1. *Понятие живучести сети*

* Способность системы функционировать в условиях агрессивного воздействия внешней среды
* На устройство воздействуют, чтобы вывести из строя, а я всё равно могу им пользоваться

Живучесть – наличие альтернативного канала передачи данных в случае возникновения внутренних/внешних проблем.

Живучесть – способность системы своевременно реализовывать своё назначение в условиях атак, отказов и аварий.

Основной построения метрических характеристик живучести являются неделимый комплекс понятий: миссия, выполнение миссии, атака, отказ, инцидент/катастрофа/несчастный случай. Понятие «миссия» характеризирует смысл существования системы. Под «атакой» понимается специально организованное воздействие, направленное на разрушение системы, либо приведение её в состояние отказа в обслуживании. Под «отказом» понимается событие, связанное с повреждением системы, и обусловленное внутренними дефектами в компонентах внешней среды, влияющих на систему, такими как ошибки, допущенные при проектировании программных систем, либо разрушение данных. Понятия инцидент/катастрофа/несчастный случай соотносятся с внешними событиями и физическими явлениями.

Для обеспечения живучести системы необходимо наличие четырёх компонентов: устойчивость к атакам, способность комплексного оценивания ущерба от атаки, восстанавливаемость основных сервисов после атак, адаптация и эволюция систем с целью снижения результативности. Первые три компонента ассоциируются с реактивной стратегией управления, последний – с проактивной.

Метрической характеристикой живучести является время разрушения системы.

1. *Базовые вопросы предпроектной стадии разработки ПО*

* Зачем это нужно?
* Что мне это даст?

Чтобы потенциальный заказчик стал работать со мной и профинансировал, нужно объяснить ему, зачем это нужно. Узнав, какие у заказчика проблемы, нужно убедить его, что ты сможешь их решить. Пример: автоматизация отчётов с помощью компьютеров.

* Что случилось?
* Чьи интересы затрагивает? В чём заключаются ценности акторов?
* Почему случилось?
* В чём коренная причина?
* Почему должны действовать совместно?
* Как действовать в условиях различия в ценностях?
* Как преобразовать ценности акторов в систему IT-сервисов?
* Поиск вариантов приемлемых затрат на получение информационных услуг.
* Какими потребительскими свойствами должны обладать отдельные IT-сервисы?
* Как реализовать программные системы, обеспечивающие необходимые IT-сервисы?
* Как правильно реализовать программную систему, реализующую IT-сервисы?

1. *Место информационной системы в модели управления урегулированием проблемной ситуацией*

ИС много что дают, но вызывают новые проблемы.

Информационная система играет ключевую роль в модели управления урегулированием проблемной ситуации. Она помогает:

* Идентифицировать проблемную ситуацию по набору показателей, значения которых имеют отклонения от нормы в течение заданного лага времени.
* Автоматически искать проблемную ситуацию и выявлять возможные альтернативы её разрешения.
* Проектировать новую схему бизнес-процесса и размещать её в репозиторий метамоделей. При возникновении аналогичной проблемной ситуации её разрешение будет проходить автоматически, путём извлечения из репозитория метамодели проблемной ситуации и метамодели бизнес-процесса её разрешения.
* Выдавать команды в соответствующие центры ответственности, которые должны разрешить проблемную ситуацию.

**

1. *Концептуальная основа модели Swiss Cheese Model*

Фокусом концепции является положение о том, что никакой инцидент не может быть обусловлен единственной причиной. Инцидент является результатом непредсказуемого сочетания не скольких факторов, истоки которых разнесены в пространстве и времени. Выделяется системная составляющая инцидента, обусловленная нерациональными организационными, проектными и технологическими решениями, и являющаяся причиной возникновения латентных дефектов. Случайные, непредсказуемые внешние воздействия приводят к активизации латентных дефектов («дыр» в слоях системы).

* *В любой системе защиты имеется множество «дыр», меняющих своё положение в пространстве и времени*
* *Наличие «дыр» само по себе не обязательно приводит к негативным событиям*
* *Негативные события возникают тогда, когда источники опасности и «дыры» оказываются на одной линии*

Методические ограничения SCM:

1. Предполагается линейная схема преобразования источника опасности в инцидент. Не учитывается то обстоятельство, что латентные дефекты в вышестоящих слоях иерархической системы могут быть обусловлены ошибочным реагированием на отказы, ранее имевшие место в нижележащих слоях.

2. Предполагается, что казуальные цепочки возникают хаотично. Отсутствуют подходы к ранжированию возможности реализации инцидентов, обусловленных различными казуальными цепочками.

3. Не предусмотрена возможность одновременного возникновения инцидентов разной природы.

4. Постулируется линейная упорядоченность событий во времени. Однако из того, что событие А предшествовало событию В, не следует, что А является причиной В. В рамках SCM не представляется возможным указать события – коренные причины инцидентов, разнесенные в пространстве и во времени.

5. Не определен подход для оценивания вклада субъективной, организационной и технологической составляющих в возникновение латентных дефектов.

Ограничения на практическое использование Swiss cheese model:

1. SCM на настоящем уровне развития не позволяет предвидеть (предсказывать) инциденты.

2. Модель не проходила верификацию.

3. Отсутствует стандарт, регламентирующий ее применение.

4. Неопределенность выявления потенциальных казуальных цепочек.

5. Неопределенность содержания «дыр» (holes) в слоях сыра.

* 6. Не определены причины изменчивости местоположения и размеров «дыр» во времени.

1. *Классификация защитных барьеров*

* *Hard Defenses (Жёсткие барьеры)*
  + *Проектные решения* (если Вы хорошо владеете технологиями проектирования это лучше, чем если не владеете, шанс совершить ошибку меньше)
  + *Конструкторские решения* (если хорошо знаете язык программирования, меньше шанс совершить синтаксическую ошибку)
  + *Ограничения на ошибочные действия* (защита от дурака – пользователь всегда ошибается, ограничить взаимодействия с клавиатурой, сделать выпадающее меню)
* *Soft Defenses (Мягкие барьеры)* – изучение синтаксиса языка программирования
  + *Процедуры*
  + *Правила*
  + *Инструкции* (стандарты)
  + *Обучение*

1. *Swiss Cheese Model как концепция*

См. 35