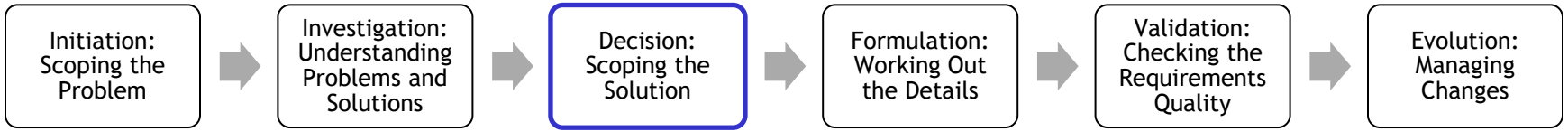
Requirements Decisions: from goals to prioritized features

Emmanuel Letier

Phase 3: Decision (aka “requirements evaluation, negotiation, prioritization”)



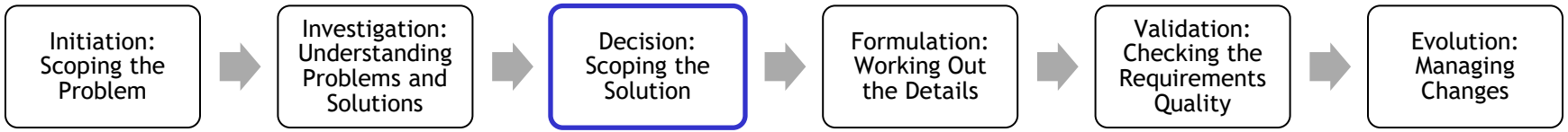
Objectives:

- Identify candidate features and qualities
- Estimate impacts of features and qualities on stakeholder goals
- Decide what features and qualities to build

Techniques:

- Impact mapping, user story mapping, goal modelling
- Strategic release planning
- Prototyping to test and compare ideas

Inputs and Outputs



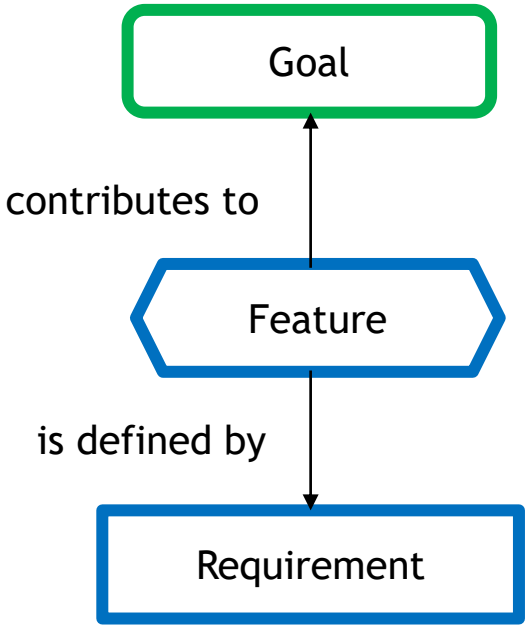
Inputs:

- Project goals, scope, stakeholders
- Issues, ideas, aspirations, opportunities

Outputs:

- Candidate features for implementation
  - prioritized by their anticipated costs and impacts on the project goals
- Note: features are not defined in detail yet.

What is a feature?



Software feature = an externally visible functionality or quality of the software

- Features contribute to stakeholders' goals
- Features are defined by a set of software requirements

Example 1: An airplane feature

Feature: Ground Braking Safety

Contributes to goals:

- Avoid [Ground Braking Deployed When Flying]
- Maintain[Ground Braking Enabled After TouchDown]

Defined by requirements:

- Ground braking shall be enabled if and only if at least one of the following two conditions are met:

  (i) the oleo struts are compressed at both main landing gears, or

  (ii) wheel speed is above 72 kts at both main landing gears.

Example 2: Ambulance Dispatching System

Feature: Identification of Duplicate Calls

Contributes to goals:

- Avoiding mobilizing more ambulances than necessary
- Faster and better handling of call backs and duplicate calls

Defined by requirements:

- *The system shall identify duplicate calls and support calls handlers in dealing with such calls.*

Detailed desired behavior for the feature to be defined in the formulate phase.

Features prioritization

Candidate features for ambulance dispatching software:

- on-screen call taking
- automated caller location detection
- identification of duplicate calls
- automatic vehicle location through GPS
- allocation of nearest available ambulance
- communication of mobilisation orders to ambulance mobile data terminals
- …

Which features

- will be delivered in the next release?
- will be delivered in later releases?
- will not be delivered?

Common Difficulties in Software Projects

- Software features not aligned with stakeholder and business goals
  - High-value features are missing or inadequate.
  - Some of the implemented features are not needed or low value.
- Scope creep
  - Continuous addition of new features that are unrelated or only vaguely related to the project main goals.
- Unstructured, hard-to-understand requirements
  - Long list of “shall statements”, large collection of user stories, etc.
  - This impedes requirements understanding and validation.

Techniques

1. Impact Mapping
2. User Story Mapping
3. Goal Modeling
4. Strategic Release Planning

This is a toolbox of some of the most common techniques.

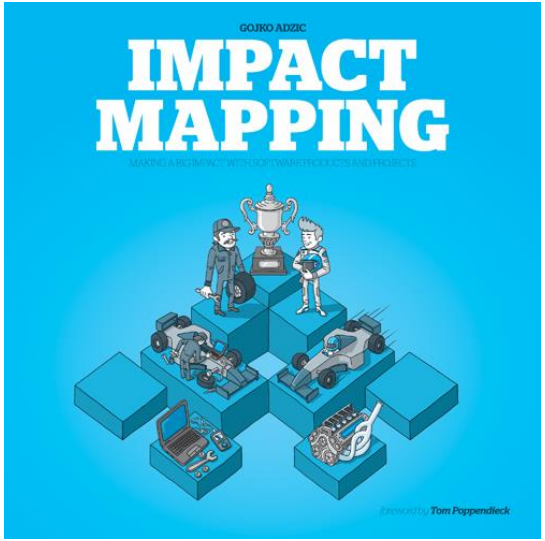
Choose the right technique or set of techniques for a particular situation.

Impact Mapping

Impact Mapping

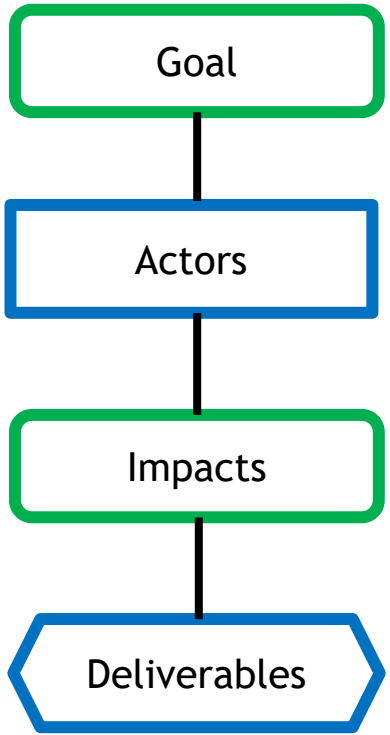
A workshop facilitation technique to help business stakeholders define their goals and ways to achieve them in the world, before discussing software features.

- It is a quick lightweight technique that combines the initiation, investigation and decision phases in a single workshop.
- It can be used either as a substitute or complement to more detailed techniques.



http://impactmapping.org

Impact Mapping: concepts



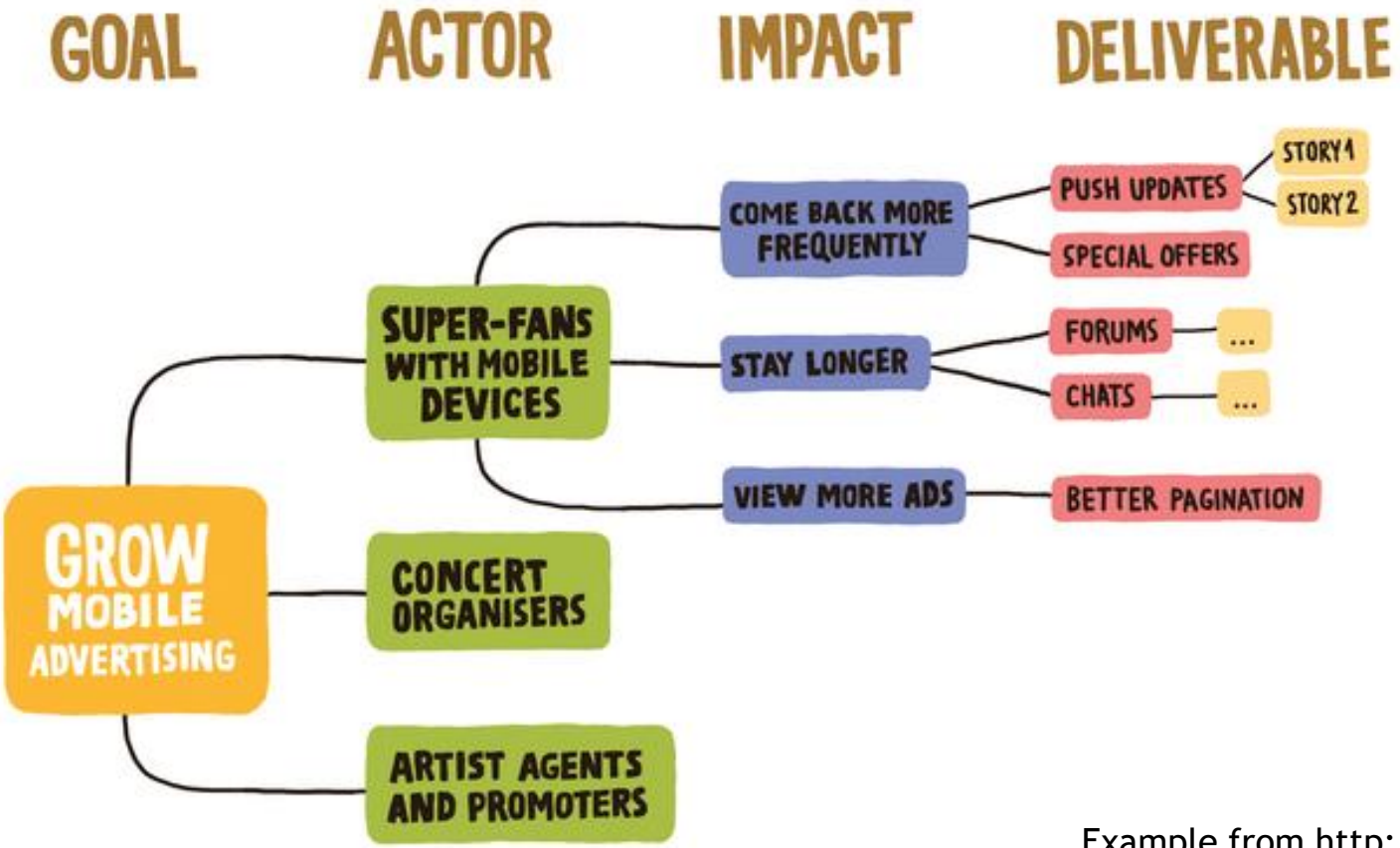
A measurable stakeholder goal

People who might contribute to the goal

Actor's behaviour change that would contribute to the goal

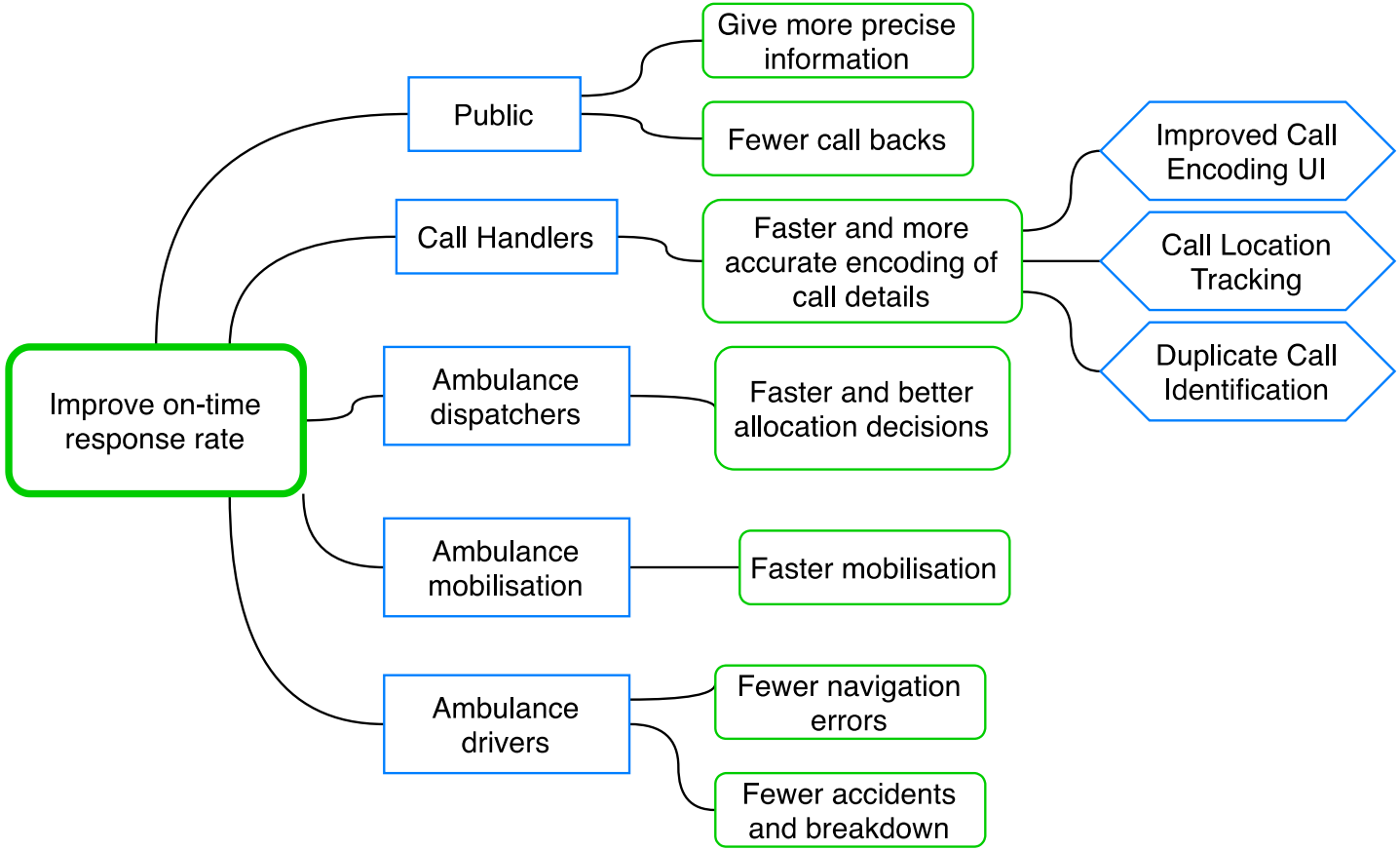
Software feature or other initiatives that would bring about the desired impacts

Example: Impact map for an app selling concert tickets



Example from http://impactmapping.org

Example: Emergency Ambulance Service



User Story Mapping

User Story

A **user story** is a short description of functionality, told from the perspective of a user, that provides some value to a user or other stakeholders

Common user story format

As a <role>

I want <action>

So that <goal>

Example

As a student

I want to see my timetable

So that I know when and where my classes are

User stories are ubiquitous in agile project management (e.g. scrum, kanban)

Agile Project Management Boards

Projects / Beyond Gravity

Board

4 days remaining | Complete sprint | ...

Epic | GROUP BY Choices

TO DO 12

- Implement feedback collector — NUC-205 — 9
- Bump version for new API for billing — NUC-206 — 3
- Add NPS feedback to wallboard — NUC-208 — 1

IN PROGRESS 4

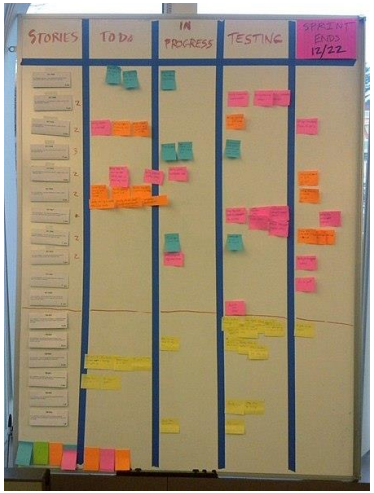
- Update T&C copy with v1.9 from the writers guild in all products that have cross country compliance — NUC-213 — 1
- Tech spike on new stripe integration with paypal — NUC-215 — 3
- Refactor stripe verification key validator to a single call to avoid timing out on slow connections — NUC-216 — 3
- Change phone number field type to 'phone' — NUC-217 — 1

IN REVIEW 4

- Multi-dest search UI web — NUC-338 — 5

DONE 4

- Quick booking for accomodations - web — NUC-336 — 4
- Adapt web app no new payments provider — NUC-346 — 3
- Fluid booking on tablets — NUC-343 — 5
- Shoping cart purchasing error - quick fix required. — NUC-354 — 1



What are user stories for?

User stories are work items for project management

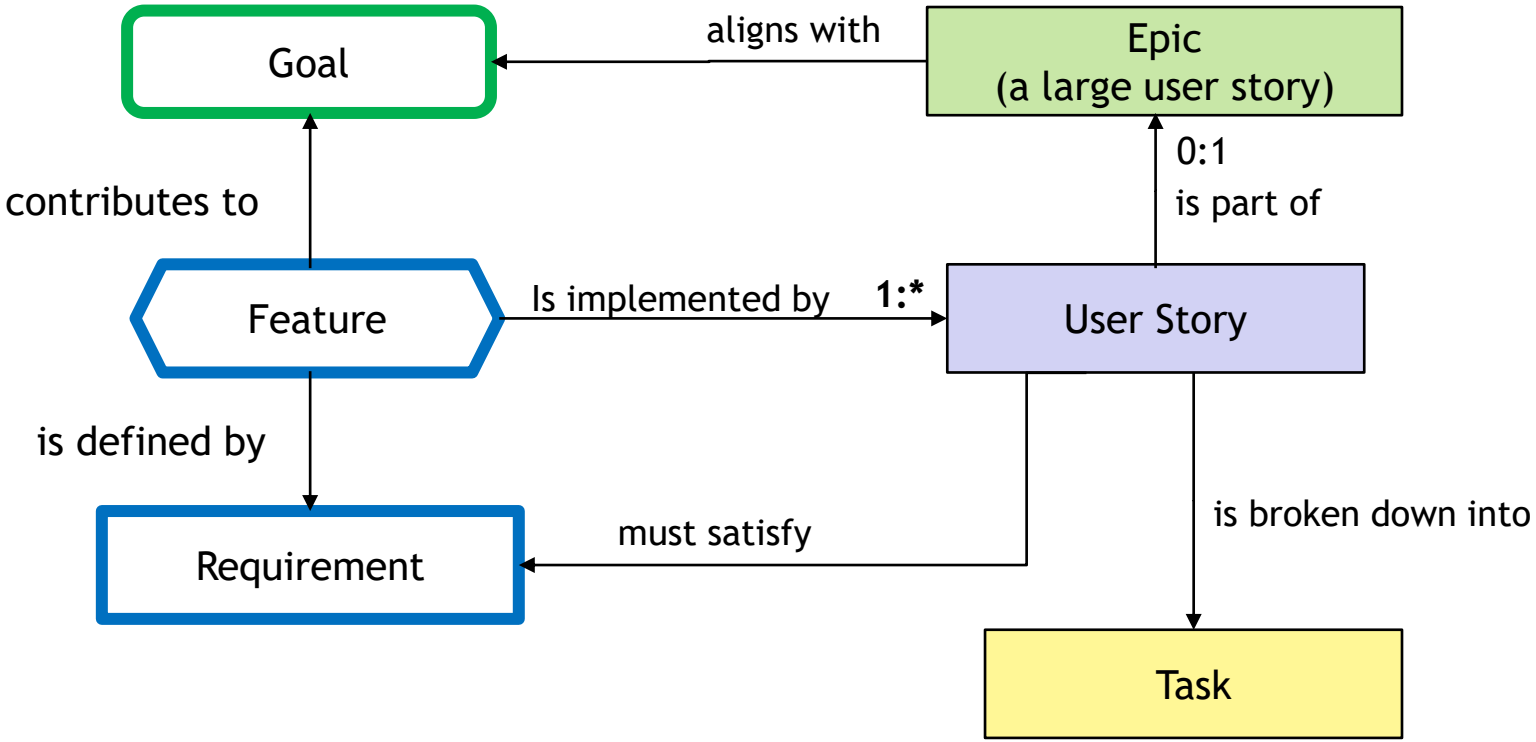
- to estimate effort (‘story points’)
- to plan work for next iteration (‘spring planning meeting’)
- to track progress (user stories status, team velocity)
- ***to trigger detailed requirements discovery and formulation***
- to assign development tasks to developers

Once done, the user story can be thrown away.

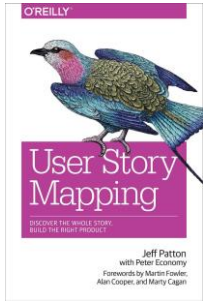
User stories are not requirements

- A feature can be the result of implementing multiple user stories over several iterations. Some stories are change requests.
- The sequence of user stories does not define the feature’s externally visible behaviors and qualities.
- Analogy: trying to understand a feature by reading commit messages

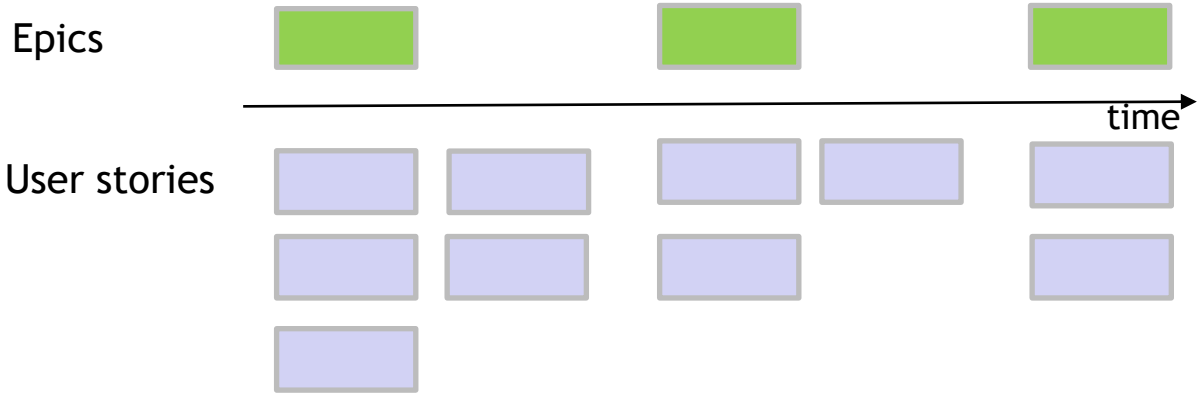
Relation between user stories and requirements



User Story Mapping



A workshop to identify, structure, and prioritize user stories



- Epics are large user stories organised temporally in order in which you would explain how users would use the system
- Epics are refined into atomic user stories shown under them
- User stories are listed by priority, most important at the top

Example: Ambulance dispatching

Receive Emergency Call	Select Ambulance	Mobilise Ambulance	Monitor Mobilisation Progress
Encode incident details	Show nearest available ambulance	Send Mob order to MDT	Detect when mob order ignored
Locate incident by caller location	Staff selects from proposed ambulances	Send Mob order to Station Printer	...
Detect duplicate calls	...	...	

User Story Mapping in Practice

Pictures from Jeff Patton's blog post introducing story mapping

http://jpattonassociates.com/the-new-backlog/

Goal Modelling for Requirements Decision

Goal Modelling

Key idea: Use stakeholders' goals as key abstraction for eliciting, documenting, and analyzing requirements

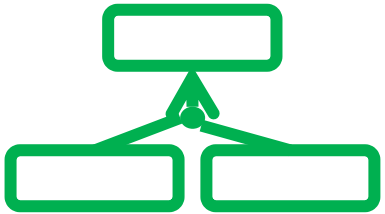
Many goal modelling frameworks and notations:

- **KAOS:** focus on precise formulation and analysis of goals, requirements and assumptions.
- **i*, Tropos:** focus on visualising and analysing complex relations between stakholders, their goals, tasks and resources.
- **User Requirements Notation**: combines goal modelling, use case maps and process models (standard of the International Telecommunication Union)
- **ArchiMate motivation and strategic views**: goal modelling for enterprise architecture (standard from The Open Group)
- **SysML**: focus on relations between system requirements and software requirements

Goal-Feature Mapping: basic concepts

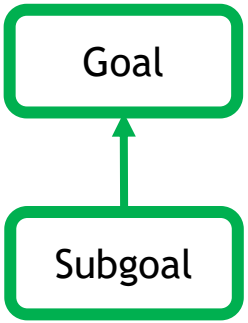
Simplest form of goal modelling for the requirements decision phase.

Something a stakeholder wants to be true of the world

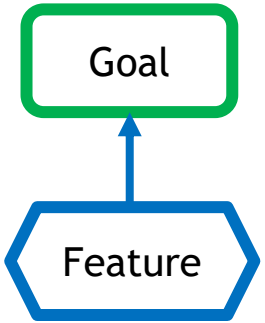


Goals can be AND-refined into subgoals

Satisfying all subgoals imply satisfying the parent goal.



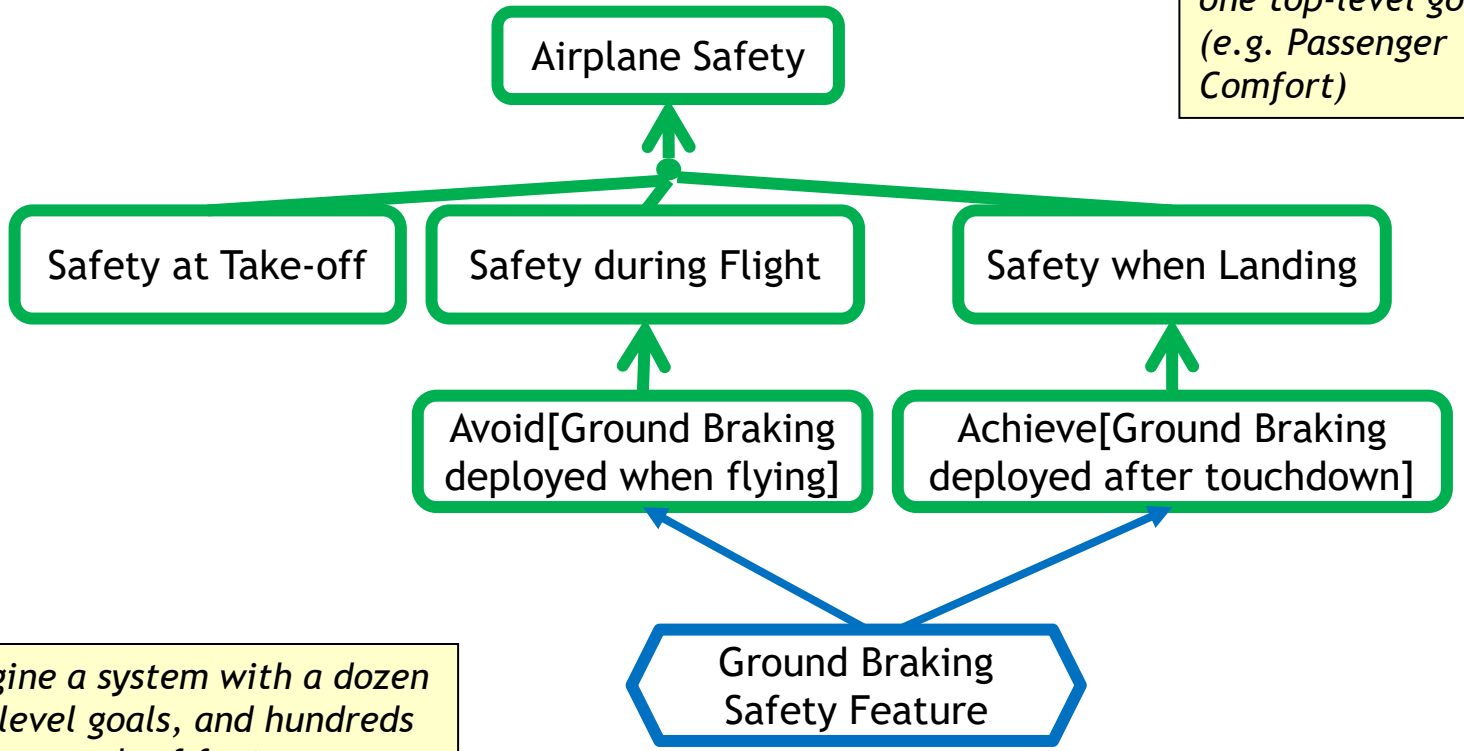
The subgoal contributes to the parent goal



The feature contributes to the goal

Example 1: Airplane Safety

Note: A goal model can have more than one top-level goal (e.g. Passenger Comfort)

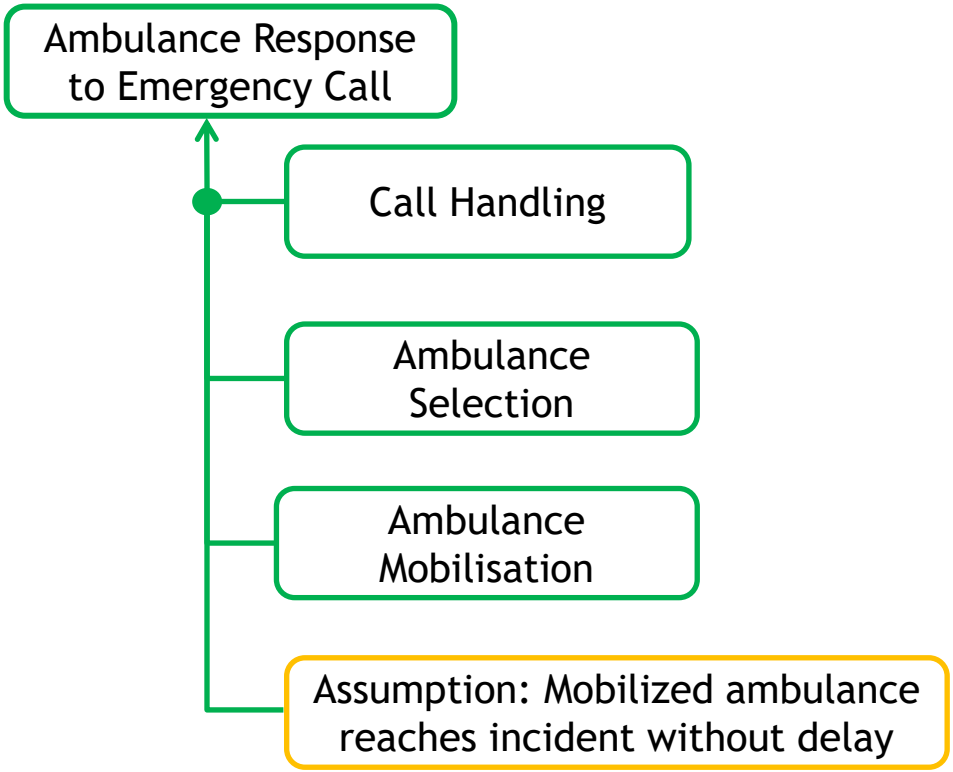


Imagine a system with a dozen top-level goals, and hundreds or thousands of features.

When to use Goal-Feature Maps

1. To break down goals into subgoals, and discover features one subgoal at a time (separation of concerns).
2. To relate features to goals they contribute to
   - This helps understanding stakeholders' rationale for a feature and managing scope creep.
   - A goal-feature map can be created bottom-up (from features to goals) as well as top-down (from goals to features).
3. To structure request for proposal (RFP)
   - RFP structure (table of content) = goal refinement structure

Example 2: Ambulance Dispatching System

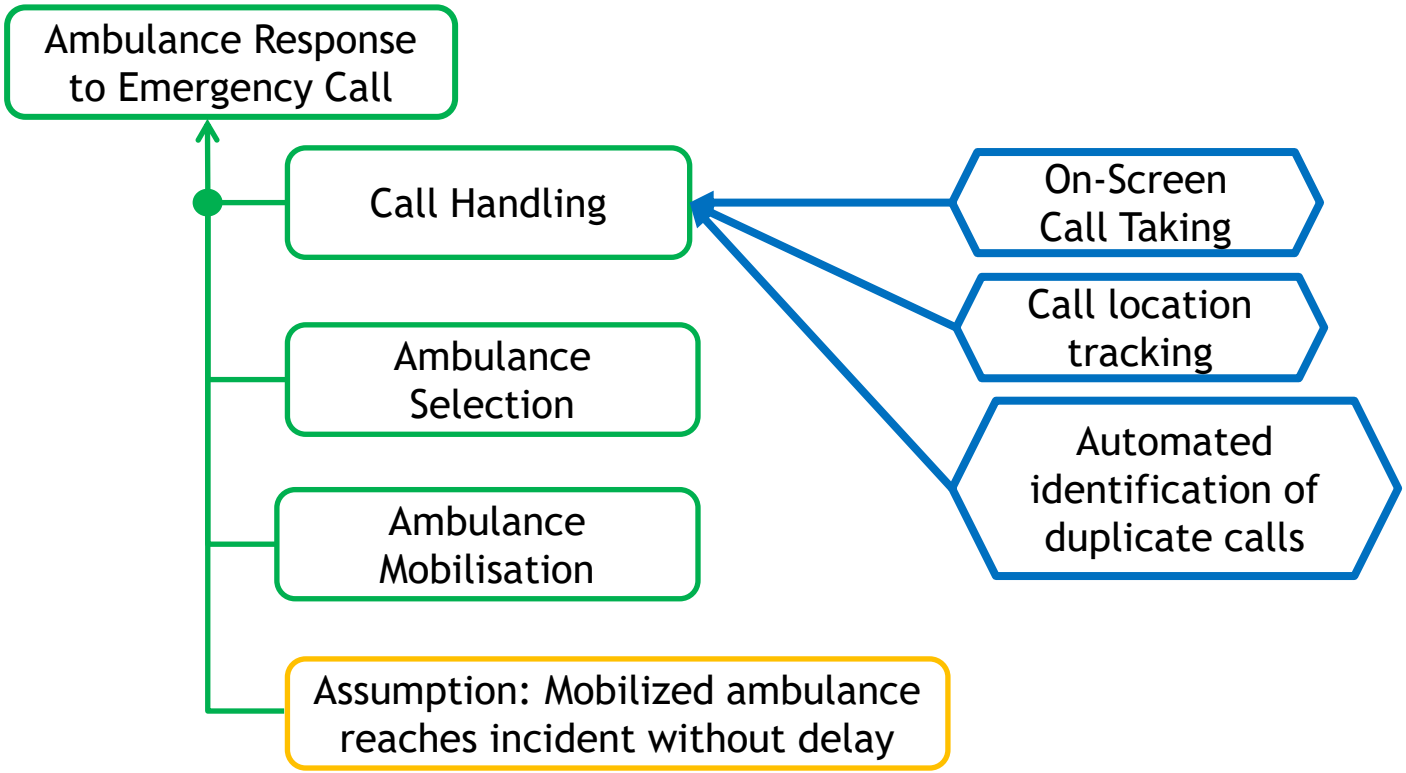


Goal Definition

Think of each goal is a self-contained problem. A goal is not defined by its subgoals.

Goal	Definition
Ambulance Response To Emergency Call	An ambulance must arrive at incident within 8 minutes after the first call for category A incidents and within 14 minutes for category B incidents.
Call Handling	All emergency call must be responded to within 10 seconds. The nature of the incident, its location, and other details should ideally be identified within 1 minute.
Ambulance Selection	Once the incident details and location are known, the nearest suitable available ambulance should be identified and allocated to the incident.
Ambulance Mobilization	Once an ambulance is allocated to an incident, the ambulance crew should be mobilized to the incident. The time between received the first call and the ambulance mobilization should ideally be less than 3 minutes.

Example of Features supporting Call Handling

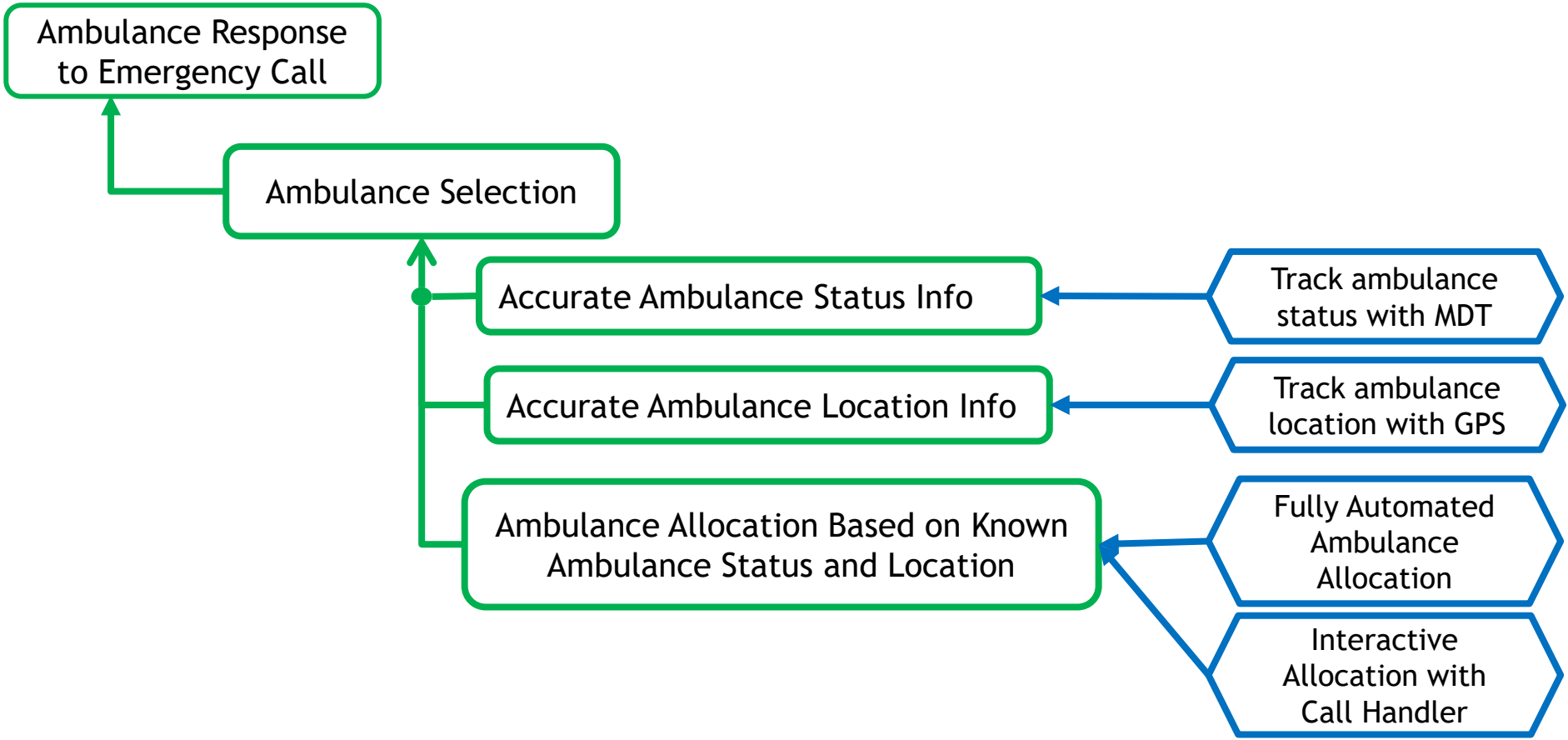


Feature description

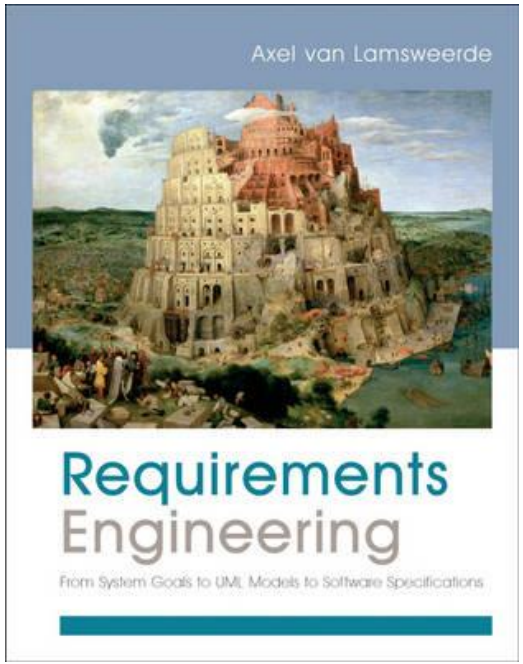
Give each feature a brief description. Detailed requirements will be worked out later.

Feature	Brief description
On-Screen Call Taking	allows call handlers to fill in an incident form containing information about the incident reported by the caller.
Automated Caller Location Detection	supports the call taker in identifying the incident location by automatically detecting the caller location when possible.
Identification of Duplicate Calls	identifies duplicate calls and supports the call handler in dealing with such calls by providing the latest information about incident details and status.

More examples for the ambulance dispatching system



Goal modelling reference



Chapter 7. Goal-Oriented Requirements Engineering

Chapter 8. Modelling Systems Objectives with Goal Diagrams

The goal-feature mapping technique in this lecture is a simplified variant of the KAOS goal modelling framework described in the book.

Techniques

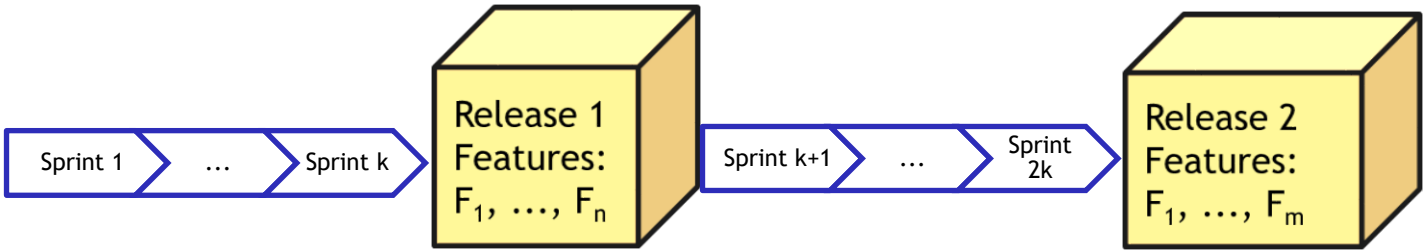
1. Impact Mapping
2. User Story Mapping
3. Goal Modeling
4. Strategic Release Planning

Strategic Release Planning

Software releases

Software release = a version of the software that is deployed to its users or released to its customers.

- A release is composed of a set of features (new or improved functionalities and qualities)



A release might be developed in multiple sprints (short iterations).

Release planning

Release planning = deciding what features to build for upcoming releases (i.e. planning for several months)

- Sprint planning = deciding what user stories (small unit of work) to build in next sprint (i.e. for the next 2-3 weeks).

E.g. Release plan for ambulance dispatching system

Release 0.1	Call encoding
Release 0.2	Automated ambulance location tracking
Release 0.3	Ambulance status update through MDT (pilot) Ambulance mobilisation through MDT (pilot)
Release 0.4	Automated ambulance allocation (pilot)
Release 1	Scale pilot to whole city

Note: more useful than traditional requirements prioritization (e.g. “MoSCoW prioritization”)

Release planning Factors

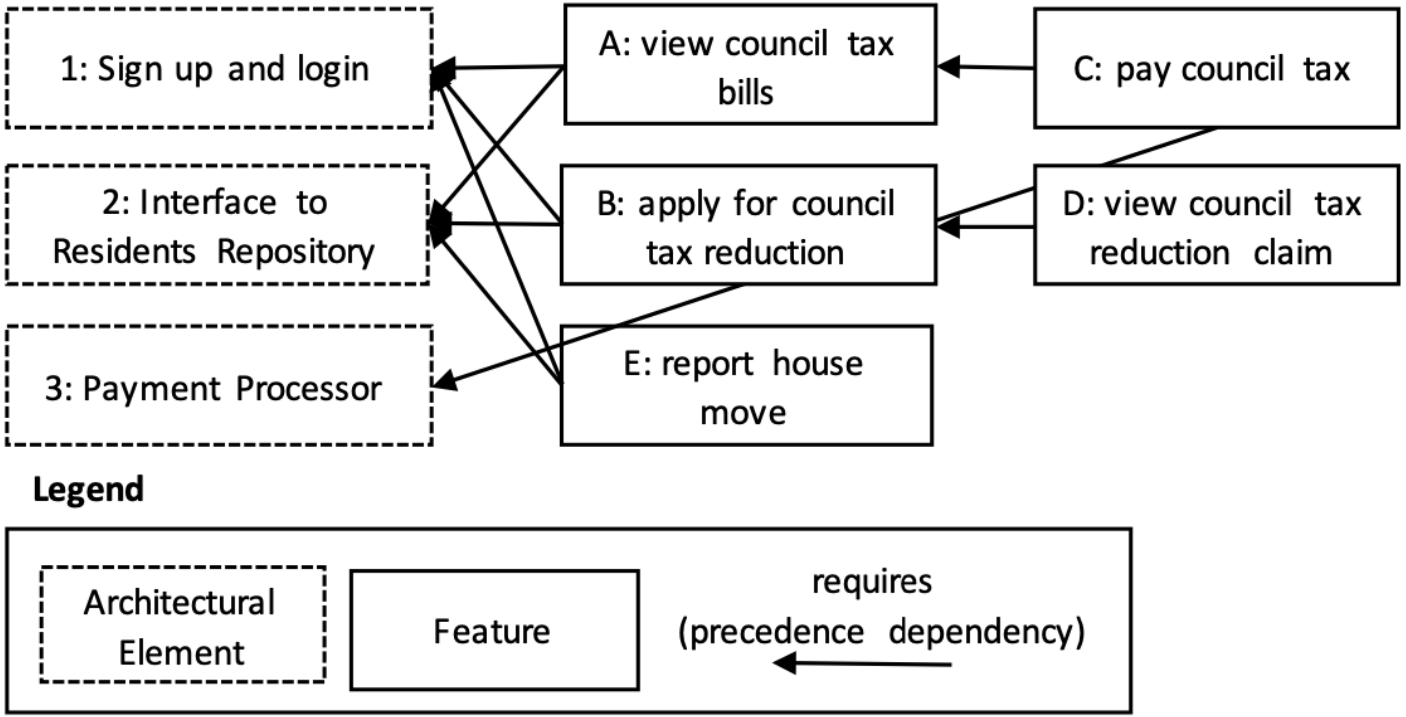
Decisions typically based on

- dependencies between features
- estimated effort (e.g. man-days or story points)
- estimated value (e.g. £ or value points)
- team capacity in each iteration

and other factors such as time constraints, stakeholder importance, etc.

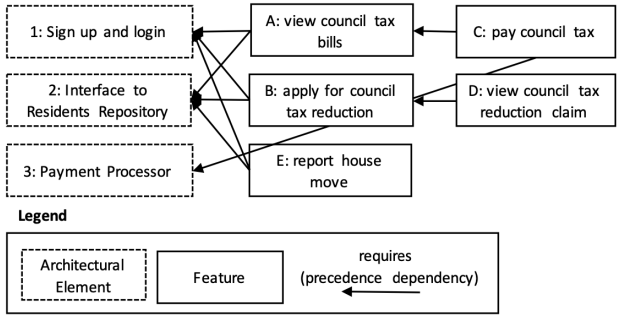
Inputs: Product backlog, dependencies & estimates

Partial product backlog and dependencies for a local gov. online services



Inputs: Product backlog, dependencies & estimates

Partial product backlog and dependencies for a local gov. online services



Feature/Arch. Element	Effort	Value
A View council tax bills	10	20
B. Apply for council tax reduction	10	20
C. Pay council tax	20	20
D. View council tax reduction claim	5	10
E. Report house. move	10	30
1. Sign up and login	10	0
2. Interface to resident repository	20	0
3. Payment Processor	30	0

Effort in story points; value in business value points.

Output: Release Plan

Release	Features	Effort	Value	Cumulative Value
1.	• Sign up and login • Interface to resident repository • View council tax bills	40	20	20
2.	• Report house move • Payment processor	40	30	50
3.	• Pay council tax • Apply for council tax reduction • View council tax reduction claim	35	50	100

Tool support generating **optimal** release plans and choosing a **preferred** plan

- EVOVLE-II/ReleasePlanner
- BEARS (Olawole Oni PhD thesis, former MSc SSE student at UCL)

Release Planning: Benefits

Introducing structure to the release planning process reduces the problems of “intuitive” release planning:

- lack of clarity about decision objectives and candidate features;
- late identification of feature dependencies;
- difficulty in engaging key stakeholders in the decision process;
- uncontrolled interference of politics and self-serving interests.

Systematic release planning processes have also been shown to be faster because more efficient.

Release Planning: Challenges

- Adequate identification of goals, features and feature dependencies
- Dealing with conflicts between multiple goals and diverse interests of different stakeholder groups
  - Unlike our example, value is usually multi-criteria
- Accuracy and relevance of effort and value estimates
  - Are story points and value points estimates reliable?
- Dealing with uncertainty about feature costs and values
  - requires a culture comfortable with uncertainty and not confusing estimates with targets

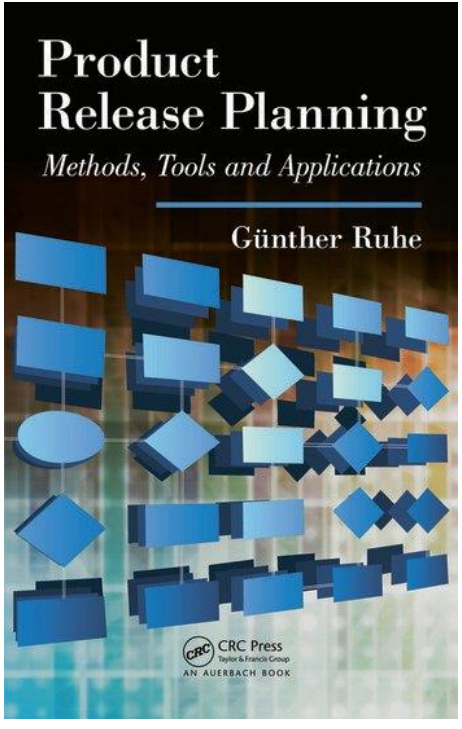
References and further reading

For a brief overview of release planning and dealing with uncertainty

Analysing Uncertainty in Software Release Planning: A Method and Experiment for Fixed-Date Release Cycles

OLAWOLE ONI and EMMANUEL LETIER, Department of Computer Science, University College London, United Kingdom

For a more detailed description of release planning and the EVOVLE method (available online through the UCL library)



Summary

What you need to know

1. Important to focus on goals, not just features.
   - Need to fight against a natural tendency to discuss features while being vague about goals
   - Two techniques: goal modelling and impact mapping
   - Know how to create goal-feature maps (see exercise)
2. We cannot be certain of a feature's real value and impact on stakeholders' goals before it is used.
   - We decide by estimating future impacts; we will be able to evaluate actual impacts once the feature is used.
3. Release planning = deciding what features to build in future releases
   - based on features' estimated cost and value.
   - For large projects, a structured process improves release planning discussions and speed.