Методология измерения функционального размера COSMIC

Версия 3.0

Общие представления

Сентябрь 2007

БЛАГОДАРНОСТИ

Авторам методологии «COSMIC-FFP»[1] 2.0 (в алфавитном порядке)

Alain Abran, École de technologie supérieure – Université du Québec,
Jean-Marc Desharnais, Software Engineering Laboratory in Applied Metrics - SELAM,
Serge Oligny, Bell Canada,
Denis St-Pierre, DSA Consulting,
Charles Symons, Software Measurement Services, UK

Рецензентам версии 2.0 1998/1999 (в алфавитном порядке)		
Moritsugu Araki, JECS Systems Research, Япония	Thomas Fetcke, Германия	Patrice Nolin, Hydro Québec, Канада
Fred Bootsma, Nortel, Канада	Eric Foltin, University of Magdeburg, Германия	Marie O'Neill, Software Management Methods, Ирландия
Denis Bourdeau, Bell Canada, Канада	Anna Franco, CRSSM, Канада	Jolijn Onvlee, Нидерланды *
Pierre Bourque, ÉCole de Technologie supérieure, Канада	Paul Goodman, Software Measurement Services, Великобритания	Laura Primera, UQAM, Канада
Gunter Guerhen, Bürhen & Partner, Германия	Nihal Kececi, University of Maryland, США	Paul Radford, Charismatek, Австралия
Sylvain Clermont, Hydro Québec, Канада	Robyn Lawrie, Австралия	Eberhard Rudolph, Германия
David Déry, CGI, Канада	Ghislain Lévesque, UQAM, Канада	Grant Rule, Software Measurement Services, Великобритания *
Gilles Desoblins, Франция	Roberto Meli, Data Processing Organization, Италия	Richard Stutzke, Science Applications Int'l Corporation, США
Martin D'Souza, Total Metrics, Австралия	Pam Morris, Total Metrics, Австралия *	Ilionar Sylva, UQAM, Канада
Reiner Dumke, University of Magdeburg, Germany	Risto Nevalainen, Software Technology Transfer Finland, Финляндия *	Vinh T. Ho, UQAM, Вьетнам
Peter Fagg, Великобритания	Jin Ng, Hmaster, Австралия	

* Организаторы основополагающей команды COSMIC, включая авторов метода COSMIC-FFP 2.0

[1] Версия 2.0 была первой доступной широкой публике версией метода, известного ранее как «COSMIC-FFP»

Рецензентам Версии 3.0 2006/07 (в алфавитном порядке)		
Alain Abran, École de Technologie Supérieure, Université du Québec, Канада	Jean-Marc Desharnais, Software Engineering Lab in Applied Metrics – SELAM, Канада	Arlan Lesterhuis*, Sogeti, Нидерланды
Bernard Londeix, Telmaco, Великобритания	Roberto Meli, Data Processing Organization, Италия	Pam Morris, Total Metrics, Австаралия
Serge Oligny, Bell Canada, Канада	Marie O'Neill, Software Management Methods, Ирландия	Tony Rollo, Software Measurement Services, Великобритания
Grant Rule, Software Measurement Services, Великобритания	Luca Santillo, Agile Metrics, Италия	Charles Symons*, Великобритания
Hannu Toivonen, Nokia Siemens Networks, Финляндия	Frank Vogelezang, Sogeti, Нидерланды	

* Редакторам версии 3.0 методологии COSMIC

Переводчикам Общих представлений о методологии COSMIC версии 3.0.1
Русаков Максим, генеральный директор SRG-IT, Россия, rmy@srgroup.ru
Колесников Григорий, консультант SRG-IT, Россия, kgv@srg-appraisal.ru

УПРАВЛЕНИЕ ВЕРСИЯМИ

В таблице ниже приведены и обобщены изменения, произошедшие с документом «Общие представления».

ДАТА	РЕЦЕНЗЕНТ(Ы)	Изменения / Дополнения
Сентябрь 2007	Комитет по проведению измерений COSMIC	Выпуск первой публично доступной версии. Содержание частично основано на второй главе «Руководства по измерению» версия 2.2, но существенно расширено.

ПРЕДИСЛОВИЕ

COSMIC - это стандартизированный метод измерения функционального размера программного обеспечения (ПО), используемый преимущественно в предметной области бизнес-приложений и приложений, работающих в режиме реального времени, а также для приложений, включающих функциональные элементы обоих типов таких ПО.

В декабре 2002 COSMIC был одобрен Международной Организацией по Стандартизации (ISO/IEC JTC1 SC7) в качестве международного стандарта ISO/IEC 19761 «Проектирование программного обеспечения – COSMIC-FFP – Метод измерения функционального размера программного обеспечения» («Software Engineering – COSMIC-FFP – A functional size measurement method») (здесь и далее «ISO/IEC 19761»).

Задачи настоящего документа

Задача документа «Методология COSMIC 3.0. Общие представления» - кратко изложить метод измерения функционального размера COSMIC версии 3.0. Этот документ будет интересен читателям, которые:

- незнакомы с измерением функционального размера ПО и нуждаются во введении в предмет;
- хотят получить общее представление о методологии COSMIC без погружения в детали;
- знакомы с уже существующими методами измерения функционального размера ПО «первого поколения» (такими как «IFPUG», «MkII» или «NESMA») и которые рассматривают возможность применения методологии COSMIC.

Документация методологии COSMIC

Для полного ознакомления с документацией методологии COSMIC, пожалуйста, обратитесь к документу «Методология COSMIC 3.0. Документация и Термины», который содержит полный список терминов и определений. Этот документ и остальные документы COSMIC можно найти на сайте www.cosmicon.com в открытом доступе.

Важные документы, которые могут быть интересны читателям:

- Стандарт ISO/IEC 19761. Включает основные нормативные определения и правила методологии;
- «Методология COSMIC 3.0. Руководство по измерению». Содержит правила и определения методологии, а также подробные объяснения и примеры для наиболее полного понимания и применения методологии. Это основной рабочий документ, который понадобится пользователям на практике;
- «Методология COSMIC 3.0. Более сложные задачи и связанные с этим проблемы». Содержит помимо базовой методологии такие дополнительные материалы, как: приблизительная оценка размера ПО на ранних стадиях проекта, конвертация функционального размера, измеренного другими методами, в функциональный размер COSMIC и др.
- Прочая документация COSMIC: руководство по использованию методологии в специфических предметных областях программного обеспечения, различные примеры применения метода, научные статьи, данные по производительности и т.д. Кроме того, доступны руководства по измерению и на других языках. Все это вы сможете найти на странице www.cosmicon.com.

Более общую информацию об измерениях функциональных размеров и их применении, о преимуществах метода COSMIC, об организациях COSMIC и их деятельности, поставщиках сервисов COSMIC, новостях и т.д. вы сможете найти на странице www.cosmicon.com.

Комитет по проведению измерений COSMIC.

СОДЕРЖАНИЕ

1

ВВЕДЕНИЕ

На программное обеспечение (далее ПО) тратится существенная часть бюджета большинства корпораций. Многие организации признают важность контроля таких расходов. Для анализа показателей качества и эффективности разрабатываемого и поддерживаемого ПО необходимы методики измерения ПО.

С одной стороны измерения функционального размера нужны разработчикам ПО для оценки собственной эффективности при создании продукта или предоставлении услуги. Результаты измерений могут быть использованы, например, для анализа эффективности или улучшения производительности проектных характеристик и т.д.

С другой стороны результаты измерений, выраженные в количественной форме, нужны, для того, чтобы оценить или измерить размер ПО, исходя из требований, предъявляемых на ранних стадиях проекта. Полученный размер затем используется как основной параметр для оценки трудозатрат на проект или для количественного определения эффективности продукта или услуги с точки зрения пользователя или владельца. Функциональные измерения должны быть независимыми от типа используемой технологии разработки или реализации. В таком случае их можно использовать для сопоставления значений производительности, полученных с использованием различных методик и технологий.

Метод функциональных точек (далее FPA)[2] является примером метода измерения функционального размера. Этот метод анализа можно применять для бизнес приложений, где он широко использовался для анализа и оценки продуктивности (Abran, 1996; Desharnais, 1988; Jones, 1996; Kemerer, 1987). С помощью этого метода можно успешно оценивать специфические функциональные характеристики программного обеспечения.

Однако метод функциональных точек неоднократно критиковался за то, что он неодинаково эффективен для разных видов ПО [Conte, 1986; Galea, 1995; Grady, 1992; Hetzel, 1993; Ince, 1991; Jones, 1988; Jones, 1991; Kan, 1993; Whitmire, 1992]. В частности, он не достаточно хорошо подходит для оценки ПО, работающего в режиме реального времени.

В 1997 году был предложен полный метод функциональных точек (далее FFP) (версия 1.0)[3], целью которого было расширение метода функциональных точек для того, чтобы обеспечить применимость метода и для ПО реального времени, а также технического и системного ПО. Тесты показали, что FFP также подходит для измерения функционального размера бизнес приложений и приводит к таким же результатам[4], что и при использовании метода FPA.

В 1998 году группа разработчиков метода FFP объединила свои усилия с разработчиками метода COSMIC[5], который предлагал принципы измерения функционального размера второго поколения. В результате появилась общедоступная методология измерения COSMIC-FFP версия 2.0 для предварительного тестирования, опубликованная в октябре 1999 года.

Начиная с версии 2.0, метод измерения функционального размера COSMIC совершенствовался, для обеспечения полного соответствия как стандарту ISO/IEC 14143-1: 1998 (а затем ISO/IEC 14143-1: 2007) так и принципам COSMIC.

Начиная с версии 3.0, название метода COSMIC-FFP было упрощено до COSMIC.

---

[2] Albrecht A.J., Gaffney Jr. J.E., «Software function, source lines of code and development effort prediction: a software science validation», IEEE Transactions on Software Engineering, Vol. SE-9, pp. 639-648, November 1983. FPA в настоящее время известен как метод IFPUG.

[3] St-Pierre D., Maya M., Abran A., Desharnais J.-M., Bourque P., «Full Function Points: Counting Practices Manual», Technical Report 1997-04, Université du Québec à Montréal, Montréal, Canada. Доступен по адресу: www.lrgl.uqam.ca/ffp.html

[4] Oligny, S.; Abran, A.; Desharnais, J.-M.; Morris, P., «Functional Size of Real-Time Software: Overview of Field Tests», in Proceedings of the 13th International Forum on COCOMO and Software Cost Modeling, Los Angeles, CA, October 1998.

[5] Для наиболее полной информации см. www.cosmicon.com.

О проекте COSMIC

Благодаря взрывному росту количества и разнообразия заказов и привлечения внешних ресурсов на разработку ПО, поставщикам и заказчикам потребуются более точные методы оценки и измерения эффективности выполнения этих работ. Такие методы оценки должны работать одинаково достоверно для всех типов ПО. Первое поколение методов измерения размера ПО не было достаточно строгим, чтобы соответствовать запросам рынка, или работало только со строго определенными типами ПО. Промышленности же были крайне необходимы более точные и более широко применимые методы измерения размеров программного обеспечения.

Группа COSMIC поставила перед собой цель удовлетворить нужды, прежде всего, поставщиков ПО, которые сталкиваются с задачей конвертации требований заказчиков в функциональный размер выпускаемого ПО, как ключевого шага в оценке стоимости проекта, а во вторую очередь - заказчиков, которым необходимо знать функциональные размеры поставляемого ПО для оценки эффективности работы поставщика.

COSMIC, the COmmon Software Measurement International Consortium (Международный Консорциум Измерений Типичного Программного Обеспечения) - международная добровольная инициатива по оценке проектов, объединяющая группу экспертов, как практиков, так и теоретиков, из стран Азии/Океании, Европы и Северной Америки. Цель проекта COSMIC - разработать, проверить на практике, вывести на рынок и найти применение новым методам измерения размера и производительности ПО. Эта цель в настоящее время достигнута, а методология принимается по всему миру во все большем количестве организаций как государственного, так и частного сектора.

После того, как в 1999 году были изложены принципы методологии COSMIC, в 2000/01 годах совместно с несколькими международными компаниями и научными учреждениями (академическими институтами) были проведены первые успешные испытания этого метода. Список статей, описывающих результаты этих испытаний, а также многие другие исследования опубликованы на сайте www.cosmicon.com. Процесс развития методологии COSMIC начался в 2001 году. В декабре 2002 года методология была одобрена на соответствие стандарту и опубликована Международной Организацией по Стандартизации (ISO) в начале 2003 как стандарт ISO/IEC 19761.

COSMIC продолжает уточнять определения и описания метода в свете опыта практического применения, хотя следует отметить, что та модель программного обеспечения, которая является основой для измерения размеров, не изменялась с момента ее первого опубликования в 1999 году. «Методология COSMIC 3.0. Руководство по измерению» - это самая полная версия с точки зрения изложения уточнений, которая по-прежнему совместима со стандартом ISO/IEC 19761. Обозначение «версия 3.0» по сравнению с «версией 2.2» указывает на то, что новая версия содержит действительно важные уточнения по сравнению с предыдущей версией. Для того чтобы получить наиболее полное представление об изменениях, произошедших при переработке версии 2.2. в версию 3.0., рекомендуем обратиться к документу «Методология COSMIC 3.0. Руководство по измерению».

Международный Консорциум Измерений Типичного Программного Обеспечения (COSMIC) планирует представить все дополнения и уточнения в Международную Организацию по Стандартизации (ISO) для включения в ISO/IEC 19761 по результатам его пересмотра в 2007/08.

В 2006 году COSMIC внедрил сертификацию начального уровня для специалистов-практиков методологии. Пользователям метода COSMIC предложили предоставить данные по производительности их проектов для базы данных «Международной Группы Стандартов по Испытаниям Программного Обеспечения» (ISBSG). Это было сделано для того, чтобы увеличить существующую базу данных по испытаниям ПО, связанных или измеренных с использованием методологии COSMIC.

Более подробную информацию о методологии COSMIC и публикациях по этой теме, деятельности в этой области и существующих экспертизах Вы сможете найти на сайте www.cosmicon.com. Информация о ISBSG доступна на сайте www.isbsg.org.

2

ОБЗОР МЕТОДОЛОГИИ ИЗМЕРЕНИЯ ФУНКЦИОНАЛЬНОГО РАЗМЕРА COSMIC

Методология COSMIC определяет стандартный подход к измерению функционального размера ПО. В этой главе описываются и обсуждаются следующие секции:

- типы ПО, для которых был разработан метод измерения функционального размера COSMIC (иными словами область применимости метода), секция 2.1
- обзор моделей ПО, используемых для измерения размера, секция 2.2. Эти модели представляют собой набор базовых понятий метода COSMIC. Важно правильно понимать эти базовые принципы, потому что для измерения функционального размера специалист должен установить соответствие между информацией из предоставленных ему материалов (например, технического задания или программного кода) и понятиями в моделях COSMIC.
- Обзор измерительного трехфазного процесса COSMIC:
  - Фаза выбора стратегии измерения, проводится до начала измерений (подсекция 2.3.1)
  - Фаза установления соответствия (подсекция 2.3.2)
  - Фаза измерения (подсекция 2.3.3)

Результатом процесса измерения является размер, выраженный в виде количества «Функциональных точек COSMIC» (далее ФТ). Перечисленные выше фазы проиллюстрированы на рис. 2.0.

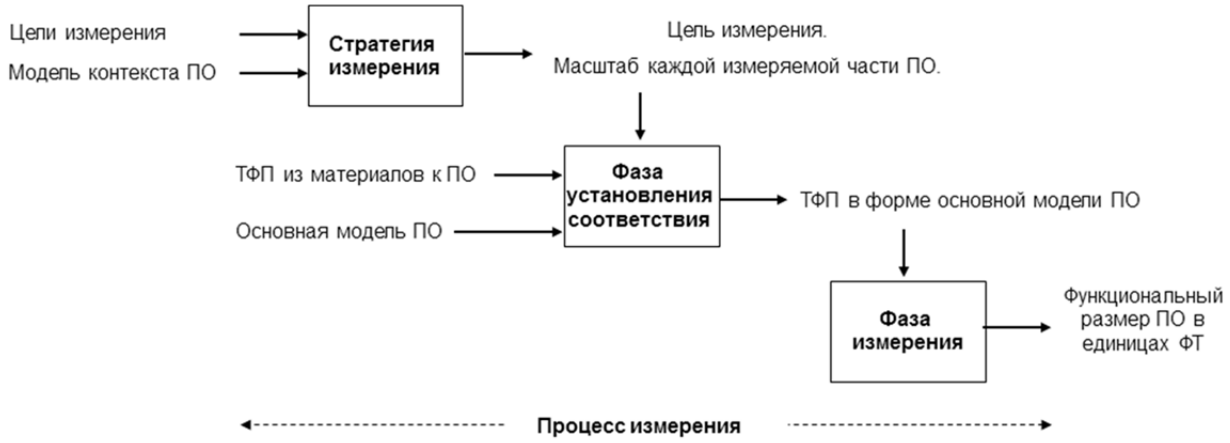

Рисунок 2.0 – Структура метода COSMIC

Обзоры подсекций 2.3.1., 2.3.2 и 2.3.3 и содержащиеся в них определения, принципы и правила более подробно изложены в главе «Руководства по измерению» и сопровождаются иллюстрированными примерами.

2.1 Применимость метода COSMIC

2.1.1 Области применимости

Метод COSMIC разработан для оценки функционального размера ПО в следующих областях:

- Бизнес-приложения, которые обычно используются как инструментарий в бизнес-процессах (банковское дело, страхование, бухгалтерский учет, кадры, продажи, логистика и производство). Такое ПО, как правило, работает с большим количеством данных, поскольку его разработка обычно мотивируется необходимостью эффективно обрабатывать большие объемы данных о событиях реального мира.
- ПО, работающее в режиме реального времени, задачами которого является поддержание или контроль событий, происходящих в реальном мире. В качестве примера можно привести

телефонные станции и коммутацию сообщений. ПО, встроенное в устройства, предназначенные для контроля техники (бытовые электроприборы, лифты, моторы автомобилей и самолетов), процессов и автоматической обработки данных. ПО, являющееся частью операционных систем компьютеров.

- Гибриды перечисленных выше приложений, например, системы бронирования в режиме реального времени для авиалиний или отелей.

2.1.2 Неприменимость

Метод измерения COSMIC не проектировался для оценки функционального размера приложений, использующих большой объем математических операций, т.е. такого ПО, которое характеризуется сложными математическими алгоритмами или другими специальными сложными правилами (например, систем с искусственным интеллектом, систем для компьютерного моделирования, самообучающегося ПО, метеорологических систем и т.д.), или ПО, занимающегося обработкой непрерывного потока данных (например, аудио- и видео-данных, в компьютерных играх, музыкальных инструментах и т.д.).

Также метод COSMIC неприменим для количественного измерения такой характеристики ПО, как «сложность» (вне зависимости от трактовки термина).

Однако в таких случаях можно определить локальные расширения метода COSMIC. В «Руководстве по измерению» объясняется, в каком контексте можно использовать такие локальные расширения и приведены примеры локальных расширений.

2.2 Модели программного обеспечения COSMIC

В этой части представлен обзор метода COSMIC и всех его основных понятий. Определения всех понятий приведены в документе «Метод COSMIC 3.0: Обзор документации и Глоссарий». В этой секции термин соответствующий модели ПО, употребляющийся в первый раз, будет выделен **жирным шрифтом**.

Необходимо понимать ВСЕ модели и базовые понятия COSMIC, которые будут описаны ниже. Кроме того, при проведении измерений необходимо уметь использовать ВСЕ принципы и правила, описанные в главах «Руководства по измерению». Только следуя описанному порядку процедур можно быть уверенным, что проведенные измерения значимы и воспроизводимы и/или могут быть сопоставлены с измерениями, проведенными независимо.

2.2.1. Требования функционального пользователя

Метод измерения COSMIC подразумевает применение группы моделей, принципов, правил и процессов к **Функциональным Требованиям Пользователя** (далее **ФТП**) относительно некоторой части ПО. Результатом является «количественное значение»[6] представляющее собой **функциональный размер** этой части ПО. Единицей измерения функционального размера является «**Функциональная точка COSMIC**» (ФТ).

Функциональный размер, полученный методом COSMIC, не зависит от технических решений, изложенных в документации по эксплуатации измеряемого ПО. Функциональность ПО связана с обработкой информации, которую ПО должно предоставить **пользователям**, а не с техническими методами реализации.

Таким образом, ФТП описывает функции, которые ПО должно исполнять для **функциональных пользователей**. Функциональные пользователи - это отправители и/или целевые получатели данных в соответствии с требуемой функциональностью. ФТП исключает любые технические требования или требования к качеству. При проведении измерений используются только функциональные требования (ФТП).

Выделение функциональных требований пользователя на практике

На практике редко можно найти материалы, явно выраженные в форме непосредственно пригодной для проведения расчета без необходимости интерпретировать информацию. Часто приходится выделять ФТП из доступных материалов, учитывая при этом и подразумеваемые требования, перед тем как проводить соответствие ФТП с понятиями моделей COSMIC.

[6] По определению Международной Организации по Стандартизации (ISO), см. документ «Методология COSMIC 3.0: Документация и Термины».

ФТП можно выделить из проектировочных материалов, которые издаются до создания ПО (техническое задание, результат анализа данных или функционала ПО на основе предъявляемых требований и пр.). Таким образом, функциональный размер ПО можно измерить еще до непосредственной реализации ПО.

Случается, что необходимо провести измерение конкретной части ПО без доступных материалов или с небольшим количеством материалов, описывающих архитектуру и дизайн ПО так, что ФТП не описаны (в том случае, если ПО уже разработано и используется). В таком случае ФТП можно выделить из материалов доступных пользователям ПО (представления экранов или отчетов, анализ потоков данных и т.п.).

Выделение функциональных требований пользователя из материалов, описывающих программное обеспечение

Процесс выделения ФТП из различных типов материалов, очевидно, зависит от типов используемых материалов. Методы и процессы составления списка ФТП непосредственно связаны с конкретной предметной областью ПО и могут сильно различаться. По этой причине привести их полноценное описание в рамках метода COSMIC практически невозможно.

В любом случае следует помнить, что ФТП существуют и могут быть получены из предоставленных материалов. Кроме того, COSMIC издает «методические руководства», в которых описаны некоторые аспекты нахождения ФТП[7] в конкретных предметных областях.

Метод COSMIC ограничивается описанием и определением понятий моделей COSMIC, то есть, целями и задачами выделения или получения ФТП. Эти понятия заложены в двух моделях COSMIC: «Модель контекста программного обеспечения», «Модель многофункционального программного обеспечения».

2.2.2. Модель контекста программного обеспечения

Конкретная часть ПО, которую нужно измерить, должна быть четко определена (с точки зрения масштаба измерения), и это определение должно учитывать окружение в виде другого программного и/или аппаратного обеспечения, с которым происходит взаимодействие рассматриваемой части ПО. Модель контекста ПО вводит принципы и понятия, необходимые для такого определения.

ПРИНЦИПЫ – модель контекста программного обеспечения COSMIC
а. ПО ограничено аппаратным обеспечением.
б. ПО обычно структурировано по **слоям**.
в. Слой может содержать одну или несколько отдельных частей ПО на одном уровне, а каждая из этих частей в свою очередь может также состоять из отдельных **компонентов, расположенных в одном слое**.
г. Для любой части ПО, подлежащей измерению, должны быть определены рамки измерения, которые в свою очередь должны относится к одному слою.
д. Рамки измерения зависят от **цели измерения**.
е. Функциональные пользователи это все отправители и/или целевые получатели данных.
ж. ПО взаимодействует с функциональными пользователями за счет **перемещения данных** через **границу.** Также ПО может обмениваться данными с **постоянным хранилищем данных** внутри своих границ.
з. ФТП могут быть выражены с разными уровнями детализации.
и. Уровень детализации, при котором обычно следует проводить измерение должен соответствовать уровню **функциональных процессов** (см. секция 2.2.3).
к. Если невозможно провести измерение на уровне функциональных процессов, тогда ФТП следует измерять приближенным способом и масштабировать в соответствии с уровнем детализации функциональных процессов[8].

[7] Вопрос перехода между разными уровнями детализации разобран в документе «Методология COSMIC 3.0. Более сложные задачи и связанные с этим проблемы».

[8] Вопрос перехода между разными уровнями детализации разобран в документе «Методология COSMIC 3.0. Более сложные задачи и связанные с этим проблемы».

Рассмотрим подробнее изложенные принципы и понятия, а также проиллюстрируем их простыми примерами.

Для начала нужно отличать два способа представления компьютерной программной/аппаратной системы, которая описывает контекст измеряемого ПО:

- Физическое представление. Оно показывает, как на практике ПО структурировано по слоям, каждый из которых соответствует своей специфичной функции. Такое представление показывает, как происходит взаимодействие частей ПО за счет аппаратных средств и, возможно, за счет других промежуточных слоев ПО.
- Логическое представление является абстракцией физического представления и применяется для измерения функционального размера. Такое представление показывает, что функциональные пользователи ПО (отправители и/или целевые получатели данных) взаимодействуют с ПО через границу, и что ПО отвечает за обмен данными с постоянным хранилищем данных. На этом уровне абстракции все аппаратные и программные средства, обеспечивающие такие взаимодействия, во внимание не принимаются.

Несмотря на то, что только второе представление необходимо для измерения функционального размера, полезно рассмотреть оба подхода, т.к. необходимо уметь различать их. Более того нужно понимать что, метод COSMIC очень специфично использует такие термины, как «слой» и «компонент». (Эти термины используются в разных смыслах в области информационных технологий.)

На рисунке 2.2.2.1 показано физическое представление типичного бизнеса приложения в контексте таких слоев ПО, как операционная система, драйверы устройств и т.п., и слоев аппаратного обеспечения. На рисунке 2.2.2.2 в качестве примера показано такое же физическое представление ПО, работающего в режиме реального времени.

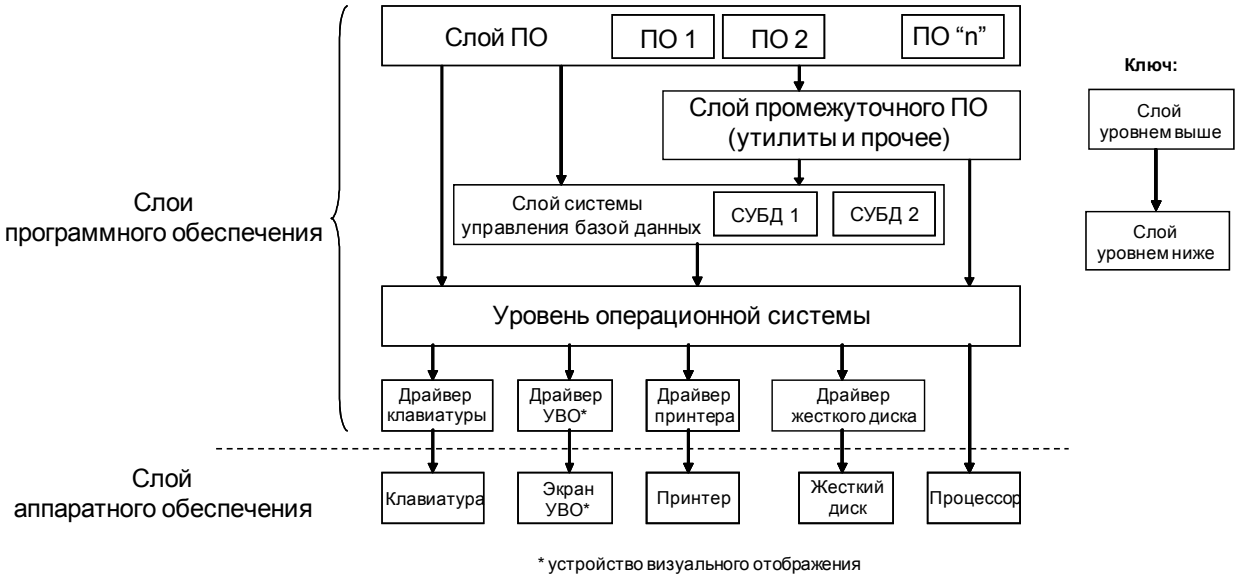

Рисунок 2.2.2.1 – Типичная архитектура ПО для бизнеса

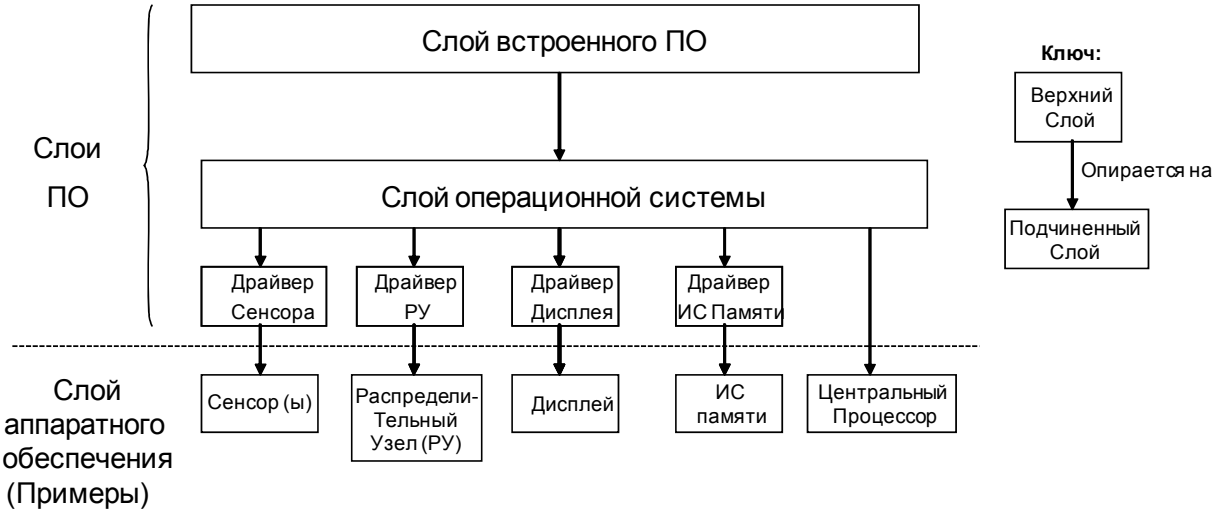

Рисунок 2.2.2.2 – Типичная послойная архитектура для системы со встроенным ПО, работающим в режиме реального времени

Принцип (a)

ПО, используемое пользователем, ограничено аппаратными устройствами ввода-вывода (мышь, клавиатура, принтер или экран); ПО, работающее в режиме реального времени, обычно ограничено инженерными устройствами такими, как сенсоры или реле. ПО также ограничено устройствами постоянного хранения информации (жесткий диск) или другими типами памяти, которые можно использовать для хранения информации.

Принцип (b)

В том случае, если измеряемая часть ПО является частью, разработанной и разбитой по слоям архитектуры, определить слой, на котором данная часть расположена, достаточно просто. Однако если окружение ПО со временем выросло и развилось, может оказаться, что слои трудно разделить. В этом случае, стоит использовать некоторые правила из метода COSMIC для разграничения слоев.

Принцип (c)

Например, независимые ключевые компоненты бизнес приложений (в случае трехзвенной архитектуры: пользовательский интерфейс, бизнес логика, управление данными) нужно считать относящимися к одному слою. Проводить измерение размера таких компонентов можно независимо, и в методе COSMIC существуют правила для такого измерения. На практике если основные компоненты конкретной части ПО выполнены на разных технических платформах, очень важно иметь возможность проводить измерения их размера независимо, что необходимо для быстрого и эффективного измерения и оценки.

Любая часть ПО может быть разложена на субкомпоненты на разных уровнях декомпозиции (например, вплоть до уровней отдельных модулей или объектов класса). В таком случае для измерения функционального размера можно использовать метод COSMIC на любом из таких уровней. Однако при разделении необходимо определить единый подход для разделения ПО на субкомпоненты, чтобы обеспечить сопоставимость результатов измерений.

Принцип (d)

Рамки измерений функционального размера должны полностью находиться в одном слое. Дело в том, что каждый слой имеет специализированную функцию и может быть разработан с применением отличных от других слоев технологий.

При необходимости следует проводить измерение размера предварительно разделив ПО на несколько слоев, а затем объединить результат измерения каждого слоя. Однако в таком случае объяснение, понимание и сравнение итогового размера может оказаться затруднительным. Поэтому существуют правила объединения размеров компонентов из разных слоев измеряемой части ПО, которые приведены в секции объединения результатов измерения в «Руководстве по измерению».

Принцип (e)

Предположим, что приложения на рисунках 2.2.2.1 и 2.2.2.2 представляют собой отдельные ПО, каждая из которых разработана независимой проектной группой. В таком случае для большинства типичных целей измерений имеет смысл определить масштаб измерения как «приложение целиком».

Однако если какой-либо приложение разработано в трехзвенной архитектуре (как указано в принципе (c) выше), каждый на основе отличающейся технологии, и/или разработанных разными проектными командами, тогда, если целью измерения является оценка трудозатрат, имеет смысл оценивать размер каждого компонента в отдельности. Результаты измерения можно затем использовать в качестве данных для ввода в формулу оценки трудозатрат, которая должна учитывать различные технологии разработки и/или характеристики разных проектных команд по каждому из указанных компонентов.

Принцип (f)

Чтобы проиллюстрировать принципы (f) и (g) следует рассмотреть логическое представление ПО. На рисунках 2.2.2.3 и 2.2.2.4 показано логическое представление, где отмечены взаимодействия функциональных пользователей с ПО (для бизнес приложения и для приложения реального времени соответственно).

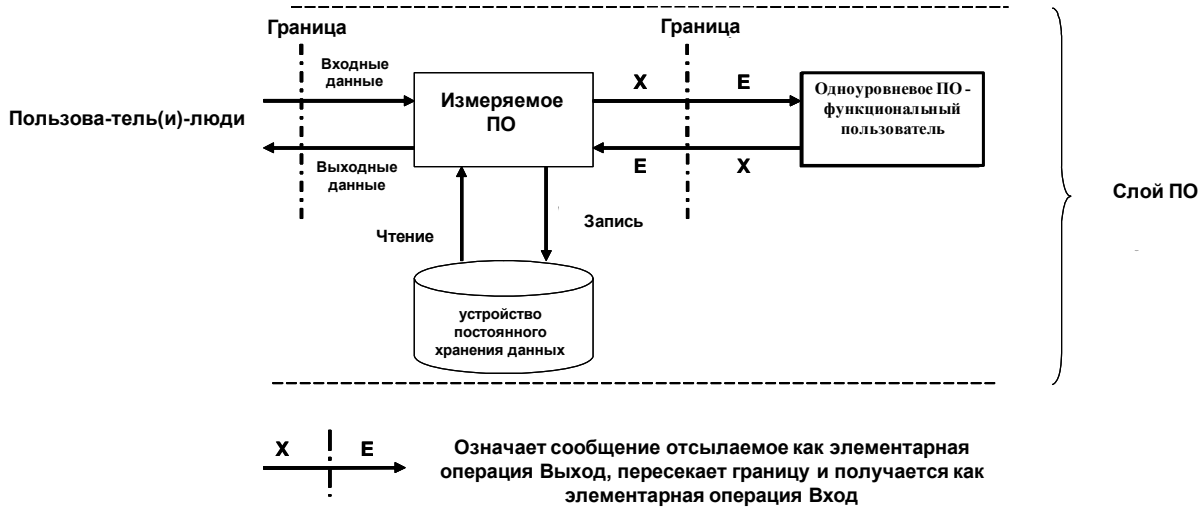

Рисунок 2.2.2.3 – Бизнес-приложение, у которого два функциональных пользователя: человек, и одноуровневое ПО (логическое представление)

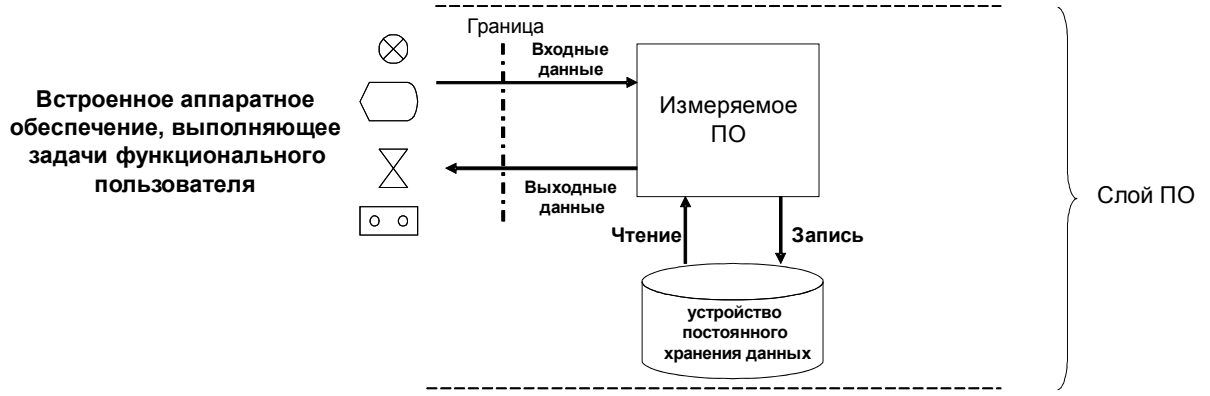

Figure 2.2.2.4 – Встроенное ПО, работающее в режиме реального времени, функциональными пользователями, которого являются различные аппаратные средства (логическое представление)

Рассмотрим пример бизнес приложения. На рисунке 2.2.2.1 показано, что «пользователями приложения» может быть операционная система, любые аппаратные устройства (например, клавиатура, принтер и подобные), а также пользователь-человек – про них можно сказать, что они все взаимодействуют с приложением непосредственно или косвенно. Но не все указанные типы пользователей подходят под определение функционального пользователя, как отправителя данных и/или целевого получателя данных. Операционная система и аппаратные устройства

обеспечивают такой обмен данными, но не являются отправителями и/или целевыми получателями данных.

Для бизнес приложений ФТП обычно описывают требуемую функциональность с точки зрения пользователя-человека и, возможно, других приложений, которые отправляют в и/или получают данные из данного приложения. Следовательно, указанные пользователи и интегрированные приложения будут являться функциональными пользователями данного приложения, что и показано на рисунке 2.2.2.3.

Благодаря строгому разделению функциональности на слои, как показано на рисунке 2.2.2.1, при определении функционального размера обычно можно игнорировать любые программные слои или аппаратные средства, такие как, например, операционная система или экран, которые обеспечивают взаимодействие функционального пользователя с приложением. Как правило, при определении функциональных пользователей возникает никаких сомнений.

Для примера для встроенного ПО, работающего в режиме реального времени, ФТП обычно будут описывать функциональность необходимую с точки зрения аппаратных средств (сенсоров, клапанов и подобных устройств), которые приложение должно поддерживать. Следовательно, эти устройства будут функциональными пользователями, как показано на рисунке 2.2.2.4. Однако в некоторых случаях, как показано в «Руководстве по измерению», ФТП некоторых типов ПО могут быть написаны для нескольких типов функциональных пользователей и поэтому описывать разную функциональность и, как следствие, - разный функциональный размер[9].

Принцип (g)

На рисунках 2.2.2.3 и 2.2.2.4 показано, что функциональные пользователи взаимодействуют с ПО через границу посредством двух типов перемещений данных (**Вход** и **Выход**). ПО также обменивается данными с устройством постоянного хранения данных посредством дополнительно двух других типов перемещений данных (**Чтение** и **Запись**). Указанные типы перемещений данных будут определены далее, секция 2.2.3.

Граница определяется как «концептуальный интерфейс между ПО и его функциональными пользователями».

Граница позволяет провести четкое различие между чем-либо, что является частью измеряемого ПО (например, то, что находится на стороне ПО относительно проведенной границы) и чем-либо, что является частью из окружения функциональных пользователей (например, то, что находится на стороне функциональных пользователей относительно проведенной границы). Постоянное хранилище данных не рассматривается в качестве функционального пользователя ПО и, следовательно, находится на стороне ПО относительно проведенной границы.

На рисунке 2.2.2.5 показано логическое представление некоторого бизнес-приложения, которое было разработано в трехзвенной архитектуре, как описано выше в принципах (c) и (e). Предполагая, что компоненты были разработаны с использованием различных технологий, и цель измерений диктует необходимость оценить размер каждого отдельного компонента, граница должны быть проведены для каждого компонента отдельно.

[9] Исключением, когда операционная система может быть функциональным пользователем, является случай, когда предъявляется требование к операционной системе предоставить, например, сигнал времени для запуска функционального процесса.

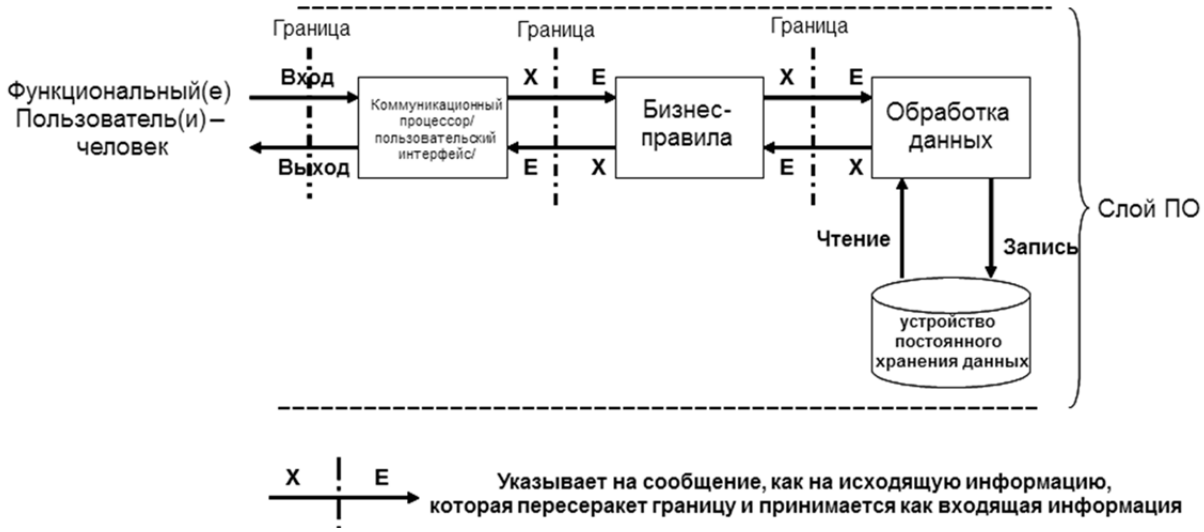

Рисунок 2.2.2.5 – Пример бизнес-приложения, когда компоненты должны измеряться отдельно (логическое представление)

В логическом представлении на рисунке 2.2.2.5 показано, что функциональным пользователями компонента «пользовательский интерфейс» являются пользователь-человек и компонент «бизнес-правил». Каждый из функциональных пользователей взаимодействует с ним посредством Входов и Выходов через границу. Из рисунка также видно, что только компонент управления данными взаимодействует с постоянным хранилищем данных, и что его функциональным пользователем является только компонент бизнес-правил. В таком логическом представлении ФТП для каждого из компонентов можно измерять отдельно.

Принцип (h) и (i)

Функциональные требования к ПО могут выражаться на разном **уровне детализации**. Понятие **уровня детализации** связано с количеством деталей в описании части ПО. Это понятие нужно отличать от **уровня декомпозиции**, который связан с делением ПО на компоненты. Уровень детализации для проведения измерения должен находится на уровне функциональных процессов (секция 2.2.3).

В начале процесса разработки нового ПО процесс определения требований обычно начинается с создания и одобрения общего технического задания, которое потом дорабатывается и уточняется. Следовательно, ФТП могут существовать на разных уровнях детализации.

При использовании **функционального анализа** (метода сбора требований к ПО) наиболее типичной является иерархия уровней. На первом уровне определяется основная функция. При детализации этот уровень распадается на несколько функций уровня 2, каждая из которых в свою очередь состоит из функций уровня 3 и т.д. На определенном уровне этой иерархии детализация остановится на индивидуальных **функциональных процессах**. В секции 2.2.3 об этом рассказано подробнее.

По мере увеличения детализации увеличивается масштаб измерений, что, в свою очередь, приводит к увеличению измеренного функционального размера. (Следует отметить, что это явление отличается от эффекта «расползания масштаба», в котором размер увеличивается из-за увеличения масштаба ПО.)

Следовательно, для того, чтобы иметь возможность сравнивать результаты измерения из различных источников, все измерения должны быть соотнесены (или масштабированы) со стандартным уровнем детализации, т.е., с «детализацией на уровне функциональных процессов». В большинстве случаев, при измерении функционального размера полностью специфицированного или уже существующего ПО, измерения с детализацией на уровне функциональных процессов является очевидным.

Каждый знаком с тем, как проводится измерение расстояний на карте при использовании различных масштабов, например, когда в 1 см или 1 мм карты умещается 1 км, а также с переводом расстояний из одного масштаба в другой. При измерении функционального размера конкретной части ПО методом COSMIC используют только один стандарт – «детализацию на

уровне функциональных процессов» и только одну единицу измерения. Таким образом, если необходимо сравнить результаты измерений, проведенных на разных уровнях детализации, мы должны изобрести свои собственные локальные коэффициенты пересчета для перевода полученных нами размеров до условных единиц «детализации на уровне функциональных процессов». Эти понятия разобраны более подробно в секции, посвященной стандартному уровню детализации в «Руководстве по измерению».

Принцип (j)

Необходимость измерения размера ПО на уровне детализации более общем, чем уровень функциональных процессов, обычно возникает на ранних стадиях разработки нового ПО. При этом требования, предъявляемые к ПО, постоянно изменяются.

Если невозможно провести измерение на уровне детализации функциональных процессов, то измерения следует проводить на уровне детализации максимально приближенному к уровню функциональных процессов, а затем масштабировать измерения до стандартного уровня (глава о проведении измерений на ранних стадиях в «Методология COSMIC 3.0. Более сложные задачи и связанные с этим проблемы»).

Таким образом, для упрощения процесса измерения ФТП модель контекста ПО в соответствии с методологией измерения функционального размера COSMIC предоставляет набор таких понятий и принципов, как слой ПО, одноуровневые компоненты, охват измерений, перемещения данных и граница конкретной части ПО, которые могут быть изложены на разных уровнях детализации. В процессе выбора стратегии измерения (секция 2.2.4) рекомендуется применять эти понятия и принципы для того, чтобы определить какие измерения необходимы и как их следует интерпретировать.

2.2.3 Основная модель программного обеспечения в методе COSMIC

После интерпретации функциональных требований в понятиях модели контекста составляется Основную Модель ПО, которая определяет компоненты функциональности, которые будут измерены. Основная Модель ПО подразумевает, что указанные ниже принципы верны для любых типов ПО, которые можно измерить методом COSMIC. (Все необходимые определения и термины указаны в словаре [10]).

ПРИНЦИПЫ – Основная модель ПО в методе COSMIC
a. ПО получает **входящую информацию** от функциональных пользователей и **выводит результат** функциональным пользователям.
b. На основе функциональных требований к ПО можно выделить уникальные функциональные процессы.
c. Каждый функциональный процесс состоит из процессов более низкого уровня (**подпроцессов**).
d. Подпроцессом может быть либо **перемещение**, либо **обработка** данных**.**
e. Каждый функциональный процесс инициируется **Входом** со стороны функционального пользователя, который таким образом передает информацию функциональному процессу о том, что функциональный пользователь обнаружил некоторое **событие.**
f. Операция перемещения перемещает одну **группу данных**
g. Группа данных состоит из уникального набора **атрибутов данных**, которые описывают один **объект.**
h. Выделяют четыре типа перемещений данных. Вход перемещает группу данных от функционального пользователя в ПО. **Выход** перемещает группу данных функциональному пользователю из ПО. **Запись** перемещает группу данных из ПО в постоянное хранилище данных. **Чтение** перемещает группу данных из постоянного хранилища данных в ПО.
i. В функциональный процесс следует включать, по крайней мере, один Вход и либо Запись, либо Выход, таким образом, получается не менее двух перемещений данных.

[10] Как указано в справочнике терминов, целью любого метода измерения функционального размера ПО является определение типов, а не частота употребления данных или функций. Далее по тексту при изложении базовых понятий COSMIC слово «тип» употребляться не будет, если только разделение «типа» и «частоты» не окажется необходимым.

ПРИНЦИПЫ – Основная модель ПО в методе COSMIC
j. В качестве упрощения для целей измерения, подпроцессы обработки данных, не измеряются отдельно. Полагается, что функциональность любой обработки данных включена в перемещение данных, с которым она связана.

Принципы (a) – (e)

Задачей ПО является реагирование на события, которые происходят *на стороне функционального пользователя*, другими словами, в мире функциональных пользователей. Функциональный пользователь уведомляет ПО о начале события и/или отправляет данные о событии. ПО должно сделать что-то полезное для функционального пользователя в качестве ответа на событие. «Что-то полезное» мы называем «функциональным процессом». Таким образом, все ФТП к ПО могут быть выражены в виде листа событий и соответствующих им функциональных процессов, которые предоставляют ответ ПО на каждое событие.

Принципы (c) и (d) говорят о том, что функциональный процесс можно представить в виде двух типов подпроцессов, а именно: перемещение и обработка данных. ПО, в свою очередь, способно только перемещать и/или обрабатывать данные.

На рисунке 2.2.3.1 ниже, проиллюстрированы принципыs (б) - (г) основной модели ПО.

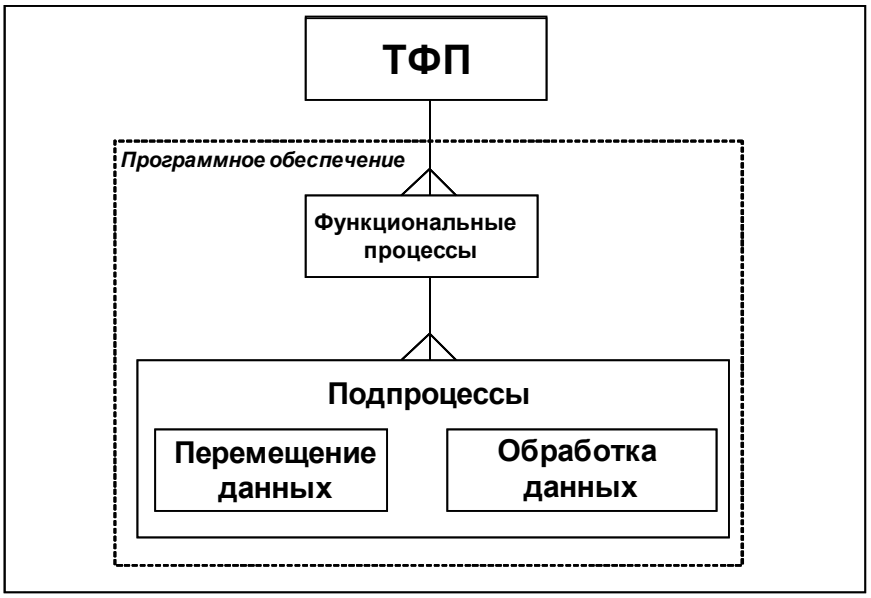

Рисунок 2.2.3.1 – Структура функционального пользователя

Принципы (f) и (g)

Каждое перемещение данных затрагивает только одну группы данных, которая представляет собой информацию об одном объекте, другими словами «предмет интереса» функционального пользователя. Примером из области бизнес-приложений является относительно простой функциональный процесс формирования счета, который обычно включает следующие перемещения данных (объекты в пунктах ниже представлены в кавычках):

- Два Входа для групп данных о «счете» и «позиции в счете», принимая во внимание, что счет предполагает несколько позиций товара. Первый Вход, связанный с объектом «счет», осуществляет вызов рассматриваемого функционального процесса.
- Два Чтения групп данных о «покупателе» и «продукте» для проверки, разрешено ли данному покупателю делать заказ, и существуют ли запрашиваемые наименования продуктов, и есть ли они в наличии.
- Две Записи групп данных о «счете» и «позиции в счете», чтобы записать информацию в постоянное хранилище данных.

- Один или несколько Выходов групп данных, содержащих, например, сообщение о «подтверждении счета», включая итог по «счету», инструкцию для склада на сбор каждого «наименования счета» и прочее.

Все указанные объекты представляют собой реальные или концептуальные предметы из реального мира функциональных пользователей. ПО обрабатывает информацию об этих предметах (объектов интереса). При измерении функционального размера необходимо правильно определить объекты интереса для того, чтобы определить количество перемещений данных.

При измерении ПО, работающего в режиме реального времени, или встроенного ПО, работают в точности те же принципы, хотя очень часто функциональный пользователь и объект на практике фактически неразличимы. Например, предположим, что функциональному процессу требуется получить текущую температуру с датчика, который устроен так, что непосредственно способен осуществлять связь с ПО. Тогда датчик является функциональным пользователем конкретной части ПО. В таком случае, датчик осуществляет перемещение данных - «Вход», вероятно, только с двумя значениями атрибутов (ID датчика и температура). Два указанных значения содержат информацию о датчике (как объекте), и тогда есть основания полагать, что объектом является тот предмет, чью температуру измеряет датчик.

Теперь рассмотрим простой пример функционального процесса для измерения температуры с помощью датчика и с контролем температуры относительно заданной. Этот функциональный процесс вызывается через регулярные промежутки времени сигналом от часов, и состоит из следующих перемещений данных:

- Вход сигнала от часов, который запускает функциональный процесс
- Вход информации датчика (ID датчика, значение текущей температуры)
- Чтение заданной температуры из постоянного хранилища данных (в предположении, что температура может быть задана и изменена другим функциональным процессом)
- Выход, содержащий сигнал на включение или выключение нагревательного элемента, в том случае если есть необходимость изменить состояние нагревательного элемента

Следует отметить, что в этом примере все группы данных кроме одной содержат только один атрибут данных.

Принцип (h)

Этот принцип говорит о том, что четыре типа перемещений данных различаются по своему источнику и целевому назначению. Перемещение данных либо пересекает границу между измеряемым ПО и его функциональным пользователем (Вход, Выход), или перемещает данные между ПО и постоянным хранилищем данных (Чтение, Запись), как показано на рисунке 2.2.3.2 ниже.

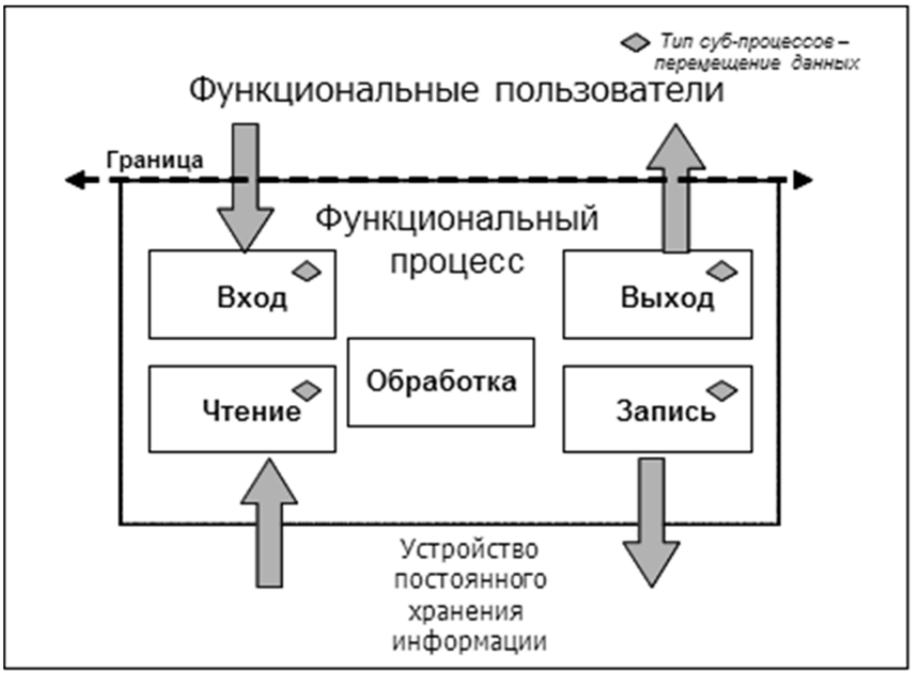

Рисунок 2.2.3.2 – Составляющие части функционального процесса и некоторые их взаимосвязи

Принцип (i)

Этот принцип утверждает, что функциональный процесс подразумевает, по крайней мере два перемещения данных (следует из описанных выше принципов). Функциональный процесс, который осуществляет только одно перемещение данных и никак их не обрабатывает, будет бесполезным с практической точки зрения. Значит, все функциональные процессы должны содержать, по крайней мере, одно перемещение данных с информацией о начале события (Вход), и, по крайней мере, еще одно перемещение данных, как ответ (полезный результат) либо функциональному пользователю (Выход), либо в постоянное хранилище данных (Запись).

Принцип (к)

Метод COSMIC предполагает упрощение основной модели ПО с учетом предметных областей, для которого метод COSMIC был разработан.

В качестве первого приближения в этой версии метода измерения, подпроцессы, связанные с обработкой данных, как показано на рисунке 2.2.3.2, не выделяются отдельно и связаны или являются частью подпроцессов перемещения данных. (С этого момента и далее, для краткости, будет использоваться словосочетание «перемещение данных» вместо «подпроцесс перемещения данных»). Такое приближение используется потому, что понятия и определения, необходимые для измерения процессов обработки данных до сих пор являются предметом обсуждения среди разработчиков ПО. Особенности рассмотрения функционала обработки данных как части перемещения данных описаны в подсекции, посвященной обработке данных в «Руководстве по измерению».

Выбор такого приближения объясняет, почему стандартный метод COSMIC подходит для определения размера ПО, исполняющего перемещения данных, как в случае бизнес-приложений, так и приложений, работающих в режиме реального времени, а также объясняет почему этот метод не подходит для определения размера ПО, исполняющего другие типы обработки данных (или алгоритмов). В «Руководстве по измерению» также отмечается, что необходимо с осторожностью подходить к измерению малых частей ПО и, особенно, малых изменений в ПО, поскольку в таких случаях предположения из принципа (к) могут не выполняться.

«Руководство по измерению» также описывает механизм определения локального расширения к методу COSMIC, которое позволяет явно учитывать функционал, связанный с обработкой данных. Для таких случаев, определены специальные соглашения об отчетности.

Используя понятия и их определения, принципы и правила методологии измерения функционального размера COSMIC, можно установить соответствие между ФТП, полученными из

материалов по программного обеспечению, и основной моделью ПО, т.е. описать ее. Описанная модель будет содержать все элементы, необходимые для проведения измерения ее функционального размера, и не будет учитывать нерелевантную для функционального пользователя информацию.

После такой процедуры к описанной модели применяют правила и процессы измерения и получают численное значение функционального размера конкретной части ПО (правила измерения указаны в секции 2.2.3 ниже).

2.3 Обзор процесса измерения методом COSMIC

При проведении измерений функционального размера ПО необходимо руководствоваться последовательностью трех связанных стадий:

1. Выбор стратегии измерения с использованием принципов модели контекста ПО;
2. Установление соответствий между доступными материалами к измеряемому ПО и основной моделью ПО;
3. Измерение определенных элементов этой модели.

2.3.1 Стадия выбора стратегии измерения

Прежде чем начинать измерение необходимо согласовать с лицами заинтересованными в измерении и задокументировать:

(а) цель измерения;

(б) границы измеряемого ПО;

(в) функциональных пользователей ПО и, как следствие, границы каждой части ПО;

(г) уровень детализации, с которым требуется провести измерение.

Цель измерения - критически важный элемент выбора стратегии, т.к. она определяет три других параметра (б), (в) и (г). При этом зачастую определение указанных трех параметров – является итеративным процессом.

(а) Цель измерения

Цель измерения устанавливает причины проведения измерения и назначение результатов. Это, в свою очередь, помогает определить не только остальные три параметра стратегии измерения, но и, например, требуемую точность измерения.

Как показано в главе «Методологии COSMIC 3.0. Более сложные задачи и связанные с этим проблемы», посвященной измерению ПО на ранней стадии, можно оценить функциональный размер ПО с помощью приближенного варианта исполнения стандартного метода COSMIC. Приближенный вариант исполнения можно применять на ранних стадиях проекта до того, как все требования будут установлены на уровне детализации достаточном для проведения измерения функционального размера в точном соответствии с правилами метода.

(б) Границы измеряемого ПО

Цель измерения определяет общие границы измеряемого ПО, что, в свою очередь, определяет какой функционал будет включен в измерение, а какой будет исключен. В зависимости от цели, может потребоваться разделить общие границы на несколько функциональных частей ПО, каждая из которых будет измерена отдельно, каждая со своими границами.

Такое деление границ необходимо если общие границы включает ПО расположенное более чем в одном слое, потому что любая часть измеряемого ПО должна оставаться полностью внутри одного слоя.

Во-вторых, деление может оказаться необходимым, например, если цель измерения связана с оценкой трудозатрат, а измеряемое ПО представляет собой отдельные компоненты, которые будут разрабатываться на основе различных техник и/или технологий и/или будут исполняться на различных платформах и/или будут разрабатываться разными командами. В таком случае каждый из компонентов должен иметь свой собственные границы измерений.

(в) Функциональные пользователи и границы каждой части измеряемого ПО

Изучение входящих и исходящих потоков данных позволяет определить функциональных пользователей для каждой из частей ПО. Функциональные пользователи являются отправителями или целевыми получателями данных.

В большинстве случаев функциональные пользователи становятся очевидны после определения цели измерения и ФТП. За редким исключением функциональные пользователи могут зависеть от целей измерения. Пример, приведен в подсекции, посвященной функциональным пользователям в «Руководстве по измерению».

Когда функциональные пользователи известны, границу – абстрактную границу между функциональными пользователями и измеряемой частью ПО – можно легко установить.

(г) Уровень детализации измерений

Обычно уровень детализации материалов для проведения измерений должен позволять выделить функциональные процессы, включая их деления на перемещения данных.

Когда целью измерений является определение размера функциональных требований существующего ПО, то после выделения функциональных процессов, уровень детализации становится очевиден.

С другой стороны, на ранних стадиях разработки, измерение можно провести до того, как отдельные функциональные процессы и связанные с ними перемещения данных были четко определены.

Еще более сложный случай измерений, когда функциональные требования описаны на разных уровнях детализации на момент необходимости проведения измерений. В таком случае нужно определить некоторые локальные «функциональные единицы» на основе доступных материалов ПО, разработать подход для масштабирования уровня детализации, на котором размер измеряется подсчетом локальных функциональных единиц, до уровня детализации, на котором можно выделить и измерить функциональные процессы (например, в функциональных точках COSMIC). Такие методы масштабирования обсуждаются в секции определения стандартного уровня детализации в документах: «Руководство по измерению» и в «Более сложные задачи и связанные с этим проблемы».

После завершения шагов (а) – (г) может понадобиться повторное прохождение этих пунктов. Например, из-за того, что какие-то требования раскрываются при детальном установлении стратегии измерений, а это может привести к необходимости уточнения масштаба измерения части ПО.

Полное обсуждение с определениями и примерами целей, масштабов, функциональных пользователей и уровней детализации приведено в главе, посвященной стратегии измерения в «Руководстве по измерению».

2.3.2 Стадия установления соответствия

На этой стадии используются ФТП, полученные из материалов к ПО[11], с учетом модели контекста ПО. Для получения основной модели ПО на стадии установления соответствия необходимо выполнить следующие шаги:

- определить события в мире функциональных пользователей, на которые ПО должно реагировать, т.е. определить функциональные процессы;
- определить перемещения данных (Вход, Выход, Чтение, Запись) для каждого функционального процесса, который в свою очередь зависит от определения групп данных, которые перемещаются.

Полное обсуждение с определениями и примерами понятий и шагов на стадии установления соответствия приведено в соответствующей главе «Руководства по измерению».

2.3.3 Стадия измерений

В начале стадии измерения используют основную модель ПО и, применяя строго определенные правила и процессы, определяют численное значение, величина которого прямо пропорциональна функциональному размеру модели, в соответствии со следующим принципом:

[11] В том виде, в котором эти материалы найдены, или в том виде, в котором создавались внутри организации, или в том виде, в котором они представлены в самой программе.

ПРИНЦИП – принцип измерения методом COSMIC
Функциональный размер некоторой части ПО прямо пропорционален числу перемещений данных в этой части.

Характеристики набора правил и процессов определяющие определение размера в численном виде:

Характеристика 1 – единица измерения

Стандарт измерения, а именно 1 ФТ (функциональная точка COSMIC), определена соглашением как эквивалент одного перемещения данных.

Характеристика 2 – аддитивность размеров в рамках границ измерения

Функциональный размер функционального процесса определен как арифметическая сумма числа составляющих его перемещений данных. Более развернуто эта формулировка выглядит так: функциональный размер любой части ПО в рамках заданных границ измерения[12] представляет собой арифметическую сумму функциональных размеров функциональных процессов этой части ПО.

Характеристика 3 – размер изменений в конкретной части ПО

Функциональный размер любого требуемого изменения в конкретной части ПО является арифметической суммой числа перемещений данных, которые должны быть добавлены, изменены и удалены по причине вносимых изменений.

Характеристика 4 – минимальный и максимальный размер функционального процесса

Согласно приведенным характеристикам и принципу (g) основной модели ПО, минимальный размер функционального процесса – 2 ФТ, потому что наименьший функциональный процесс должен иметь, по крайней мере, один «Вход» (на входе в процесс) и либо один «Выход» (как результат процесса), либо одну «Запись» (как альтернативный полезный результат).

Из того, что изменение может затронуть только одно перемещение данных, следует, что минимальный размер вносимого изменения – 1 ФТ.

В соответствии с указанными характеристиками, не существует верхнего предела функционального размера функционального процесса и, следовательно, не существует верхнего предела функционального размера любой части ПО.

Дополнительные принципы и детально разобранные правила и процессы стадии измерения функционально размера на основе ФТП к конкретной части ПО, выраженных в основной модели ПО, приведены в соответствующих главах «Руководства по измерению», некоторые выводы и итоги представлены в Приложениях Б и В.

[12] Правила об объединении размеров частей ПО с разными масштабами, см. главу «Руководства по измерению», стадия измерения.

ПРИЛОЖЕНИЕ А – COSMIC ОТПРАВКА КОММЕНТАРИЕВ И ЗАЯВОК НА ВНЕСЕНИЕ ИЗМЕНЕНИЙ

Комитет по проведению измерений COSMIC (COSMIC КПИ) особенно заинтересован в получении обратной связи, комментариев и, при необходимости внесении изменений в «Руководство по измерению» COSMIC. В данном Приложении указано как связаться с COSMIC КПИ.

Все обращения в COSMIC КПИ следует направлять на следующий электронный адрес:

mpc-chair@cosmicon.com

Неформальная обратная связь и комментарии

Неформальные комментарии и/или любую информацию, касающуюся «Руководства по измерению», такую как трудности в понимании или применении метода COSMIC, общие предложения по улучшению, и т.п., следует отсылать на указанный выше электронный адрес. Сообщения будут зарегистрированы и приняты к рассмотрению в течение двух недель с момента получения. COSMIC КПИ не гарантирует обязательного ответа на сообщения общего характера.

Формальный запрос на изменения

Если, по мнению читателя «Руководства по измерению», в тексте обнаружилась ошибка, или требуется уточнение, или существует необходимость улучшения текста, можно направить формальный запрос. Формальные запросы на изменение текста регистрируются и принимаются к рассмотрению в течение двух недель с момента их получения. Каждому запросу на изменение присваивают серийный номер, и направляют членам COSMIC КПИ, представляющим группу международных экспертов по методу COSMIC. Период рассмотрения запроса экспертами составляет минимум 1 месяц и может занять большее количество времени, если запрос представляет серьезное затруднение. После рассмотрения запроса изменение может быть принято либо отклонено, либо ему будет присвоен статус «ожидает дальнейшего рассмотрения» (последний вариант реализуется, например, если есть зависимость от других запросов), а результат будет сообщен отправителю так скоро, насколько это будет возможно.

Формальный запрос обрабатывается только в том случае, если в нем будет указана следующая информация:

- Имя, должность и место работы человека, отправляющего запрос;
- Контактные данные человека, отправляющего запрос;
- Дата запроса;
- Общая постановка цели запроса на изменение;
- Конкретный отрывок текста, требующий изменения, замены или удаления (или четкая ссылка на него);
- Предлагаемый дополнительный или замещающий текст;
- Полное разъяснение, почему изменение необходимо.

Форма для отправки запроса на изменение доступна на сайте: www.cosmicon.com.

Если решение по запросу на исправление после рассмотрения членами COSMIC КПИ будет положительным, в последней версии «Руководства по измерению» появится исправление.

Вопросы, касающиеся применения метода COSMIC

COSMIC КПИ приносит свои сожаления, по поводу отсутствия возможности давать консультации по применению или использованию метода. Существуют коммерческие организации, которые их дают, проводят тренинги или предоставляют специальные средства для поддержки метода COSMIC. Для дополнительных деталей обратитесь к сайту: www.cosmicon.com.