

# РАЗРАБОТКА МЕТОДОЛОГИИ ИНТЕРФЕЙСНОГО ПЛАНИРОВАНИЯ ДЛЯ ПРОЕКТИРОВАНИЯ ПРОГРАММНОЙ СИСТЕМЫ В УСЛОВИЯХ НЕОПРЕДЕЛЕННОСТИ

Розанцева В.В., Разумовский А.И.

Институт проблем управления им. В.А. Трапезникова РАН, Москва, Россия

valeria.rozantseva@yandex.ru, razumowsky@yandex.ru

*Аннотация. Доклад рассматривает проблемы разработки программных систем (ПС) в условиях технической, технологической, когнитивной и ментальной неопределенности: разрывы между этапами жизненного цикла (ЖЦ) ПС, несогласованность команд и сложности проектирования взаимодействий. Цель — разработка методологических принципов интерфейсного планирования (ИП) для координации процессов ЖЦ ПС с функционалом системы. Методология ИП объединяет концептуальный дизайн, человеческие ресурсы, функциональные решения и элементы пользовательского интерфейса. Результат — программное решение, позволяющее последовательно наращивать функциональность при этом сохраняя адаптивность ПС к изменениям.*

*Ключевые слова: программная система, неопределенность, несогласованность коммуникации, интерфейсное планирование, жизненный цикл ПС.*

## Введение

Несмотря на прогресс исследований разработки технологических программных систем (ПС), например, в сфере цифровых двойников [1], специалисты регулярно сталкиваются с трудностями на этапе внедрения проектов, что свидетельствует о необходимости совершенствования технологий и рабочих методик. Ключевой проблемой остается расхождение между прогнозируемыми и фактическими результатами ИТ-проектов [2], вызванное разрывом между этапами жизненного цикла (ЖЦ) программной системы (ПС): анализом требований, проектированием, реализацией, внедрением и эксплуатацией. Устранение этих диссонансов позволит обеспечить стабильную работу систем, повысить точность реализации спецификаций и расширить возможности для последующих модификаций. Такой подход минимизирует риски возникновения ошибок, сократит время их исправления и повысит общую эффективность разработки.

Помимо несоответствий в результатах, значимым вызовом остается отсутствие слаженности между участниками процессов [3]. Дефицит коммуникации приводит к накоплению нерешенных вопросов в ходе работы. Для формирования продуктивной команды критически важны такие параметры, как численность группы, профессиональный состав, уровень компетенций участников и степень их синергетического взаимодействия.

Успех любого программного обеспечения во многом определяется качеством пользовательского интерфейса. Несмотря на достаточную изученность вопросов проектирования интерфейсов [4], их практическая реализация должна отвечать ряду критериев. К ним относятся: стилевая адаптивность, возможность инкрементального развития, масштабируемость, высокая отзывчивость, ресурсная автономия, кроссплатформенность и поддержка мультимедийных форматов. Среди ключевых принципов проектирования выделяют гармоничное расположение элементов управления, единообразие стиля для всех модулей системы, обеспечение интерактивной обратной связи и минимизацию текстовых блоков. Учет когнитивных особенностей целевой аудитории при создании интерфейсов способствует повышению эффективности взаимодействия с приложением [5], что подчеркивает важность интеграции технических и психологических аспектов в разработке. По аналогии с пользовательским интерфейсом, более значимую роль в обеспечении взаимодействия контекстов процесса разработки ПС мог бы сыграть интерфейс участников разработки, названный нами интерфейсным планированием.

Целью исследования является поиск методологических принципов интерфейсного планирования разработки ПС в условиях технической, технологической, когнитивной и ментальной неопределенности. Совокупность таких принципов должна быть направлена на координацию и синхронизацию этапов разработки, а также интеграцию пользовательских решений с системным функционалом. Основные задачи включают оптимизацию координации участников проекта, систематизацию сбора требований и структурирование функционала системы с выделением модулей и определением их иерархии. Ожидаемым результатом станет каркас интерфейсных решений, программно выполняющий роль инструмента управления и выступающий централизованным узлом

взаимодействия между всеми участниками разработки ПС: исполнителями, пользователями, функциональными задачами и ресурсами.

Методология интерфейсного планирования ИП позволит трансформировать линейное восприятие ЖЦ ПС [6] в интегрированную модель целостного временного пространства. Данный подход способен увеличить охват взаимодействия между элементами системного дизайна, исполнителями технических решений и интерфейсными компонентами, закладывая основу для их взаимной корректировки и дополнения. Исследование формирует фундамент для непрерывной разработки сложного распределенного программного обеспечения через синхронизацию действий разработчиков, оптимизацию планирования функционала, а также итеративного и вневременного учета пользовательских потребностей в эксплуатационном режиме. Реализация предложенной методологии позволит повысить качество продуктов, сократить количество недочетов и обеспечить если не гарантированную, то целеустремленную адаптацию практики к изменяющимся требованиям.

## 1. Метод

Эффективная разработка современного программного обеспечения требует перехода от строго последовательного жизненного цикла к модели, позволяющей процессам каждого этапа ЖЦ ПС действовать совместно, скоординировано. Это необходимо для осуществления немедленной эксплуатации отдельных модулей ПС по мере их готовности, что ускоряет получение полезности и повышает надежность внедрения. Ключевым аспектом становится возможность вневременного планирования и реализации взаимодействия между всеми участниками процесса: разработчиками, пользователями целевой ПС, функциональными элементами самой ПС и разнообразными ресурсами.

Традиционно первым шагом по обыкновению является четкое определение участников и их ролей. Разработчики отвечают за создание, тестирование и интеграцию функциональных элементов, обеспечивая их качество и возможность независимого развертывания. Пользователи, включая бизнес-пользователей и конечных потребителей, формулируют требования, предоставляют обратную связь на работающие части системы и начинают эксплуатировать готовые модули в реальных или приближенных к реальным условиях. К ресурсам можно отнести технические средства (серверы, базы данных, сети, библиотеки, тестовые среды), информационные активы (документация, требования, модели данных), а также прочие человеческие ресурсы (профильные специалисты и команды).

Однако, в действительности четкое распределение ролей и статусов участников оказывается сдерживающим развитие ПС фактором, поскольку устанавливает барьеры коммуникации. Нашей же задачей видится снятие таковых барьеров для установления нужной коммуникации в соответствующее время. Поэтому мы предполагаем изначально равный статус всех участников.

Также в традиционных подходах к созданию ПС предполагается глубокая декомпозиция системы на независимые модули. В условиях довлеющей многосторонней неопределенности такой путь ведет к отягощению проекта перекрестными внутренними связями, дублированием и размыванием контроля разработки с сомнительным результатом в конце.

В рамках начального этапа методологии разработки ПС мы создадим условия (или – среду), обеспечивающие возможность исправления ошибок. В сравнении с традиционными подходами упор делается не на предотвращение ошибки, а на вневременную возможность ее коррекции. Это требует введения специального термина «сцепление», как фундаментального методического понятия, определяющего системные отношения между всеми участниками ЖЦ ПС. Сцеплением в методологии интерфейсного планирования устанавливается отношение между любыми двумя участниками процесса разработки ПС (рис. 1). Здесь этому способствует равный статус каждого из участников процесса разработки, что дает возможность осуществлять связь вне рамок соответствия между участниками (1):

$$\forall x \forall y (EqualStatus(x, y) \rightarrow (CanCommunication(x, y))) \quad (1)$$

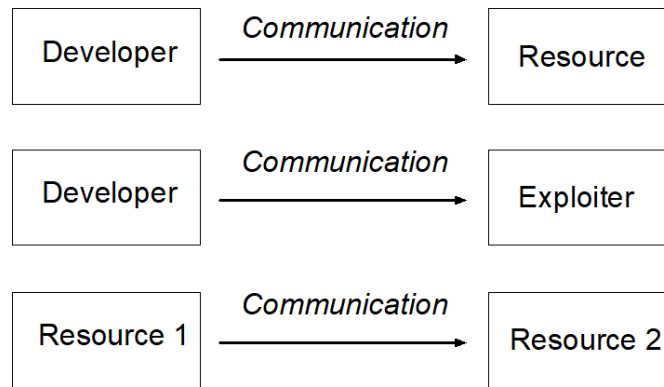


Рис. 1. Установление произвольного сцепления между участниками равного статуса

Выражение (1) утверждает: для любых двух участников процесса разработки, если их статус равен, то они могут свободно общаться друг с другом, выходя за рамки формальных ограничений. Очень важен здесь выход за рамки формальных отношений, что позволит действовать менее зажато, менее скованно. Одновременно, кроме этого, можно избежать предварительной установки правил коммуникаций, которые всегда осложняют взаимодействия.

Семантика термина «сцепление» содержит автономность каждой новой связи по отношению к уже имеющимся в системе. Тем самым достигается их независимость, и не ведет к возникновению помех в развитии ПС. Однако предполагая ситуацию разрыва сцепления, при необходимости, например, замены функционального решения на иное или перенаправления потоков данных, следует включить в метод возможность разрыва существующего сцепления без потерь качества структуры ПС. Для достижения указанной цели мы предлагаем усилить термин «сцепление» определением «косвенное». Тогда для разрыва станет достаточно удалить лишь один из двух составляющих косвенное сцепление отрезков (рис. 2).

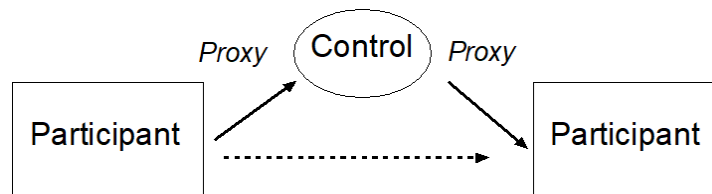


Рис. 2. Косвенное сцепление между участниками разработки ПС

## 2. Результаты и Обсуждение

Согласно выражению (1), как было отмечено выше, во взаимодействии важен выход за формальные отношения, что обеспечивается отсутствием исходного статуса участника при установлении нового сцепления. Статус проявляется при завершении планирования интерфейса, когда утверждается, например, что определенный разработчик связан с поставленной ему конкретной задачей или разработчик передает на эксплуатацию свой результат конкретному лицу. Равный изначально статус или роль предотвращает создание «якоря» привязки к определенным характеристикам участника (человек, тестировщик, библиотека, документ) и освобождает при таком планировании от когнитивного обременения желаемого плана.

Дополнительное преимущество дает косвенная связность, когда сцепление состоит из двух сегментов, и один из них позволительно разорвать без разрыва второго. Затем осуществить обновление сцепления, соединяя оставшийся неизменным сегмент с добавлением нового. Узел косвенного сцепления, прокси – это контроллер обмена данными между участниками. Реализация косвенного сцепления может быть осуществлена на основе асинхронного интерфейса.

При разрыве одного из составляющих косвенное (прокси) сцепление отрезков образуется возможность установления новой связи между иными участниками. Кроме того, прокси-сцепление позволит избегать жестких окончательных решений, формируя адаптер ситуативного реагирования и вмешательства. Таких планов у каждого из участников может быть более одного, однако при косвенном сцеплении реализация множественных планов легко осуществима.

Формально назначение и использование косвенного сцепления (РС) может быть выражено посредством сочетания двух отрезков  $O1$  и  $O2$  через общую точку-контроллер  $P_i$ , (2):

$$PC = \{O1(P_x, P_i), O2(P_i, P_y)\} \quad (2)$$

Разрыв сцепления формализуется выражением:

$$Break(PC) = Remove(O1) \vee Remove(O2) \quad (3)$$

Вставка новой связи (NC) может быть выражена:

$$Break(O1) \vee Break(O2) \rightarrow Enable(NC) \quad (4)$$

Преимущества косвенной связи можно выразить следующим образом:

$$PC \rightarrow \neg RigidDecisions \wedge AdaptivePotential \quad (5)$$

Полнота модели сформирована при помощи аддитивных операций вставки элемента-сцепления (4), его разрыва (3), а также необходимости обеспечения смягчающего разрыва сцепления, усложняя его через косвенность (5).

В предложенной методологии есть один видимый изъян. Он заключается в необходимости запрета косвенного сцепления между людьми – разработчиками и пользователями. Если допустить существование косвенной связи между ними, тогда это будет означать разрыв ответственности, и он может существенно осложнить разработку ПС. Поэтому в конечном счете в методе присутствуют два вида сцепления – прямое и косвенное. Каждое из них применяется непосредственно при планировании интерфейса, когда определенным становятся вид сцепления и статус его участников.

### 3. Выводы

В представленном докладе предложена методология интерфейсного планирования разработки ПС в условиях разноплановой (технической, когнитивной, ментальной) неопределенности. Чтобы иметь возможность планировать и осуществлять произвольное взаимодействие (сцепление) вне зависимости от временной точки на пути ЖЦ ПС, между участниками такого взаимодействия не должно быть исходного различия в статусе. Кроме этого, необходимо разделить виды сцепления на прямое (непосредственное) – между людьми, – и косвенное – между людьми и объектами и средствами разработки.

В качестве наглядной иллюстрации методологии ИП хорошо подойдет взятое из живой природы устройство пчелиных сот (рис. 3).

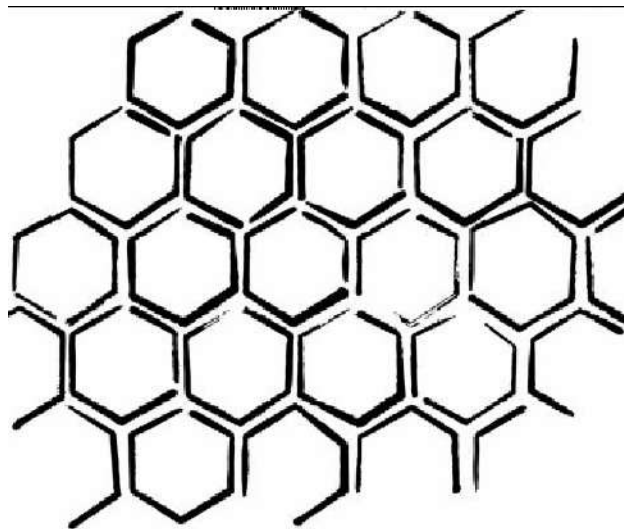


Рис. 3. ИП, как аналог пчелиной сотовой архитектуры

Подобная архитектура позволяет обеспечить следующие принципы разработки ПС:

- образование сцепления, состоящего из двух и более элементов планирования;
- минимизация ограничений между произвольными участниками разработки ПС посредством их исходного равного статуса;
- препятствование осуществлению новых планов и целей за счет косвенного сцепления.

Исходным образцом для создания представленной методологии ИП послужила работа, связанная с созданием программно-учебного космического тренажера, где особенно эффективную роль сыграла косвенная асинхронная взаимосвязь между функциональными элементами [7].

## Литература

1. *Tao F. et al.* Digital twin in industry: State-of-the-art // IEEE Transactions on industrial informatics. – 2019. – Vol. 15, № 4. – P. 2405–2415.
2. *Martinez I., Viles E., Olaizola I.G.* A survey study of success factors in data science projects // 2021 IEEE International Conference on Big Data. – 2022. – P. 6852.
3. *Berntzen M. et al.* A taxonomy of inter-team coordination mechanisms in large-scale agile // IEEE Transactions on Software Engineering. – 2022. – Vol. 49, № 2. – P. 699–718.
4. *Moreno L. et al.* Designing user interfaces for content simplification aimed at people with cognitive impairments // Universal access in the information society. – 2024. – Vol. 23, № 1. – P. 99–117.
5. *Habib T. A. et al.* User Interface Design for the Orca Music Player Mobile Application // Tsabit Journal of Computer Science. – 2024. – Vol. 1, № 1. – P. 18–26.
6. *Pargaonkar S.* A comprehensive research analysis of software development life cycle (SDLC) agile & waterfall model advantages, disadvantages, and application suitability in software quality engineering // International Journal of Scientific and Research Publications (IJSRP). – 2023. – Vol. 13, № 08. – P. 345–358.
7. *Толок А.В., Разумовский А.И., Бронников С.В., Павлов Д.В., Кузин С.А., Локтев М.А., Плаксин А.М., Ромакин В.А.* Разработка трехмерных моделей для учебно-тренировочных средств космического комплекса // Космическая техника и технологии. – 2019. – № 3 (26). – С. 69–76.