**1. Определение предметной области**

Предметную область можно определить как сферу человеческой деятельности, выделенную и описанную согласно установленным критериям. В описываемое понятие должны входить сведения об ее элементах, явлениях, отношениях и процессах, отражающих различные аспекты этой деятельности. В описании *предметной области* должны присутствовать характеристики возможных воздействий окружающей среды на элементы и явления *предметной области*, а также обратные воздействия этих элементов и явлений на среду. Работа по изучению и анализу *предметной области*: проектировании интеллектуальных систем оказывает решающее влияние на эффективность ее работы.

Специфика *предметной области* может оказывать существенное влияние на характер функционирования проектируемой интеллектуальной системы, выбор метода представления знаний, способов рассуждения о знаниях, и т. д.

Предметную область можно определить как *объект* или производственную систему со всем комплексом понятий и знаний о ее функционировании. При исследовании проблемной области необходимы знания о задачах, решаемых в производственной системе, и стоящих перед ней целях. Определяются также возможные стратегии управления и эвристические знания, используемые в процессе эксплуатации производственной системы.

**2. Анализ предметной области (анализ осуществимости, бизнес - моделирование)**

Одна из первых задач, с решением которых сталкивается разработчик программной системы - это изучение, осмысление и *анализ предметной области*. Дело в том, что предметная область сильно влияет на все аспекты проекта: требования к системе, взаимодействие с пользователем, модель хранения данных, реализацию и т.д.

*Анализ предметной области*, позволяет выделить ее сущности, определить первоначальные требования к функциональности и определить границы проекта. Модель *предметной области* должна быть документирована, храниться и поддерживаться в актуальном состоянии до этапа реализации. Для документирования могут быть использованы различные средства.

Для управления обсуждением области действия проекта можно использовать методику "будет - не будет". В простейшем случае - это *список* с двумя столбцами, в одном из которых записывается, что проект будет делать, а во втором - что не входит в проект. Такой *список*, формируется заинтересованными лицами при рассмотрении каждой *бизнес-цели проекта*, используя любую технику, например метод "мозгового штурма" (см. тему "Выявление требований"). Полученные характеристики позволяют четко определить границы проекта и довольно просто преобразуются в предположения, которые фиксируются в документе.

Функциональная *область действия* определяет услуги, предоставляемые системой, и вначале до конца неизвестны. В определении услуг системы может помочь *список* "Действующее лицо/Цель", в котором перечислены все цели пользователя, поддерживаемые системой. При его разработке в первую графу вписываются имена основных действующих лиц, т.е. тех, кто имеет цели, во вторую графу - цель каждого действующего лица, а в третью - приоритет или предположение о том, в какую версию войдет эта услуга. Формы списков приведены на рисунке.

Для определения основных функций продукта можно использовать, например, краткое описание варианта использования. Описание каждой функции можно представить также в виде списка, состоящего из трех *граф*: действующее лицо, цель и краткое описание варианта использования.

*Анализ предметной области* является основой для *анализа осуществимости* проекта и определения образа (концепции) продукта и границ проекта.

**Анализ осуществимости**

Разработка новых программных систем должна начинаться с *анализа осуществимости*. На основании анализа предметной области, общего описания системы и ее назначения необходимо принять решение о продолжении или завершении проекта. Для этого необходимо ответить на следующие вопросы.

1. Отвечает ли система бизнес-целям организации-заказчика и организации-разработчика?
2. Можно ли реализовать систему, используя известные технологии и не выходя за пределы заданной стоимости и заданного времени?
3. Можно ли объединить систему с другими уже эксплуатируемыми системами?

Для ответа на первый (и главный) вопрос нужно опросить заинтересованных лиц, например, менеджеров подразделений, в которых будет использоваться система, для выяснения того,

* что произойдет с организацией, если система не будет введена в эксплуатацию;
* как система будет способствовать целям бизнеса;
* какие текущие проблемы поможет решить система и т.д.

После получения и обработки информации готовится отчет, в котором должны быть даны рекомендации относительно продолжения разработки системы.

**Постановку бизнес-задачи** надо обсуждать с Заказчиком, или будущим Владельцем системы.

Вопросы, которые ему стоит задать, это:

1. Почему вообще пошла речь о создании системы?
2. В чём Вы видите её назначение?
3. Какие бизнес-возможности она должна реализовать?
4. Какие проблемы должна решить?

В качестве "Стандарта" на вопросы такого рода смотрите *шаблон документа* "*Stakeholder* Request", например, из *RUP*. Бизнес-требования может выразить Заказчик или Эксперты *предметной области*. Они обычно фиксируются в виде списка из 10-30 ключевых свойств продукта - в качестве шаблона см. Vision из *RUP*.

**Бизнес-моделирование** надо проводить на основе информации от, а лучше совместно с экспертами *предметной области*. Вопросы по сути сводятся к "Что, почему, когда, как и кем происходит в *предметной области* и как оно взаимосвязано?":

1. Каковы основные понятия предметной области, их определения и взаимосвязи? Результат можно оформить в виде глоссария и/или концептуально-семантической модели предметной области.
2. На основании каких правил - международных, федеральных, муниципальных, районных и т.д. законов, указов, стандартов, спецификаций, регламентов и т.д. - происходит то, что происходит в предметной области? Результат оформляете в виде структурированного списка или прикрепляете к элементам концептуальной модели.
3. Что реально (какие процессы, события, факты) происходит и в какой последовательности, взаимосвязи? Результат оформляете в виде сценариев описания бизнес-процессов (что достаточно универсально) или диаграмм *SADT* (*IDEF0*, IDEF3, *DFD*) / *ARIS* (eEPC и т.д.) / UML (Business Use-case Diagram (BUC) + *Activity Diagram* + *Sequence Diagram*). Это один из сложнейших этапов.
4. Какими свойствами обладает каждое из выделенных понятий - структурными и поведенческими? Результат описывается в виде таблиц с атрибутами Концептуальных сущностей или Детальной концептуальной моделью - ER - *IDEF1X* / UML *Class Diagram* (*BOM*).

Существует российский стандарт *функционального моделирования* P 50.1.028-2001, созданный на основе *IDEF0*.

**Определение требований** - частично Бизнес-требования и Требования, проистекающие из *предметной области* вы уже определили выше, теперь осталось исследовать *Пользовательские требования* и *Системные требования* и ограничения к отдельным аспектам качества системы. *Пользовательские требования*, как можно понять, нужно выявлять из общения с потенциальными пользователями системы. Вопросы:

1. На какую систему будет похожа создаваемая?
2. С какими системами и как давно вы работаете?
3. Какое у вас образование?
4. Каковы ваши ожидания от системы - что и как она должна делать, какие задачи помогать решать, как должна выглядеть?
5. Какие шаги необходимо предпринять для решения каждой задачи?
6. В каком случае вы будете считать, что система "Хороша"?

Результаты анкетирования/интервьюирования обычно представляют в виде пользовательских историй (*User* *Story*, Agile) или Пользовательских сценариев (Use-case), также возможно их диаграммное *представление* средствами диаграмм потока *работ*(IDEF3), *ARIS*, Activity/State *UML Diagram*. Подробнее про работу с Пользователями могут рассказать специалисты по Проектированию взаимодействия, интерфейсам и эргономике.

*Системные требования* нужно выяснять у IT-специалистов Заказчика, если таковые имеются, из специфики контекста использования системы, опыта построения аналогичных систем (у IT-Экспертов-Архитекторов) и Специалистов по отдельным аспектам системы, значимым для данного проекта (Юристы, Эргономисты, и т.д.) и Заказчика:

1. Будет ли система единичной или тиражируемой?
2. В каких странах она будет работать?
3. Насколько важна информация, хранящаяся, обрабатываемая и передающаяся системой?
4. Каков возможный ущерб от потери той или иной информации?
5. Сколько пользователей будет работать с системой сегодня, завтра, через год?

Переработанный результат оформляется в виде Системных требований (*Software* Requirement *Specification*, стандарт IEEE-STD-830-1998, или ТЗ ГОСТ 34-602-89 или неформально в виде Supplementary Specificaion из *RUP*).

Приложения для настольных компьютеров подобны широкоугольным объективам в том смысле, что в типичных случаях они отображают значительный объем информации, который позволяют предоставлять пользователю экраны большого размера. В отличие от этого мобильные приложения напоминают увеличительное стекло или объектив с переменным фокусным расстоянием. Они предоставляют пользователю возможность быстро просматривать необходимые подробные данные, быстро переходить к ограниченным наборам данных и получать к ним *доступ*, а также принимать решения в реальном масштабе времени. Как правило, мобильные приложения пре доставляют более специализированный набор сценариев по сравнению с приложениями, ориентированными на настольные компьютеры. Очень важно точно определиться с тем, на каких сценариях должно специализироваться ваше *приложение*.

Прежде чем приступать к реальной разработке приложения, определите *подмножество* функциональных средств, к которым *пользователь* сможет получать быстрый *доступ* в манере, свойственной мобильным устройствам. В случае создания нового приложения, аналога которого для настольных компьютеров не существует, выпишите ключевые сценарии, которые пользователи смогут выполнять с помощью вашего приложения, а также порядок действий пользователя, обеспечивающий использование этих сценариев на мобильном устройстве. Во многих случаях вам будет легче придать этим сценариям реальные очертания, если вы подготовите соответствующие рисунки или создадите прототипы. Если подразумевается определенная *группа* конечных пользователей, пообщайтесь с ними и предоставьте им возможность поработать некоторое время с экспериментальными версиями своих приложений, чтобы они могли дать о них свои отзывы.

*Оптимальный подбор предоставляемых средств определяет все остальное*

Если вы правильно выделите ключевые сценарии и возможности вашего приложения, то это окажет определяющее влияние на всю оставшуюся часть процесса разработки. Наличие явно сформулированного описания того, как *конечные* пользователи будут использовать ваше *приложение*, и детальное понимание их потребностей окажут вам неоценимую помощь при настройке производительности приложения, а также проектировании пользовательского интерфейса, коммуникационной системы и модели памяти.

Если вы не определите важнейшие с вашей точки зрения сценарии и возможности, то в результате вы получите бессистемную смесь средств, объединенных в одно *приложение*. Отсутствие явного списка основных функций приложения или разделения функций на группы в соответствии с их приоритетами приведет к тому, что пользовательский *интерфейс* не будет оптимизирован для эффективного решения ключевых задач. Например, если ожидается, что *пользователь* в основном будет заинтересован во вводе данных, то вы должны оптимизировать пользовательский *интерфейс* таким образом, чтобы обеспечить как можно более точное и надежное выполнение операций ввода. И наоборот, если ввод данных используется лишь изредка, то вариант пользовательского интерфейса ввода, оптимизированного не самым идеальным образом, может оказаться вполне допустимым, что позволит перебросить ресурсы проектирования и разработки на другие направления. Лишь только если группой разработчиков будут идентифицированы, перечислены и согласованы наиболее важные сценарии, эксплуатационные характеристики приложения могут быть настроены для их выполнения должным образом, а *конечные*пользователи не будут лишены важных для них средств из-за недосмотра.

Чтобы процесс разработки мог быть успешно завершен, составьте *список* ключевых требовании, которым должно удовлетворять *приложение*, и возможноестей, которые оно должно обеспечивать, и пусть этот *список* будет первым разделом вашего основного документа проекта.

**3. Формирование и документирование требований к проекту**

Для всех проектов разработки программного обеспечения, кроме самых тривиальных, важно иметь единственный "руководящий документ", в котором определяются:

1. требования проекта и целевое назначение завершенного продукта;
2. философия проекта;
3. архитектура приложения;
4. текущее состояние работ в данном направлении;
5. план, в соответствии с которым продукт будет переводиться из его текущего состояния в состояние успешного завершения.

Для крупных проектов могут предусматриваться дополнительные документы, но всегда должен существовать один основной документ самого верхнего уровня, в котором определяются основные *цели проекта*, его нынешнее состояние и план *работ*. Этот документ должен быть реально действующим документом, который, по мнению всех участников группы, правильно определяет направления работы.

Единственный руководящий документ, который управляет всем процессом разработки, должен включать несколько разделов:

* **Цели проекта**. В этом разделе основного документа должно быть ясно указано, для чего предназначается завершенный продукт, и какая философия лежит в основе проекта.
* **Состояние проекта**. Этот раздел основного документа содержит точное описание текущего состояния проекта. Он является частью календарного плана проекта и позволяет судить о степени приближения работ над проектом к завершению. Здесь должны быть определены контрольные точки проекта, на достижение которых должны концентрироваться усилия, а также указано, какие элементы должны быть доработаны для завершения текущего контрольной точки процесса разработки. Здесь же должен предоставляться список наиболее значимых факторов риска или факторов, которые необходимо исследовать для разрешения соответствующих проблем.
* **Архитектура приложения и соответствующие диаграммы состояний**. Эта информация отражает технические аспекты плана. Для мобильных приложений этот раздел должен содержать соответствующие диаграммы состояний, описывающие дискретные состояния, в которых может находиться приложение, и связь этих состояний с объемами памяти и ресурсов, которые будут в ней храниться. (Далее об этом будет говориться более подробно.) Такой раздел фактически играет роль соглашения между всеми участниками группы разработчиков, в котором они обязуются придерживаться в своих реализациях установленных в нем требований. Если вы выступаете в роли единственного разработчика, то этот документ даст вам возможность оставаться честным перед самим собой; у каждого, кому довелось хотя бы однажды самостоятельно разрабатывать крупный проект, наверняка иногда возникало желание срезать тот или иной угол, чтобы добиться работоспособности средства, пусть даже это и будет в ущерб разумным принципам проектирования. Срезоть углы гораздо сложнее, если перед вашими глазами находится соглашение, в котором указано, что вы должны в явной виде формулировать все свои предложения по ускорению работы над проектом. Этот раздел не должен быть чрезмерно длинным или сложным, ибо в противном случае выполнять его требования будет трудно, и им будут просто пренебрегать. В нем должно быть сформулировано, что необходимо сделать для того, чтобы проект удерживался в организационном русле, и, что еще важнее, в нем должны оперативно учитываться любые согласованные изменения проекта.
* **План разработки с указанием отдельных контрольных точек**. Еще более важную, чем календарный график, роль играет взвешенный план, устанавливающий дис кретный набор контрольных точек проекта, которые должны проходиться по мере приближения проекта к завершению. По самому своему определению контрольные точки позволяют оценивать прогресс, достигаемый на пути к определенной конечной цели. Каждая контрольная точка представляет собой точку покоя, в которой вы можете остановиться, чтобы оценить состояние работ, подчистить код, который не был своевременно приведен в порядок, и при необходимости скорректировать проект. По мере выполнения проектных работ цели могут меняться, что влечет за собой необходимость корректировки контрольных точек; это допускается. В соответствии с эволюцией проекта может меняться его архитектура, что, в свою очередь, может потребовать внесения изменений в модель состояний; и это нормально. Можно почти не сомневаться, что пользовательский интерфейс также будет несколько раз переделываться. Отслеживание выполнения (или констатация невозможности выполнения) предварительно определенных контрольных точек является самым надежным мерилом прогресса, достигнутого в работе над проектом, и подсказывает, какие коррективы должны быть внесены по мере приближения к финишной линии.