Intelligent Software Engineering

Requirements Engineering

Zhilei Ren

Dalian University of Technology

September 25, 2025

Definition of Requirements Engineering

Requirements engineering is an interdisciplinary function that mediates between the domains of the acquirer and supplier or developer to establish and maintain the requirements to be met by the system, software or service of interest. Requirements engineering is concerned with discovering, eliciting, developing, analyzing, verifying (including verification methods and strategy), validating, communicating, documenting and managing requirements[1].

[1] ISO/IEC/IEEE 29148 Systems and software engineering —Life cycle processes – Requirements engineering

As Proposed by the Project Sponsor

As Specified in the Project Request

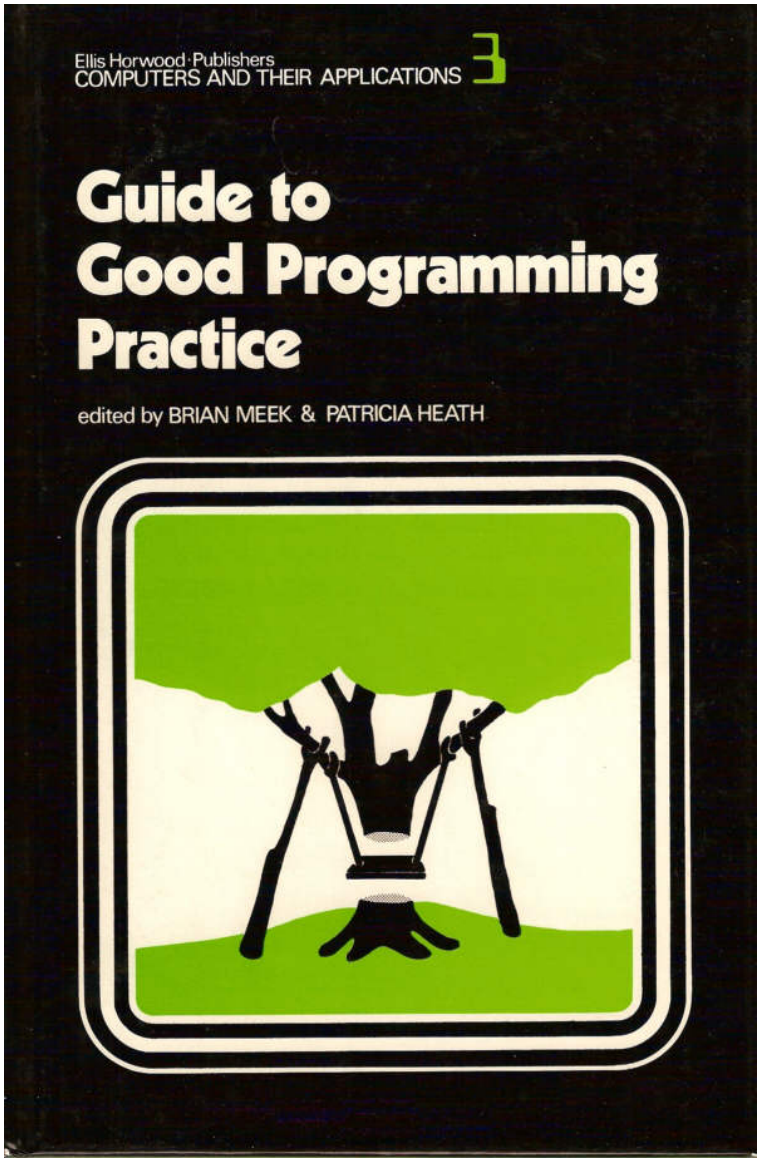
As Designed by the Senior Systems Analyst

As Produced by the Programmers

As Installed at the User's Site

What The User Wanted

Guide to Good Programming Practice, 1979



Research Topics in Requirements Engineering

1. Requirements Classification
2. Requirements Prioritization
3. Feature Model Optimization
4. Prototype Generation

Techniques for Requirements Engineering Research

Next Release Problem

Given:

- A set of software requirements $R = \{r_1, r_2, \ldots, r_n\}$,
- A set of customers $C = \{c_1, c_2, \ldots, c_m\}$,
- Each customer $c_j \in C$ requests a subset of requirements $R_j \subseteq R$ and provides a profit $p_j > 0$ if all requirements in R_j are satisfied,
- Each requirement $r_i \in R$ has an associated cost $\text{cost}(r_i) > 0$,
- A total available budget $B > 0$.

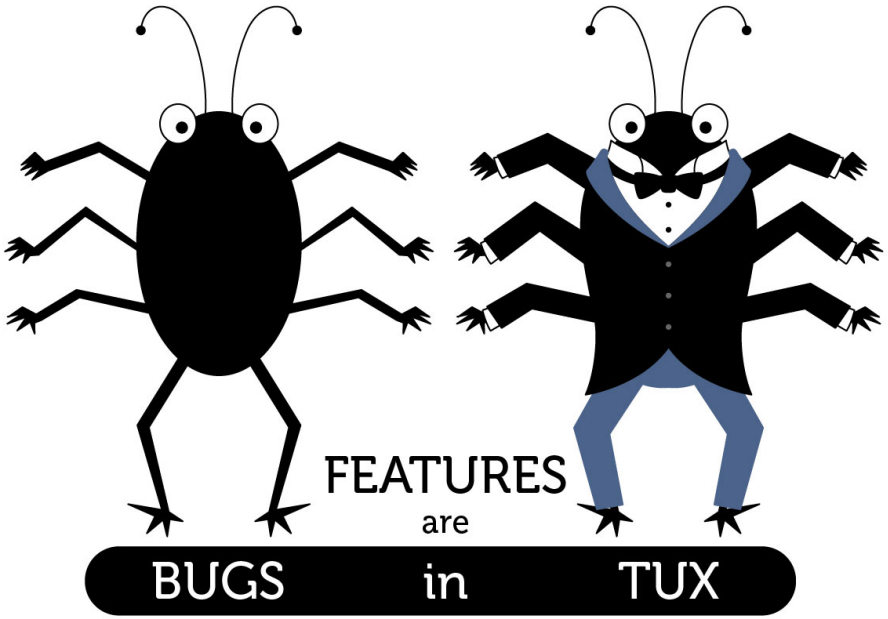
The goal is to select a subset of requirements $R' \subseteq R$ such that:

1. The total cost does not exceed the budget: $\sum_{r_i \in R'} \text{cost}(r_i) \leq B$,
2. The total profit is maximized:

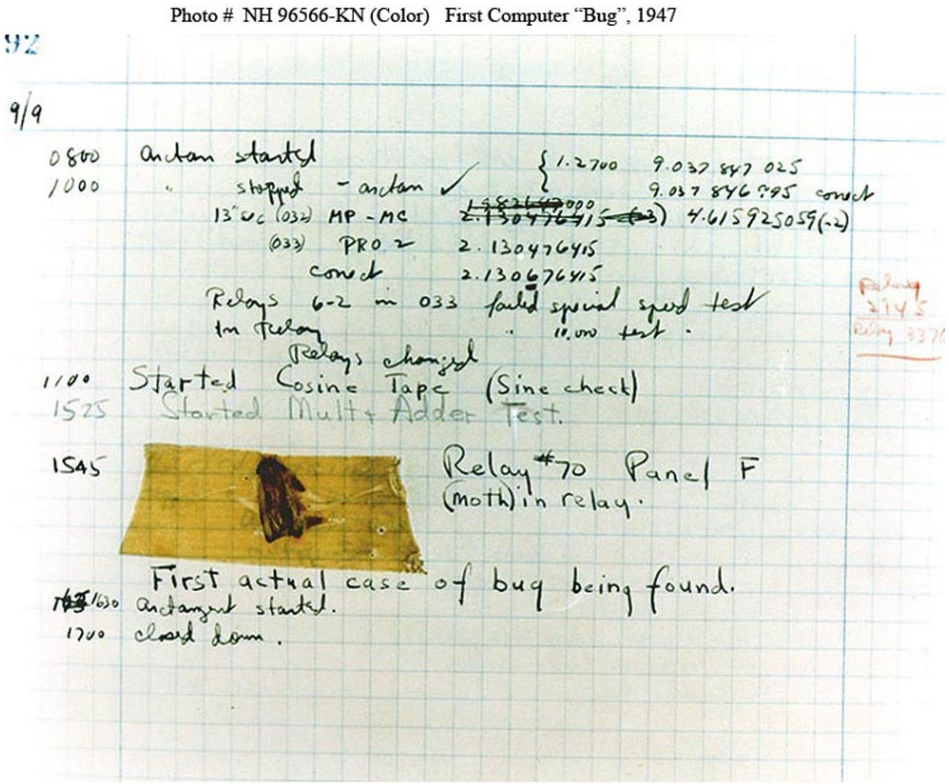
$$\max_{R' \subseteq R} \sum_{\substack{c_j \in C \\ R_j \subseteq R'}} p_j.$$

This is known as the **Next Release Problem (NRP)** and is a well-known NP-hard problem in requirements engineering and software release

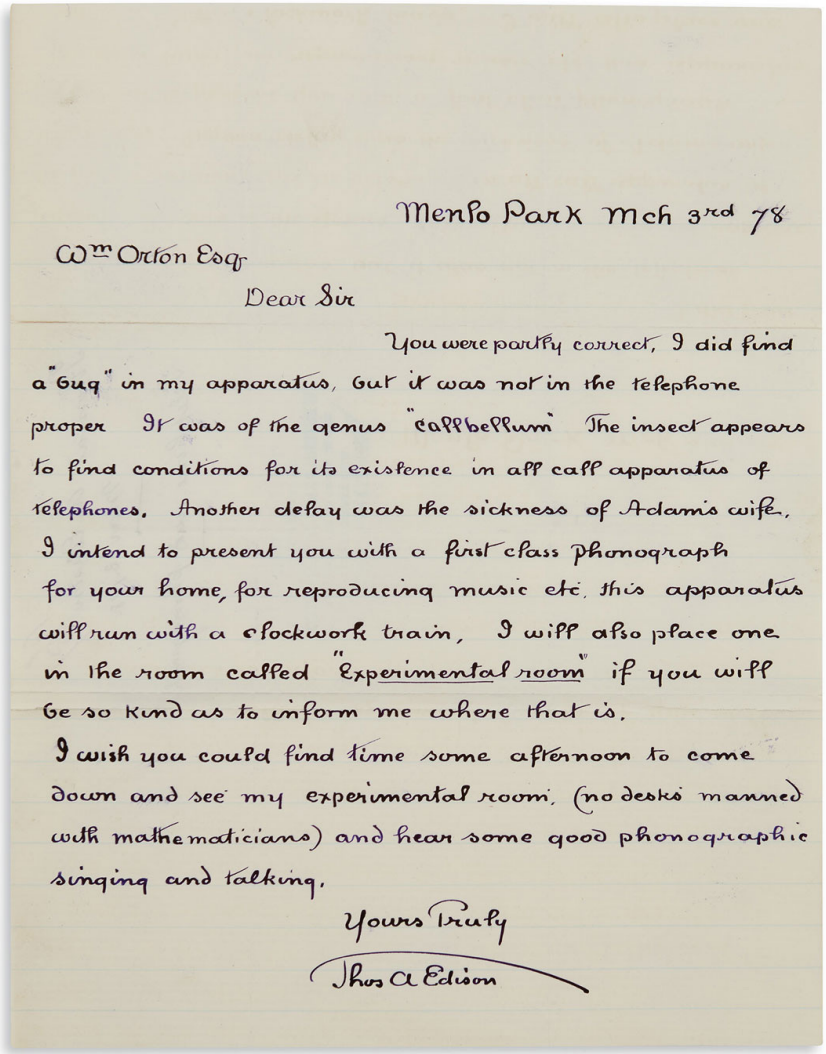
bug or feature?



The First Computer Bug



The First Computer Bug



Menlo Park Mch 3rd 78

Wm Orton Esqr

Dear Sir

You were partly correct, I did find a "bug" in my apparatus, but it was not in the telephone proper It was of the genus "callbellum" The insect appears to find conditions for its existence in all call apparatus of telephones. Another delay was the sickness of Adams's wife. I intend to present you with a first class phonograph for your home, for reproducing music etc. this apparatus will run with a clockwork train, I will also place one in the room called "Experimental room" if you will be so kind as to inform me where that is.

I wish you could find time some afternoon to come down and see my experimental room, (no desks manned with mathematicians) and hear some good phonographic singing and talking.

Yours Truly

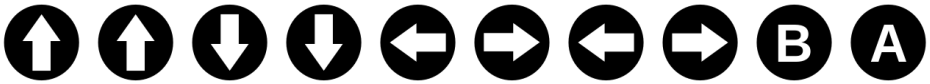
Thos A Edison

Combo

Combos were a design accident; lead producer Noritaka Funamizu noticed that extra strikes were possible during a bug check on the car-smashing bonus stage. He thought that the timing required was too difficult to make it a useful game feature, but left it in as a hidden one[2].

[2] https://en.wikipedia.org/wiki/Combo_(video_games)

彩蛋



Outline

What is PlantUML?

- **Definition:** Open-source tool for creating diagrams from plain text descriptions
- **Core Idea:** Uses simple, human-readable Domain Specific Language (DSL)
- **Foundation:** Java-based tool leveraging Graphviz for layout
- **Philosophy:** Focus on content rather than manual layout

  "PlantUML is a versatile component for quickly and directly creating diagrams."

Key Advantages

Advantage	**Description**
Version Control Friendly	Text files work with Git - enables change history, diffing, collaboration
Efficiency & Speed	Faster than manual graphical editing, especially for complex diagrams
Maintainability & Consistency	Easy updates and consistent styling with themes
Automation & Integration	Integrates with documentation pipelines, build systems, CI/CD

UML Diagrams Supported

- Sequence Diagram
- Use Case Diagram
- Class Diagram
- Activity Diagram
- Component Diagram
- Deployment Diagram
- State Diagram
- Object Diagram

Visual Example:

[Diagram placeholder]

Beyond UML Diagrams

- Architectural Diagrams (C4 model)
- Entity Relationship Diagrams (ERD)
- Wireframes / UI Mockups (salt library)
- Gantt charts for project management
- Mind Maps for brainstorming
- JSON/YAML visualization
- Network diagrams

How PlantUML Works

1. **Write:** Create text file (.puml) with PlantUML syntax
2. **Process:** Java processor parses text, converts to Graphviz DOT language
3. **Render:** Layout engine generates final image
4. **Output:** Get image in desired format (PNG, SVG, etc.)

Text $\rightarrow$ **PlantUML** $\rightarrow$ **Graphviz** $\rightarrow$ **Diagram**

Syntax Example: Sequence Diagram

```
@startuml
actor User
participant "Web Browser" as Browser
participant Server

autonumber
User -> Browser: Enter URL
Browser -> Server: HTTP Request
Server -> Server: Process Request
Server --> Browser: Return HTML
Browser --> User: Display Page
@enduml
```

- `@startuml/@enduml`: Diagram boundaries
- `actor`, `participant`: Element declarations
- `->`, `-->`: Solid/dashed arrows
- `autonumber`: Automatic message numbering

Syntax Example: Use Case Diagram

```
@startuml
left to right direction
actor "Library User" as User
usecase "Borrow Book" as Borrow
usecase "Search Catalog" as Search

User --> Borrow
User --> Search
@enduml
```

- `left to right direction`: Layout control
- `actor`, `usecase`: Actor and use case definitions
- `->`: Connection arrows

Getting Started with PlantUML

Online Servers (Quick Start):

- Official server: plantuml.com
- Community site: planttext.com
- No installation required

Local Installation (Recommended):

- Prerequisites: Java JRE + Graphviz
- IDE Plugins: VSCode, IntelliJ, Eclipse
- Command Line: Use `plantuml.jar`

Summary & Key Takeaways

- **Why PlantUML?** Version control, automation, efficiency
- **Text-Based:** Simple, readable language for diagrams
- **Wide Support:** Comprehensive UML + additional diagram types
- **Easy Integration:** Fits modern development workflows
- **Quick Start:** Online editors local integration

Embrace efficient diagram creation and maintenance!

Thank You & Questions

Resources

- Official Website: plantuml.com
- Online Demo: plantuml.com/plantuml
- Documentation: plantuml.com/guide

Questions?