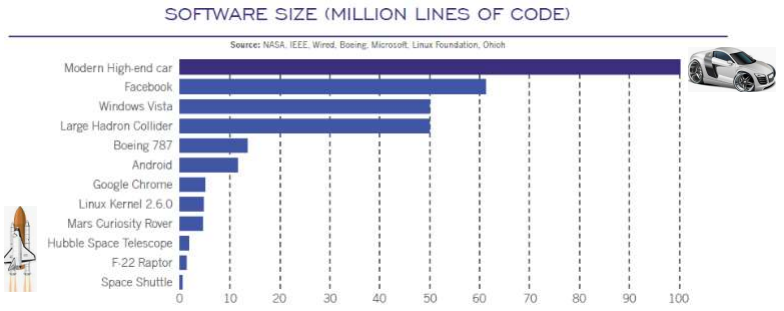
Software Engineering

Chapter 1 Introduction

Objectives

- Professional software development
- Software engineering ethics
- Case study



Software engineering

- Essential for government, society, and national and international businesses and institutions.
- Concerned with theories, methods and tools for software development.

Software Project Failure Factors

1. **Increasing system complexity**
   - Complex software systems implemented in the large number of source files

In the IoT age, in which we are starting to connect absolutely everything—including toothbrushes—system complexity is skyrocketing.


DesignNews

Serving the 21st Century Design Engineer

Automation & Motion Control | Design Hardware & Software | Electronics & Test | Materials & Assem

Home

The 5 Biggest Challenges Facing Embedded Software Developers in IoT

The traditional, disconnected developer is finding that there are several new challenges that need to be addressed in order to be successful.

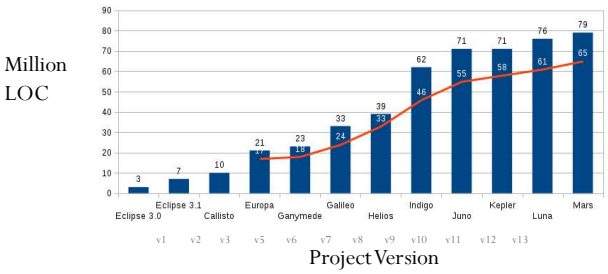
by: Jacob Beningo in Embedded Systems Conference - Minneapolis, IoT, Cyber Security, Design Hardware & Software on August 03, 2018

Developing embedded software is not as simple as it used to be. Creating a standalone device was and still often is challenging for many development teams. In the IoT age, in which we are starting to connect absolutely everything—including toothbrushes—system complexity is skyrocketing. The traditional, disconnected developer is finding that there are multiple new challenges that need to be addressed in order to achieve success:

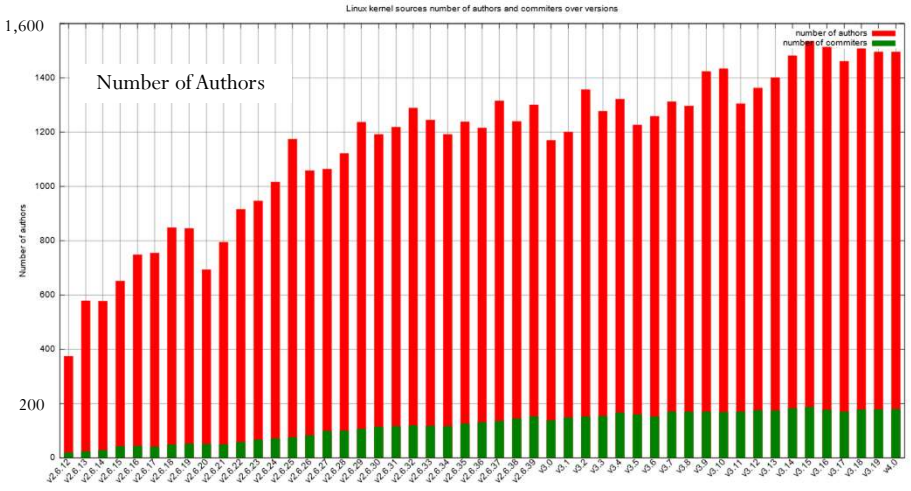
2. **Failure to use software engineering methods**
   - Writing programs without software engineering methods.
   - Getting more expensive and less reliable.

Software Systems Become Complex

- Evolution of the Eclipse platform
  - Open source projects released by Eclipse project teams.
  - 350 contributors
  - 65 million lines of code
  - 80 separate open source projects.

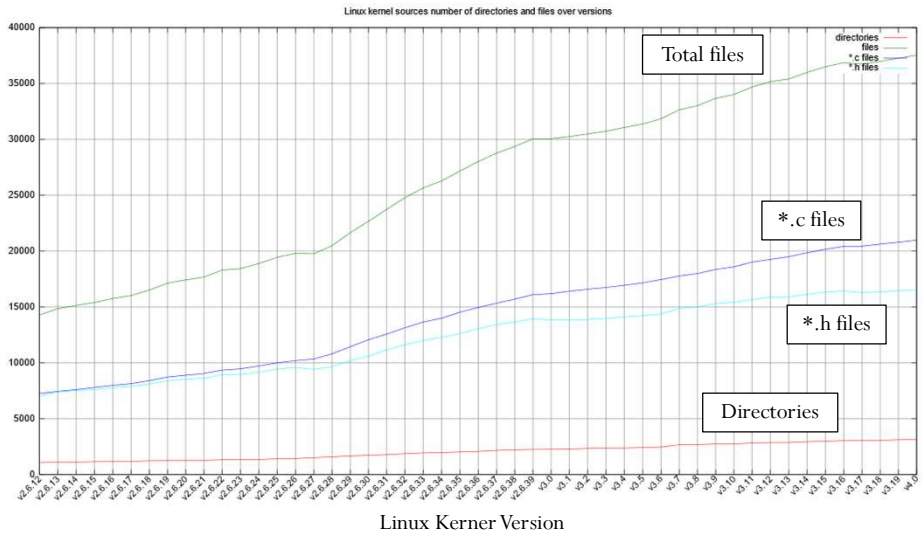


Evolution of Linux Kernel
Increased Number of Software Developers



7

Increased Number of Source Files in Linux



8

Objectives

- Professional software development
- Software engineering ethics
- Case study

9

Frequently Asked Questions about Software Engineering

10

What is Software Engineering

Question	Answer
What is software?	Computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.
What are the attributes of good software?	Good software should deliver the required functionality and performance to the user and should be maintainable, dependable and usable.
What is software engineering?	Software engineering is an engineering discipline that is concerned with all aspects of software production.
What are the fundamental software engineering activities?	Software specification, software development, software validation, and software evolution.

What is Software Engineering

Question	Answer
What is the difference between software engineering and computer science?	Computer science focuses on theory and fundamentals; software engineering is concerned with the practicalities of developing and delivering useful software.
What is the difference between software engineering and system engineering?	System engineering is concerned with all aspects of computer-based systems development including hardware, software and process engineering. Software engineering is part of this more general process.

What are Key Challenges in Software Engineering

Question	Answer
What are the key challenges facing software engineering?	Coping with increasing diversity, demands for reduced delivery times and developing trustworthy software.
What are the costs of software engineering?	Roughly 60% of software costs are development costs, 40% are testing costs. For custom software, evolution costs often exceed development costs.
What are the best software engineering techniques and methods?	While all software projects have to be professionally managed and developed, different techniques are appropriate for different types of system. For example, games should always be developed using a series of prototypes whereas safety critical control systems require a complete and analyzable specification to be developed. You can't, therefore, say that one method is better than another.

13

What are Essential Attributes of Good Software

Product characteristic	Description
Maintainability	• Software should be written in such a way so that it can evolve to meet the changing needs of customers. • This is a critical attribute because software change is an inevitable requirement of a changing business environment.
Dependability & Security	• Software dependability includes a range of characteristics including reliability, security and safety. • Dependable software should not cause physical or economic damage in the event of system failure. • Malicious users should not be able to access or damage the system.

14

What are Essential Attributes of Good Software

Product characteristic	Description
Efficiency	• Software should not make wasteful use of system resources such as memory and processor cycles. • Efficiency therefore includes responsiveness, processing time, memory utilization, etc.
Acceptability	• Software must be acceptable to the type of users for which it is designed. • This means that it must be understandable, usable and compatible with other systems that they use.

15

Software Engineering Processes & General Issues

16

Software Process Activities

- Software **specification** – customers and engineers define a software system that is to be produced under the constraints on its operation.
- Software **development** – software is designed and programmed.
- Software **validation** – software is checked to ensure that it is what the customer requires.
- Software **evolution** – software is modified to reflect changing customer and market requirements.

General Issues in Software Development

1. **Heterogeneity**
   - Increasingly, systems are required to operate as **distributed** systems **across networks** that include different types of computer and mobile devices.

2. **Business and social change**
   - Business and society are **changing** incredibly quickly as emerging economies develop and new technologies become available.
   - They need to be able to change their existing software and to rapidly develop new software.

From a software engineering viewpoint, IoT applications have two main characteristics.
*The first is distribution over a large range of processing nodes.
*The second is high heterogeneity of the processing nodes and the protocols used between them.

General Issues in Security & Scalability

3. **Security and trust**
   - All aspects of our lives, such as privacy and personal information.
   - Need to develop trust-worthy software system.
   - Essential that we can trust that software.

4. **Scalability**
   - Software has to be developed across a wide range of scales
   - Embedded systems in portable devices
   - Internet-scale, cloud-based systems

Software Engineering Diversity

- There are **many different types** of software system.
- There is **no universal** set of software **techniques** that is applicable to all of these systems.
- The software engineering methods and tools used depend on
  - The **type** of application being developed,
  - The **requirements** of the customer, and
  - The **background** of the development team.

Software Application Types

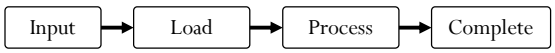
Application Types

- **Stand-alone applications**
  - These are application systems that run on a local computer, such as a PC.
  - They include all necessary functionality and do not need to be connected to a network.
- **Interactive transaction-based applications**
  - Applications that execute on a remote computer and are accessed by users from their own PCs or terminals.
    - E.g., web applications such as e-commerce applications.

Application Types (Cont.)

- **Embedded control systems**
  - These are software control systems that control and manage hardware devices.
- **Batch processing systems**
  - These are business systems that are designed to process data in large batches.
  - They process large numbers of individual inputs to create corresponding outputs.

Batch Processing

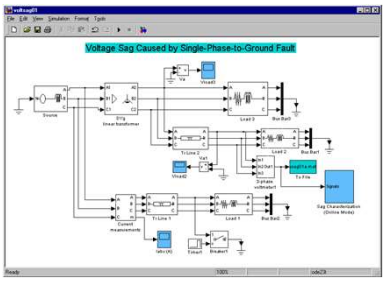



```
#! /bin/sh
for file in *; do
  if [[ -f $file ]]; then
        # process "$file"
  fi
done
```


Application Types (Cont.)

- **Entertainment System**
  - Most of these systems are games. The quality of the user interaction is the most important distinguishing characteristic of entertainment systems.
- **Systems for modelling and simulation**
  - These are systems that are developed by **scientists** and engineers to **model** physical processes or situations, which include many, separate, interacting objects.

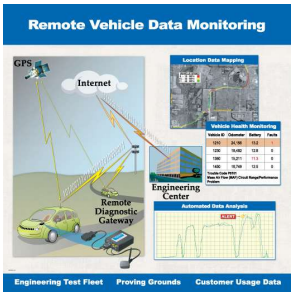
Power system modeling and simulation



Data Collection Software Systems

- **Data collection systems**
  - These are systems that collect data from their environment using a set of **sensors** and send that data to other systems for processing.

… a regular vehicle to be converted into a 'computer on wheels' - a smarter and more connected machine.

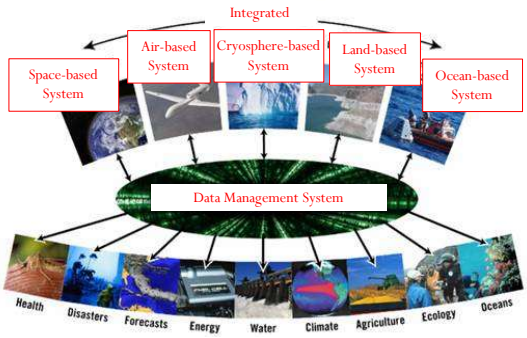


25

Systems of Systems

- **Systems of systems**
  - These are systems that are composed of a number of other software systems.

Observing Systems
Global Earth Observation System of Systems



Health · Disaster · Forecasts · Energy · Water · Climate · Agriculture · Ecology · Oceans

26

Objectives

- Professional software development
- Software engineering ethics
- Case study

Issues of Professional Responsibility

- **Computer misuse**
  - Software engineers should not use their technical skills to misuse other people's computers.
  - Computer misuse ranges from relatively trivial to extremely serious (dissemination of viruses).
  - Virus
    - Erase files
    - Scramble data on a hard disk
    - Cause erratic screen behaviour
    - Halt the system

Programming for Secure Software System

Example

- Make your classes unserializeable.
- Serialization allows attackers to view the internal state of your objects, even private portions.
- To prevent this, add this method to your classes:

```
private final void writeObject(ObjectOutputStream out)
                                throws java.io.IOException {
   throw new java.io.IOException("Object cannot be serialized");
}
```


Removing Security Warnings

- **Compiler checks help find a number problems, including security issues**
  - Effort to make existing programs conform to all the checks

```
system > gcc -Werror -Wall \
    -Wmissing-prototypes -Wmissing-declarations \
    -Wbad-function-cast \
    -Wformat-security -Wmissing-format-attribute \
    -Winline
```

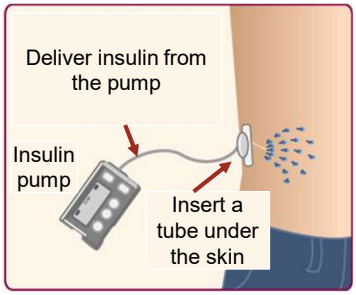
Objectives

- Professional software development
- Software engineering ethics
- Case study

31

Case Studies

1. **An insulin pump control system**
   - An embedded system in an insulin pump used by diabetics to maintain blood glucose control.

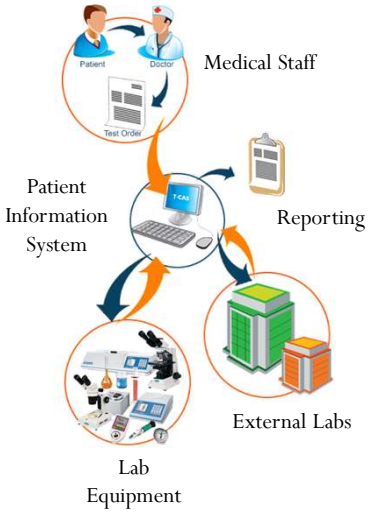


32

Information System

2. A patient information system for mental health care

- A system used to maintain records of people receiving care for mental health problems.



33

Data Collection System

3. A wilderness weather station

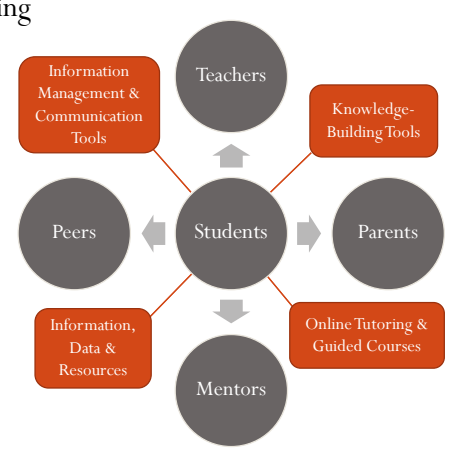
- A data collection system that collects data about weather conditions in remote areas.

34

Digital Learning Environment System

4. A digital learning environment for schools

- A system to support learning in schools



35

Software Specification (Insulin Pump Control Systems)

- **Software Specification**
  1. Collect data from a blood sugar sensor and calculates the amount of insulin required to be injected.
  2. Perform calculation based on the rate of change of blood sugar levels.
  3. Send signals to a micro-pump to deliver the correct dose of insulin.

- **Safety-critical system**
  - Low blood sugars can lead to brain malfunctioning, coma, and death
  - High-blood sugar levels can cause long-term consequences such as eye and kidney damage.

Table. Predominant Medication Error Event Types Associated with the Use of Insulin (N = 2,057, 76.6%), January 2008 to June 6, 2009

EVENT TYPE	NUMBER	% OF TOTAL REPORTS (N = 2,685)*
Dose omission	662	24.7%
Wrong drug	374	13.9%
Wrong dose/overdosage	348	13%
Other (specify)	309	11.5%
Extra dose	227	8.5%
Wrong dose/underdosage	137	5.1%

* Sum of percentages exceeds 76.6% due to rounding.

36

High-Level Functional Requirements

- **Functional Requirement Specification**
  1. The system shall be available to deliver insulin when required.
  2. The system shall perform reliably and deliver the correct amount of insulin to counteract the current level of blood sugar.

- The system must therefore be designed and implemented to ensure that the system always meets the requirements.

Medication error event types associated with the use of insulin

http://www.patientsafetyauthority.org

- Dose omission: 24.7%
- Wrong dose / over-dosage: 13.9%

Batch Processing Example

1. Open a terminal to access your EC2 server

2. Download an example source code
   - **Canvas > Pages >**
     - **"Download Batch Proc Example on Command Line"**

3. Run a shell script
   - `ubuntu> tar -xvzf example-batch-system.tar.gz`
   - `ubuntu> cd example-batch-system/`
   - `ubuntu> ./batch-process.sh`

Input & Output

- batch-process.sh

```
#!/bin/bash
rm -fr eclipse-src/

git clone https://github.com/eclipse/eclipse.jdt.core.git eclipse-src

./batch-process2.sh ./eclipse-src/TARGET SOURCE DIRECTORY
```

- batch-process1.sh (a simplified version of batch-process2.sh)

```
#!/bin/sh
for f in "$1"/*; do
  if [ -f "$f" ]; then
    FILE_NAME="$(basename "${f}")"
    SIZE="$(du -sh "${f}" | cut -f1)"
    echo "File name: $FILE_NAME"
    echo "File size: $SIZE"
  fi
done
```

```
Run this shell script
$> ./batch-process1.sh ./eclipse-src/org.eclipse.jdt.apt.core/src/org/eclipse/jdt/apt/core/util
```


Utility

- Get filename from a given path:
  - `ubuntu> basename "/Your/Path/A.java"`

    `A.java`
- Display **disk usage** statistics
  - `ubuntu> du –sh /Your/Path/A.java`

    `4.0K` (Tab) `/Your/Path/A.java`

du options

- -h "Human-readable" output.
- -s Display each specified file.

- Partition selected portions of each line
  - `ubuntu> cut -d ',' –f 1 names.csv`

    `John`
    `Arthur`
    `George`

names.csv

```
John,Smith,34,London
Arthur,Evans,21,Newport
George,Jones,32,Truro
```

- The cut command uses tab as a default field delimiter but can also work with other delimiter by using -d option.

Multiple Commands

- Display disk usage statistics
  - `ubuntu> du –sh names.csv`

```
4.0K        names.csv
```

Tab

- Combine commands via pipe
  - `ubuntu > du -sh names.csv | cut -f1`

```
4.0K
```

du options

- -h "Human-readable" output.
- -s Display each specified file.

- The cut command uses tab as a default field delimiter but can also work with other delimiter by using -d option.