопасных вычислений подразумевает построение приложений таким образом, чтобы все исключительные ситуации обрабатывались системой рациональным способом с целью минимизации потерь – простоев системы, потери информации, несанкционированного доступа к информации и т. д. – вследствие системных либо пользовательских ошибок. Начало инициативе TWC было положено письмом президента Microsoft Билла Гейтса в январе 2002 года, направленное всем сотрудникам компании, подчеркивающее необходимость поставлять более надёжные и безопасные приложения пользователям в связи с резким возрастанием угроз компьютерным системам [4].

Обычно задачи TWC относятся к сквозной функциональности и вплетены в ядро приложения, что значительно усложняет код приложения, а соответственно, и его поддержку и расширение. Также это приводит к возникновению дополнительных ошибок, вызванных запутанной логикой и сложностью приложения. Если выделить данные задачи в отдельные аспекты, логика приложения становится более прозрачной и понятной другим разработчикам.

В данной работе предлагается реализовать методы надёжности и безопасности

Листинг 1

```
// Проверка прав запуска защищенной страницы
[AspectAction("%after %call System.Windows.Forms.Form.Activate(..)")]
public static void CheckAccess()
{
   Form page = (Form)TargetObject;
   if (!AuthorizationRequires(page))
      return;
   if (IsInRole(RoleWithPrivateAccess))
      return;
   CurrentApplication.GoToStartPage();
    CurrentApplication.SetStatus('' Недостаточно полномочий для просмотра.'');
}
```

программирования с помощью АОП. Для того чтобы полученная библиотека аспектов была универсальной, она должна содержать в себе аспекты для стандартных классов .NET, то есть классов из пространства имен, вложенных в namespace System. На данный момент созданы аспекты для работы с классами оконных и консольных приложений. В перспективе возможно дополнение библиотеки аспектами TWC и для других видов приложений.

Одним из основных требований к разрабатываемой библиотеке является минимизация трудоёмкости при интеграции с улучшаемым приложением. Поэтому функциональная часть представляет собой следующую картину, показанную на рис. 1. Приложение инициирует вызовы методов аспектов (AspectActions), которые, в свою очередь, взаимодействуют с ядром библиотеки, а также вызывают методы приложения, используя механизм расширения библиотеки для конкретного приложения (Enhancement) [3].

Рассмотрим один из методов аспекта безопасности в качестве примера (листинг 1).

Данный метод аспекта вызывается после вызова метода System.Windows.Forms.Form.Activate( ), который в свою очередь вызывается каждый раз, когда приложение становится активным. Далее в свойстве TargetObject содержится ссылка на сам объект Form, в котором отработал указанный метод. Метод AuthorizationRequires(page) проверяет, был ли вызван метод для защищённой страницы. Если страница защищена, то проверяется наличие соответствующих полномочий у пользователя для просмотра защищённой страницы. В случае отсутствия прав доступа вызывается метод CurrentApplication.GoToStartPage( ), который закрывает защищённую страницу и возвращает пользователя на стартовую страницу приложения. Следующим методом CurrentApplication.SetStatus(message) мы информируем пользователя о том, что попытка открыть запрашиваемую страницу не удалась, так как недостаточно полномочий.

В рассмотренном примере в действии метода аспекта требуется реакция приложения – закрыть текущую страницу и перейти на стартовую. Это достигается с помощью упомянутого ранее механизма Enhancement. Этот метод необходимо опре-

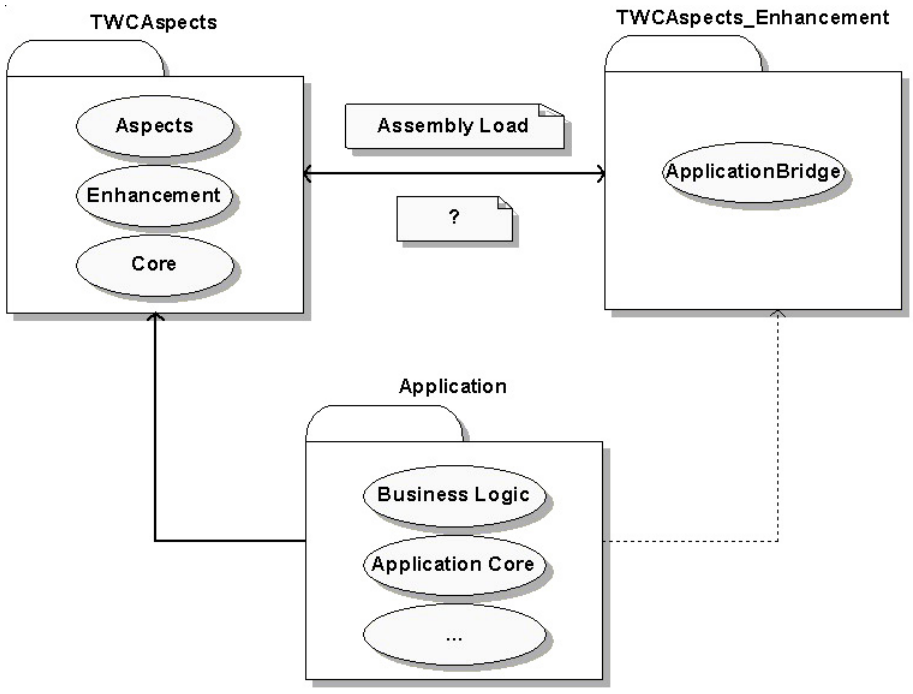

Рис. 2. Проверка наличия библиотеки Enhancement

делять отдельно для каждого приложения в соответствии с его архитектурой. Набор всех таких методов и представляет собой класс ApplicationBridge.

Реализация класса, наследуемого от абстрактного класса ApplicationBridge, должна находиться в отдельной библиотеке (DLL). Во время выполнения методов библиотеки аспектов динамически проверяется наличие DLL, специфичной для данного приложения (см. рис. 2). Этот файл должен находиться в одной директории с исполняемым файлом полученного приложения и называться TWCAspects_Enhancement.dll. Если такой библиотеки не существует, то библиотека аспектов не сможет вызывать методы приложения и будет выполнено действие по умолчанию. Для правильной работы созданной библиотеки необходимо, чтобы связь с изменяемым приложением существовала, то есть была создана DLL с реализацией методов ApplicationBridge. В противном случае теряется часть функциональности библиотеки, тем не менее, сохраняется надёжность при работе с измененным приложением.

Возвращаясь к нашему примеру, покажем (листинг 2) реализацию по умолчанию метода перехода на стартовую страницу.

Понятно, что, ничего не зная об изменяемом приложении, невозможно корректно реализовать даже такую простую операцию, как переход на стартовую страницу. Поэтому во избежание несанкционированного доступа к защищенной странице мы можем только завершить работу приложения. По этой причине требуется реализация расширения библиотеки аспектов для каждого приложения, которое будет дополняться аспектами TWC.

Тем не менее, есть методы приложения, не требующие учета его архитектуры для

Листинг 2

```
// Переход на стартовую страницу приложения
public override void GoToStartPage()
{
  if (Utilities.CurrentApplicationType.Value == ApplicationType.Console)
   {
     Console.WriteLine("Ошибка безопасности. Завершение приложения...");
      // Чтобы пользователь успел прочитать сообщение
      System.Threading.Thread.Sleep(1000);
   }
   if (Utilities.CurrentApplicationType.Value ==
                                          ApplicationType.WindowsForms)
   {
      object caller = AspectDotNet.Aspect.TargetObject;
      if (caller != null)
      {
         Type t = caller.GetType();
         if (t.IsSubclassOf(typeof(Form)))
         {
            MethodInfo mi = t.GetMethod("Close");
            mi.Invoke(caller, null);
           return;
         }
      }
   }
   // С целью защиты информации от несанкционированного доступа
   // не безопасно продолжать работу приложения
   Application.Exit();
}
```

Листинг 3

```
public override void SetStatus(string text)
{
  if (Utilities.CurrentApplicationType.Value == ApplicationType.Console)
  {
     Console.WriteLine(text);
     return;
  }
  if (Utilities.CurrentApplicationType.Value ==
                                          ApplicationType.WindowsForms)
  {
     MessageBox.Show(text,
"Сообщение библиотеки аспектов TWC",
MessageBoxButtons.OK);
     return;
  }
  // Запишем в лог уведомление
  Logger.WriteString(
   String.Format("Message cannot be displayed to user. {0}", text),
   Logger.ASPECT_MSG
  );
}
```

успешной реализации. В качестве примера такого метода, в листинге 3 показан вывод сообщения пользователю из библиотеки аспектов. Для консольных и оконных приложений данная реализация чаще всего является достаточной.

Еще одним таким примером может являться аспект ведения протокола работы программы. В листинге 4 показан пример, пишущий журнал событий программы после вызова метода System.Windows.Forms.Show() – перехода на новую страницу (Form) в оконных приложениях windows.

Таким образом, даже без адаптации библиотеки к изменяемому приложению большая часть функциональности аспектов будет работать корректно.

Рассмотрим процесс интеграции созданной библиотеки с изменяемым приложением более подробно. Он состоит из трех шагов, первые два из которых могут быть пропущены.

1. Создание реализации класса ApplicationBridge.

2. Создание библиотеки аспектов, специфичных для данного приложения.

Листинг 4

```
// Аспект протоколирования
[AspectAction("%after %call System.Windows.Forms.Show(..)")]
public static void FormShow()
{
  if (TargetObject == null)
    return;
  Type type = TargetObject.GetType();
  if (type.IsSubclassOf(typeof(Form)))
    Logger.WriteString(String.Format("Переход на страницу: {0}",
                                                        type.Name));
}
```

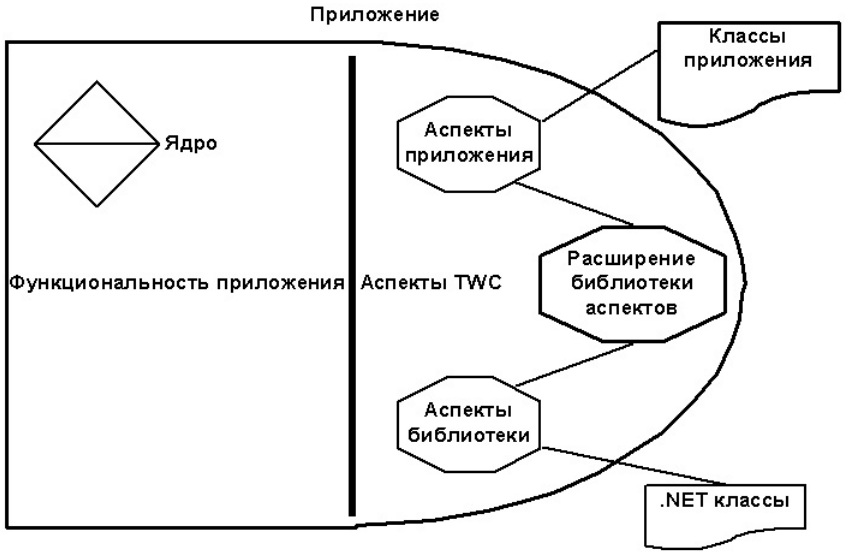

Рис. 3. Реализация приложения с интеграцией библиотеки аспектов TWC

3. Сборка нового приложения с помощью Microsoft Phoenix.

На рис. 3 показано новое приложение, полученное после интеграции в него библиотеки аспектов TWC. Такой подход к внедрению аспектов не требует больших усилий разработчика.

ЗАКЛЮЧЕНИЕ

Нужно отметить, что проблема TWC на сегодняшний день является высокоприоритетной задачей, ведь это напрямую связано с качеством конечного продукта. Все производители программного обеспечения стремятся сделать свои продукты более надёжными, тем самым завоёвывая себе хорошую репутацию и доверие пользователей. С другой стороны, если отбросить аспекты TWC, то разработка ПО значительно упрощается, что приводит к экономии ресурсов производителя. Полученная библиотека аспектов позволяет получить выигрыш в ресурсах без потери качества конечного продукта, что делает её незаменимым инструментом при разработке .NET приложений.

Практический результат данной работы – библиотека аспектов TWC, способная решать следующие задачи:

– надёжность и безопасность;

– доступность системы;

– обеспечение конфиденциальности данных;

– поведение системы в особых ситуациях (обработка ошибок);

– ведение протокола работы программы (logging).

Литература

1. Материалы портала CNews | Безопасность. «Убытки российских компаний от утечек данных превысили $1 млрд.» // Издание о высоких технологиях – CNews / http://www.cnews.ru.

2. *Сафонов В.О.* Современные технологии разработки надёжных и безопасных программ (Trustworthy Computing) // Компьютерные инструменты в образовании, 2008. № 6. С. 25–33.

3. *Фролов Д.А.* Разработка универсальной библиотеки аспектов для надёжных и безопасных вычислений в приложениях .NET. // Мат-Мех, СПИСОК-2011.

4. *Michael Howard, David LeBlanc.* Writing Secure Code. 2nd ed. Redmond, Washington, 2003.

5. *Safonov V.O.* Using aspect-oriented programming for trustworthy software development. Wiley Interscience. John Wiley & Sons, 2008.

Abstract

Due to the prompt growth of IT industry, information getting more valuable in our days. Leak of the important information can cause catastrophic consequences. Malefactors can obtain secret data using networks. Thus for now IT-security became required element in building IT-Infrastructure and has directly connection with integrity and confidentiality of whole digital data. Considering everyday interaction with IT of many different people, concepts of trustworthy, security, processing, transferring of private data getting new significance. Main idea is to keep trustworthy computing (TWC) principles in all stages of software development and products support. Methods of TWC are implemented with aspect-orientated programming (AOP) in this work.

Keywords: aspect-oriented programming, Aspect.NET, Trustworthy computing, IT security.

Фролов Денис Андреевич,
магистр математики и механики,
аспирант, инженер-исследователь
кафедры информатики СПбГУ,
m03fda@mail.ru