Структурно-параметрические связи модулей программных продуктов в рамках системы управления жизненным циклом

С.А. Базуева

Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова

Аннотация: В статье рассматриваются параметры связей модулей программных продуктов в рамках системы управления жизненным циклом изделия. Проведена классификация типов связей и сцепления элементов. Введены параметры нестабильности программных продуктов.
Ключевые слова: жизненный цикл, жизненный цикл изделия, модель жизненного цикла, инновационная деятельность, системная инженерия.

Постановка задачи связи элементов внутри жизненного цикла

Увеличение качества разрабатываемых программных продуктов (ПП) считается актуальной в теории и практике программирования. Согласно требованиям *ISO* 9000–3 необходимо включать измеряемые четкие критерии качества, то есть уровень соответствия реальных показателей требования. Оценка качества – это совокупность операций, состоящие из выбора номенклатуры и сравнения их с базовыми. При расчетных и экспертных методах оценки качества [1] достаточно сложно определять актуальность выбора характеристик качества на начальных этапах, чему и посвящена статья.

Важным принципом структурного проектирования системы управления жизненным циклом является декомпозиция на базовые модули, выделяя модели потоков данных и объектов. В базе модели потока данных производится разбиение по выполняемым функциям, а модель объектов основана на слабо сцепленных сущностях, которые имеют собственные наборы данных, состояния и операции. Согласно определению Г. Майерса [8] модульность – свойство, которое обеспечивает интеллектуальную возможность создания сколь угодно сложной программы, упростив ее

декомпозицию на ряд внутренне связанных и слабо зависящих друг от друга управляемых частей ПП.

Принципы организации программных продуктов

Базой проектирования ПП являются *закон Деметры* (*Law of Demeter*) [2] или *принцип информационной закрытости* [3]. При декомпозиции важно, чтобы число связей между отдельными модулями должно было минимальным («слабое сцепление» – *Low Coupling*), а связность отдельных элементов внутри модуля была максимальной («сильная связность» – *High Cohesion*). Структура ПП должна отвечать следующим требованиям:

- модуль должен инкапсулировать содержимое («черный ящик»);
- модуль должен иметь интерфейс с другими элементами.

Принцип «черного ящика» дает возможность рассматривать структуру модулей независимо, а также легко развивать и корректировать в процессе сопровождения. Интерфейсы позволяют выстраивать систему более высокого уровня, рассматривая модуль как целое и не обращая внимания на внутреннее устройство. Это позволяет управлять взаимодействием модулей программы (интеграцией), определяя гибкость системы.

Структурно-параметрические характеристики качества ПП

Использование при декомпозиции принципа информационной закрытости, определяет применение ***сцепления***, ***связности*** и ***нестабильности***.

Сцепление. Сцепление модулей (*Coupling*) – мера взаимозависимости программного модуля от остальных. Уровень сцепления между классами *A* и *B* увеличивается, если *A* включает атрибуты типа *B, A* обращается к методу объекта *B* и *A* включает метод, ссылающийся на *B*.

Если элементы сильно сцеплены, то система является трудно модифицируемой и сложной в понимании, вот почему сцепление между модулями необходимо минимизировать. Но при отсутствии сцепления

элементы не зависят друг от друга и не формируют единой системы, вот почему они должны быть независимыми настолько это возможно и связываться с помощью простого и устойчивого интерфейса. Слабый уровень сцепления определяет высокое качество ПП за счет:

- уменьшения числа соединений, что снижает риск появления “волнового эффекта” (ошибки в модуле повлияют на работу других);
- снижении “эффекта ряби” (внесение изменений ведет к ошибкам);
- не нужно заботиться о внутренних операторах других модулей;
- упрощение понимания системы.

Слабое сцепление достигается удалением необязательных связей, уменьшением количества необходимых связей и упрощением связей. Для ослабления сцепления минимизируется число параметров связи, создаются прямые межмодульные, локализованные, явные и гибкие связи.

На практике различают нормальное сцепление, сцепление по внешним ссылкам, по общей области и по содержимому. Модули *A* и *B* являются нормально сцепленными, если *A* вызывает *B*, *B* возвращает управление *A*, передаваемые данные представляются значениями при вызове. Нормальное сцепление существует в виде сцепления по данным, образцу и управлению [4].

Сцепление по данным (*data coupling*) – модули *A* и *B* взаимодействуют через передачу параметров. *Сцепление по образцу* (*stamp coupling*) – модули посылают составные информационные объекты с внутренней структурой. *Сцепление по управлению* (*control coupling*) – модуль *A* управляет работой модуля *B* через информационные объекты. Данные типы сцепления в различной степени приемлемы при проектировании. Следующие три типа сцепления выходят за пределы модульности.

Сцепление по внешним ссылкам – модули *A* и *B* ссылаются на общий глобальный элемент данных. *Сцепление по общей области* (*common coupling*)

– модули разделяют общую область глобальных данных. *Сцепление по содержанию* (*content coupling*) – модуль *A* ссылается внутрь *B* различными способами.

Связность модуля. Связность (*cohesion*) – это уровень прочности соединения объектов внутри модуля. Выступает внутренней характеристикой модуля. Чем выше связность, тем лучше результат проектирования. Высокий уровень связности сопряжен с надежностью, устойчивостью, возможностью повторного использования и понятностью. Низкая связность ведет к сложной поддержке, затруднению при тестировании, повторного использования.

Сцепление и связность считаются двумя взаимозависимыми методами измерения разделения системы на части. Высокий уровень сцепления говорит слабой модульной инкапсуляции и препятствует повторному использованию. Выделяют следующие типы связности: временный, функциональный, последовательный, процедурный, информационный, логический и случайный [18]. *Временная связность* – части модуля группируются в определенное время выполнения программы. *Случайная* – модуль не имеет явно выраженных внутренних связей. *Логическая* – части модуля выполняют решение одной подзадачи. *Процедурная* – части модуля связываются порядком выполняемых действий. *Коммуникационная* – группировка происходит, когда части работают с общей структурой данных. *Информационная* – части сгруппированы, таким образом, что выходные данные одной части становятся входными в другой. *Функциональная* – части сотрудничают при реализации задачи. *Объектная связность* – каждая операция обеспечивает функциональность, предусматривающая, что все свойства будут модифицироваться, использоваться и отображаться для роста качества его работы.

Связность снижается, если объекты мало связаны, внутри объектов выполняется множество различных действий, которые используют не

связанные данные. Среди минусов слабой связности [5] рост сложности понимания, поддержки и повторного использования.

Возможны случаи, когда с модулем ассоциируются ряд уровней связности. При этом если модуль имеет несколько связностей, то он получает самый сильный уровень связности. Если действия имеют различные уровни связности, то модуль получает самый слабый уровень.

Нестабильность. *Нестабильность* (*instability*) – определяет насколько действие модуля зависит от всех действий в пределах элемента и насколько все действия зависят от данного. Примером метрики *нестабильности* через *связность* с элементами, сцепленных модулей (*modules' cohesion*), используется *мера Колофелло* (*Kolofello*), оценивающая логическую стабильность модуля [19], как количество изменений – центростремительного и центробежного сцепления [20]. Центростремительное сцепление (C_e) – количество элементов, которые зависят от данного. Центробежное сцепление (C_a) – число элементов, от которых зависит данный. Формула нестабильности [6, 7]:

$$I = \frac{C_e}{C_a + C_e}.$$

Для максимально нестабильного модуля $I = 1$ указывает, что минимальные изменения в классах, приведут к значительным изменениям в данных. Для максимально стабильного модуля $I = 0$.

Меры интеграции представлены в *матрицах мер конвергенции* в форме "объект-объект".

Выводы

1. Для разработки ПП заданного качества, важно синтезировать четкие критерии качества, а также выбрать номенклатуру измеряемых показателей: связность, сцепление и нестабильность.

2. Высокий уровень связности сопряжен с устойчивостью, надежностью, возможностью повторного использования и понятностью. Низкая связность ассоциируется со сложностью поддержки, тестирования, повторного использования и понимания.

3. Нестабильность определяет насколько элемент зависит от всех элементов в пределах модуля или насколько все элементы зависят от данного.

4. Классификация параметров интеграции ПП при решении сложных задач указывает на необходимость количественного определения метрик для их оценивания в математической форме.

Литература

1. Панов М.П. Жизненный путь и цикл развития организации. Практическое пособие. – М.: Инфра-М. – 2016. – 98 с.
2. Гоппа В. Д.Введение в алгебраическую информацию. М.– 1995. – 202 с.
3. Базуева С. А. Оценка уровня горизонтальных связей системных компонентов жизненного цикла изделия // Инженерный вестник Дона, 2018, №3 URL: ivdon.ru/ru/magazine/archive/n3y2018/5153.
4. Лазутин Ю.Д. Качество жизненного цикла промышленных изделий – М.: МГТУ им. Н. Э. Баумана, 2017. – 320 с.
5. Базуева С. А. Построение вероятностной модели жизненного цикла изделия по параметрам уязвимости // Инженерный вестник Дона, 2018, №2. URL: ivdon.ru/ru/magazine/archive/N2y2018/4997.
6. Губич Л., Ковалев М., Паткевич Н. Внедрение на промышленных предприятиях информационных технологий поддержки жизненного цикла продукции – Минск: Беларуская Навука, 2013. – 190 с.
7. Михайлов А.А., Михайлова С.А. Оценка структурной интеграции информационно-измерительных систем – Известия высших учебных заведений. Северо-Кавказский регион. Серия: Технические науки, 2007. №6

с. 22-26.

8. Майерс Г. Надежность программного обеспечения. – М.: Мир. – 1980. – с. 92–113

9. Dale B.G. Managing Quality.–3-rd ed. – Blackwell Publishers, Oxford – 1999. 63 p.

10. Dickenson R.P., Campbell D.R. and Azarov V.N. Quality management implementation in Russia Strategies for change// International Journal of Quality & Reliability Management, №1. – 2000. – pp. 66-81.

References

1. Panov M.P. Zhiznennyj put' i cikl razvitija organizacii. Prakticheskoe posobie. [Course of life and cycle of development of the organization]. M.: Infra-M. 2016. 98 p.

2. Goppa V. D. Vvedenie v algebraicheskuyu informaciyu. [Introduction to algebraic information]. M: 1995. 202 p.

3. Bazueva S. A. Inženernyj vestnik Dona (Rus), 2018, №3. URL: ivdon.ru/ru/magazine/archive/n3y2018/5153.

4. Lazutin Ju.D. Kachestvo zhiznennogo cikla promyshlennyh izdelij. [Quality of life cycle of industrial products]. M.: MGTU im. N. Je. Baumana, 2017. 320 p.

5. Bazueva S. A. Inženernyj vestnik Dona (Rus), 2018, №2. URL: ivdon.ru/ru/magazine/archive/N2y2018/4997.

6. Gubich L., Kovalev M., Patkevich N. Vnedrenie na promyshlennyh predprijatijah informacionnyh tehnologij podderzhki zhiznennogo cikla produkcii. [Introduction at the industrial enterprises of information technologies of support of life cycle of production]. Minsk: Belaruskaja Navuka, 2013. 190 p.

7. Mikhajlov A.A., Mikhajlova S.A. Izvestiya vy`sshix uchebny`x zavedenij. Severo-Kavkazskij region: Texnicheskie nauki, 2007. № 6. pp. 22-26.

8. Majers G. Nadezhnost` programmnogo obespecheniya. [Software Reliability] M.: Mir. 1980. pp. 92-113.

9. Dale B.G. Managing Quality. 3rd ed. Blackwell Publishers, Oxford. 1999. 63 p.

10. Dickenson R.P., Campbell D.R. and Azarov V.N. International Journal of Quality & Reliability Management, 17. №1. 2000. pp. 66-81