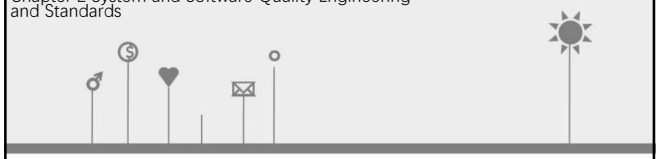
Software College Northeastern University

Software Quality Assurance and Testing

Chapter 2 System and Software Quality Engineering and Standards



wuchenni@qq.com

Contents

Chapter 2

System & SQ Engineering and Standards

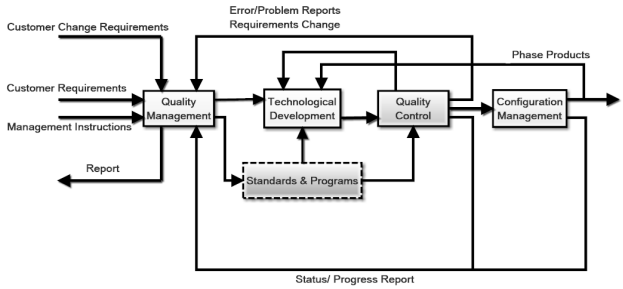
2.1 SQC Concepts and Methods

- What is Quality Control?
  - Those quality assurance actions that provide a means to control and measure the characteristics of an item, process or facility to established requirements.
  - The operational techniques and the activities that sustain a quality of product or service that will satisfy given needs; also the use of such techniques and activities.

2.1 SQC Concepts and Methods

- What is Quality Control?
  - Quality control activities are work product oriented.
  - They measure the product, identify deficiencies, and suggest improvements.
  - The direct results of these activities are changes to the product.
  - These can range from single-line code changes to completely reworking a product from design.
  - They evaluate the product, identify weaknesses and suggest improvements.
  - Testing and reviews are examples of QC activities since they usually result in changes to the product, not the process.
  - QC activities are often the starting point for quality assurance (QA) activities.

2.1 SQC Concepts and Methods



basic structure of software quality control system

2.1 SQC Concepts and Methods

- Basic approaches
  - Goal question metric approach
  - Risk management approach
  - PDCA quality control approach

2.1 SQC Concepts and Methods---GQM

- The Goal-Question-Metric (GQM) methodology was originally developed by V. Basili and D. Weiss and then significantly extended by D. Rombach.
- GQM is directed at the development of a set of corporate, division and project goals related to different business measures such as customer satisfaction, quality, financial progress, technical performance, etc..

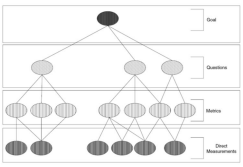
2.1 SQC Concepts and Methods---GQM

- **GQM** approach is a systematic way to tailor and integrate an organization's objectives into measurement goals and refine them into measurable values.
- It helps in systematic derivation of measurement plans.

2.1 SQC Concepts and Methods---GQM

- The GQM process
  - developing a set of corporate, division and projects goals for productivity and quality, e.g., customer satisfaction, improved quality
  - generating questions that define those goals as completely as possible in a quantifiable way
  - specifying the measures needed to be collected to answer those questions and to track process and product conformance to the goals
  - developing mechanisms for data collection
  - collecting , validating and analyzing the data in real time to provied feedback to projects for corrective action and analyzing the data in a post mortem fashion to assess conformance to the goals and make recommendations for future improvements.

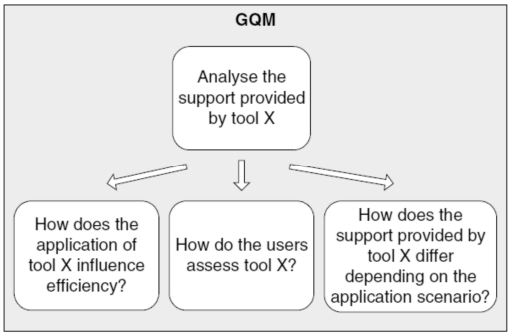
2.1 SQC Concepts and Methods---GQM



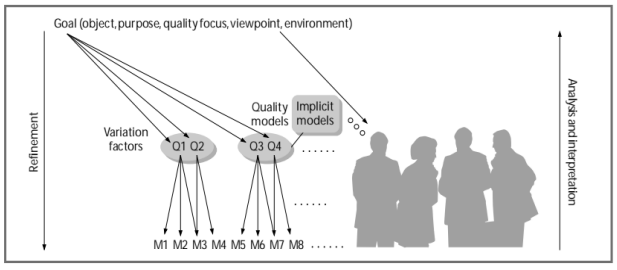
GQM paradigm

2.1 SQC Concepts and Methods---GQM

- Example-Assess the support provided by a tool X



2.1 SQC Concepts and Methods---GQM



The GOM approach to goal-oriented measurement

2.1 SQC Concepts and Methods---GQM

- Gain lift in many ways by implementing GQM
  - It supports project planning and control
  - It is used to determine strengths and weaknesses
  - It provides a rationale for the adoption and refinement of various software engineering techniques or methods
  - It allows assessment of the impact of changes in techniques and methods
  - It supports evaluation of both software processes and products

2.1 SQC Concepts and Methods---GQM

- Goals are identified to possess five attributes:
  - what is the object of interest
  - what is the purpose of studying the object of interest
  - what is the focus with regard to characteristics of the object of interest
  - who's perspective is to be supported by the goal
  - within which context or environment is the object to be studied.

2.1 SQC Concepts and Methods---GQM

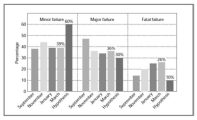
Object: Delivered product	**Purpose:** Better understanding	**Quality focus:** Reliability and its causes	**Viewpoint:** Software project team	**Environment:** Schlumberger RPS Project A

Quality focus	**Variation factors**
Number of failures • By severity (minor, major, fatal) • By detection (engineer, test group...) **Number of faults** • By life-cycle phase of detection • By modules **Cost for fixing faults (effort in hours)** • Cost by activity	**Process conformance** • Adherence to coding standards • Are the reviews done as prescribed in the process model? **Domain conformance** • Experience level of engineers **Attributes** • Complexity
Baseline hypotheses	**Impacts on baseline hypotheses**
Distribution of failures by severity • Minor 60% • Major 30% • Fatal 10% **Failure detection** • Engineer 10% • Test group 30% • OPCO 60% • Customer 0% **Faults by life-cycle phase of detection** • REQ: 5% • HLD: 10% • DD&IMP: 15% • Test: 70% **Top six fault-containing modules (ranked)** • DCT, DSS, MFT, MCF, MDS, TOT **Distribution of effort for fixing faults per introduced activities** • Requirements analysis/spec: 9.5 hours • High level design: 3.6 hours • Design and implementation 1.8 hours • Integration 1.8 hours • Evaluation and release: 3.6 hours	• Better process control results in: - Fewer failures - Fewer faults slipped through code review - Lower percentage of coding faults • Higher experience of software engineers results in fewer faults introduced • Better adherence to coding standards results in - Fewer faults in general - Less effort to locate and fix faults • Complex modules have more faults

2.1 SQC Concepts and Methods---GQM

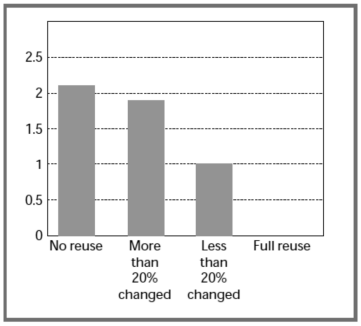
A sample question and corresponding metrics for the abstraction sheet

2.1 SQC Concepts and Methods---GQM



Trend of the severity of failures compared to baseline hypotheses

2.1 SQC Concepts and Methods---GQM



Fault density for the reuse categories.
The bars represent faults per thousand lines of source code.

2.1 SQC Concepts and Methods---GQM

- Armed with the measurement data, the project team was able to develop several rules of thumb:
  - The average fault density is 1.9 per thousand lines of source code.
  - The fault density for management functions is three times the fault density for dispensing functions.
  - The fault density for console functions is two times the fault density for dispensing functions.
  - The average effort needed to correct a failure in dispensing software is five times the effort needed to correct a failure in management functions.

2.2 SQC Models and Techniques---RM

- Risk definition
  - the effect of uncertainty on objectives (ISO)
  - the possibility of an unfortunate occurrence
  - the potential for realization of unwanted, negative consequences of an event
  - exposure to a proposition (e.g. the occurrence of a loss) of which one is uncertain
  - the consequences of the activity and associated uncertainties
  - uncertainty about and severity of the consequences of an activity with respect to something that humans value
  - the occurrences of some specified consequences of the activity and associated uncertainties
  - the deviation from a reference value and associated uncertainties.

2.2 SQC Models and Techniques---RM

- Why do we use various risk metrics?
  - To descrive or measure risk.
  - To make judgements about how large or small the risk is.
- Risk metrics/descriptions (examples)
  - The combination of probability and magnitude/severity of con- sequences.
  - A possibility distribution for the damage (for example a trian- gular possibility distribution).

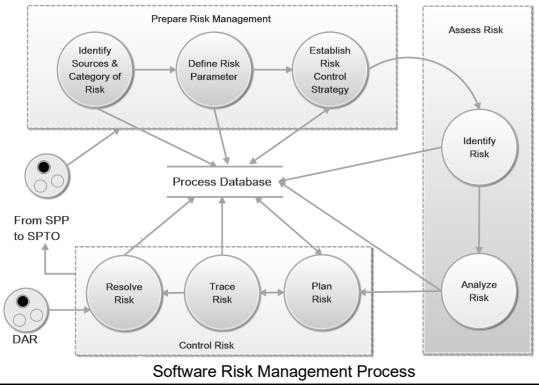
2.2 SQC Models and Techniques---RM

- Two key components of risk management
  - Risk assessment is a discovery process of identifying sources of software risk, analyzing or evaluating their potential effects and prioritizing them.
  - Risk control is a process of developing software risk resolution plans, monitoring risk status, implementing risk resolution plan, and correcting deviations from the plan.

2.2 SQC Models and Techniques---RM

- Uncertainty in risk assessments
  - Uncertainty is a key concept in risk conceptualisation and risk assessments
  - Probabilistic analysis is the predominant method used to handle the uncertainties involved in risk analy- sis, both aleatory (representing variation)（偶然的） and epistemic (due to lack of knowledge) （认识的）

2.2 SQC Models and Techniques---RM



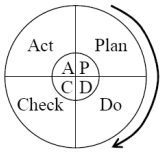
Software Risk Management Process

2.2 SQC Models and Techniques---PDCA

- PDCA cycle, also called Deming Cycle or Deming Wheel
- PDCA cycle is a continuous improvement process composed of four parts:
  - Plan, Do, Check, and Action.
- PDCA cycle is proposed by Dr. W. Edwards Deming in 1950 when he was invited to give a speech in Japan.

2.2 SQC Models and Techniques---PDCA

- PDCA is an important principle to improve product quality
- PDCA is a basic method to refine enterprise management and enterprise operation.
- It is also a basic foundation for the various iterative and spiral process models in IT project management.



2.2 SQC Models and Techniques---PDCA

- **Plan**
  - Plan means establishing the objectives and processes necessary to deliver results in accordance with set requirements. In this stage you need to plan the whole PDCA process.
- **Do**
  - Do means implementing the planned processes, taking small steps in controlled circumstances.
- **Check**
  - After implementing the planned processes, the results should be studied. Monitoring and evaluation of the processes and results against objectives and specific requirements are needed and a report of the results is necessary.
- **Act**
  - After the check step, actions based on what was studied in the previous step have to be taken to reach the necessary improvement

2.2 SQC Models and Techniques---PDCA

- Example
  - Medical Process Management by Applying PDCA to EMR
- EMR
  - Electronic Medical Records

2.2 SQC Models and Techniques---PDCA

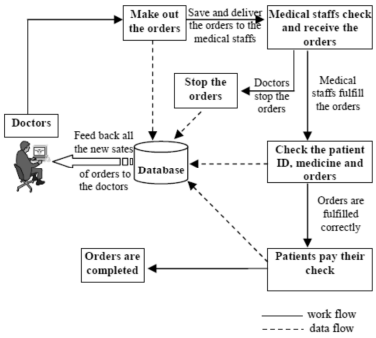
Relationships between Medical Process and PDCA

Steps	Activities	Medical processes
Plan	□. Identify and recognize the problem □. Analyze the factors which cause the problem □. Set down a plan for the whole process	Make out the orders based on patients' state of illnesses
Do	□. Implement the plan	Medical staffs fulfill the orders
Check	□. Check and analyze the results	System checks the state of fulfillment and returns the results to doctors
Act	□. Take action to standardize or improve the process □. Plan for future	Improve the medical processes and perfect the system's function

2.2 SQC Models and Techniques---PDCA

- *Design the ordering system with PDCA*
  - Analysis of the data flow of the ordering system with PDCA is as follows:
    - Data source: orders, states of the orders and queries of checking information;
    - End of the data flow: in order to form the closed cycle, states of the orders are back to the doctors and results of the checking queries back to the medical staffs;
    - Data processing: changing the states of orders, storing the orders and acquiring the orders;
    - States of the orders: make out the orders; deliver the orders to medical staffs; after checking all the information, fulfill the orders; patients pay their check, orders are completed; and doctors can stop an order when there is a suddenness or mistake.

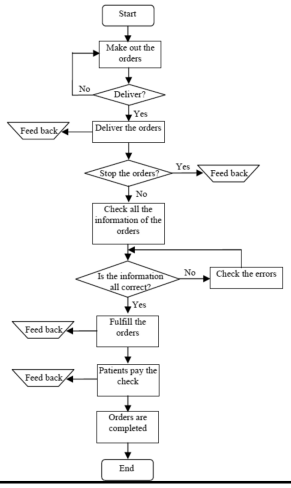
2.2 SQC Models and Techniques---PDCA



Data flow of the ordering system

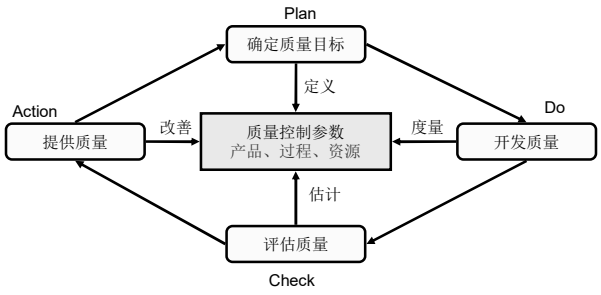
2.2 SQC Models and

- System Process of the ordering system



2.2 SQC Models and Techniques---TSQC

- Total Statistical Quality Control（全面统计质量控制） Model based on PDCA



2.2 SQC Models and Techniques--- TSQC

- 质量控制模型中的参数不是孤立的，而是具有相关性。
- 在质量控制中需要对这些参数进行综合调节、平衡。
- 参数
  - 产品：所有可交付物
  - 过程：所有活动的集合
  - 资源：活动的物质基础（人力、技术、设备、时间、资金等）

2.2 SQC Models and Techniques--- TSQC

- 信息系统参数举例——产品

类　型	举　例
文档、计划	软件开发计划、软件质量计划
规格说明	系统需求说明书、系统设计说明书
中间产品	软件设计文档
数　据	测试结果
软　件	最终系统

2.2 SQC Models and Techniques--- TSQC

- 信息系统参数举例——过程

类　型	举　例
管理过程	资源的使用、监控开发进展、任务分派
技术过程	系统设计评审、系统测试、软件编码

2.2 SQC Models and Techniques--- TSQC

• 信息系统参数举例——资源

类　型	举　例
人　力	管理人员、技术人员
设　备	软件开发设备、软件测试设备
时　间	开发进度表
资　金	投资资金

2.2 SQC Models and Techniques--- TSQC

• 举例——信息系统质量管理模型中的步骤和工具

阶段	详细步骤	可利用工具
计划阶段	分析现状，找出各种可能影响质量的问题或隐患	排列图、直方图、控制图
	分析问题的原因	因果图
	确定保障质量的关键因素	排列图、相关图
	针对关键因素，制定质量保障措施	/5W1H0方法
执行阶段	按照计划，实施质量保障	
检查阶段	检查计划实施结果	排列图、直方图、控制图
行动阶段	将成功的经验转化为相应的标准	修改相应规章制度、标准
	未解决、或新出现的问题转到下一个PDCA循环	

2.3 Software Quality Assurance

- What is Quality Assurance?
  - Quality assurance activities are work process oriented.
  - They measure the process, identify deficiencies, and suggest improvements.
  - The direct results of these activities are changes to the process.
  - These changes can range from better compliance with the process to entirely new processes.
  - The output of quality control activities is often the input to quality assurance activities.
  - Audits（审核） are an example of a QA activity which looks at whether and how the process is being followed. The end result may be suggested improvements or better compliance with the process.

2.3 Software Quality Assurance

- What is Quality Assurance?
  - A planned and systematic pattern of all actions necessary to provide adequate confidence that the item or project conforms to established technical requirements.

2.3 Software Quality Assurance

- Software Quality Assurance involves
  - reviewing and auditing the software products and activities to verify that they comply with the applicable procedures and standards.
  - providing the managers and software project team members with the results of these reviews and audits.

2.3 Software Quality Assurance

- Why are we concerned with software quality assurance?
  - Legal liability
  - Cost effectiveness
  - Customer requirements

2.3 Software Quality Assurance

- Software quality assurance (SQA)
  - Consists of a means of monitoring the softwar engineering processes and methods used to ensure quality.
  - It does this by beans of audits of the quality management system under which the software system is created.
  - These audits are backed by one or more standards, usually ISO 9000 or CMMI
- It is practically impossible to iron out every single bug before releasing it both from a difficulty point of view and due to time constraints.

2.3 Software Quality Assurance

- SQA Methodology
  - PPQA audits
    - process and product quality assurance
    - is the activity of ensuring that the process and work product conform to the agreed upon process.

2.3 Software Quality Assurance

- SQA activities
  - Verification验证 and validation确认
  - Test测试
  - Review审查
  - Audit审计
  - inspection检查

2.3 Software Quality Assurance

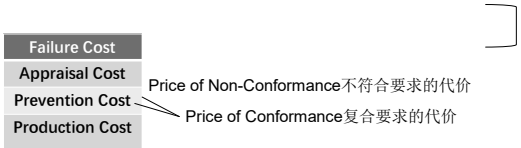
- The role of SQA
  - to find the better way, from a long-range viewpoint, over the course of all the software projects in the plant
  - to educate all those involved in developing the product in the implementation of the better way.

2.3 Software Quality Assurance

- Advantages of SQA
  - Improved customer satisfaction
  - Reduced cost of development
  - Reduced cost of maintenance

2.3 Software Quality Assurance

- Quality Cost
  - ***COQ = COF + (COA + COP)***
  - Prevention: costs incurred attempting to prevent or aviod errors 预防:为防止或避免错误而发生的成本
  - Appraisal: costs incurred attempting to detect errors
  - 评估:为检测错误而产生的成本
  - Failure: costs incurred because the other attempts were not successful
  - 失败:由于其他尝试失败而产生的成本



2.3 Software Quality Assurance

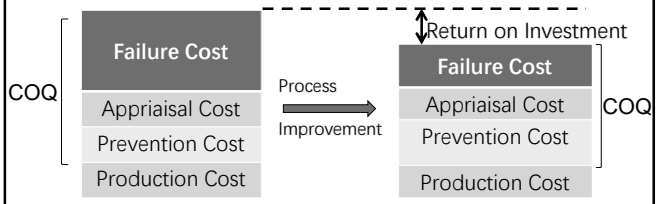
Prevention Costs (COP)
training
standard, procedures
planning
quality improvement
Audits审计
analysis

Failure Costs (COF)
correction and re-work
customer complaints
supply failure
equipment failure
manpower failure
legal fees律师费
lost benefits利益流失

Appraisal Costs (COA)
reviews
walkthroughs
testing
supplier monitoring供应商监控

2.3 Software Quality Assurance

Quality is free



Cost of implementing process improvement programs can be funded by the saving from reduction in COQ

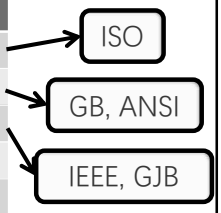
2.4 Software Quality Standards

- Why are software standards important?
  - encapsulate best (or most appropriate) practices
    - acquired after much trial and error
    - helps avoid previous mistakes
  - provide a framework around which to implement SQA process
    - ensures that best practices are properly followed
  - assist in ensuring continuity of project work
    - reduces learning effort when starting new work

Each project needs to decide which standards should be: ignored, used as is, modified, created

2.4 Software Quality Standards

Software Quality Standards Levels
International standards
National standards
Professional standards
Enterprise standards
Project specification



2.4 Software Quality Standards

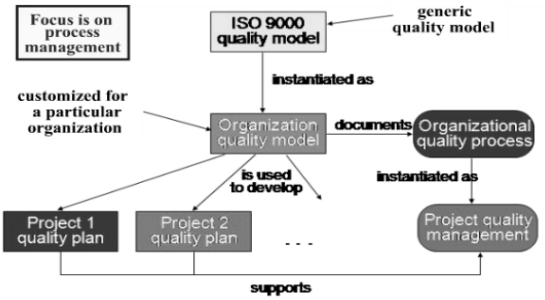
- Commonly used software quality standards
  - ISO 9001（适用于所有工程行业）/ 9000-3（ISO子集，在软件过程的使用中帮助解释该标准）
  - CMM
  - CMMI
  - IEEE Software engineering standars
  - ISO/IEC TR 15504

2.4 Software Quality Standards

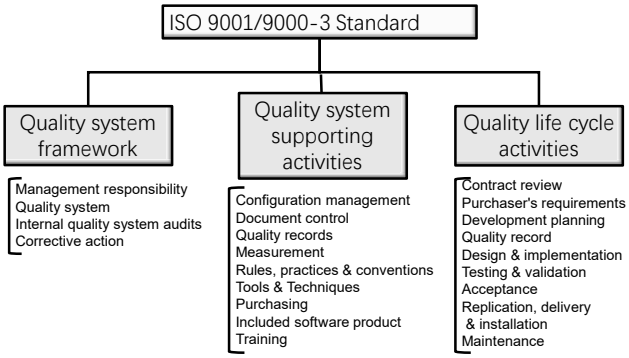
- ISO: International Organization for Standards 国际标准化组织
- CMM: Capability Maturity Model for Software 能力成熟度模型
- CMMI: Capability Maturity Model Integration 能力成熟度模型集成
- IEC: International Electro technical Commission 国际电工委员会
- TR: Technique Report 技术报告

2.4 Software Quality Standards

- ISO 9001/ 9000-3



2.4 Software Quality Standards



2.4 Software Quality Standards

- **CMM** was developed by the US Department of Defense at Software Engineering Institute.
- Objective of CMM: improve the existing software development processes
- Five maturity levels of CMM
  - Initial
  - Repeatable
  - Defined
  - Managed
  - Optimized

2.4 Software Quality Standards

- The main disadvantages of CMM model
  - When organizations use CMM, they look at each level as a target, they make their goal to reach the next level up, this can be a dangerous thought because if you become fixated on reaching the next level, you may forget the real goal, that is to improve the processes.
  - CMM does not specify a particular way to achieve these goals.
  - CMM considered helps full only if it is applied early in the software development process, that is, if there is a process that is in a crisis, it cannot be used as an emergency method for recovering from a difficult position.
  - CMM is concerned with the improvement of management related activities, not giving importance to the process related activities.

2.4 Software Quality Standards

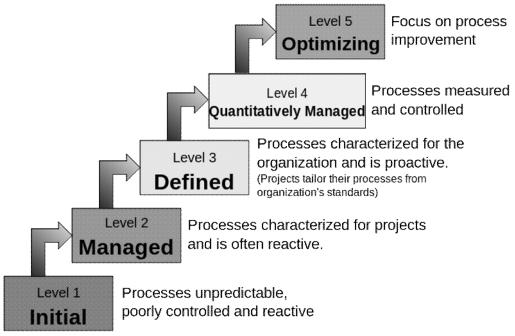
- **CMMI** is a process model that provides a clear definition of what an organization should do to promote behaviors that lead to improved performance and allow integrating the different organization functions
- CMMI is created by ombining the CMM models (SW-CMM V2.0, Integrated Product Development (IPD), and System Engineering CMM (SE-CMM)

2.4 Software Quality Standards

- CMMI consist of five maturity level are defined as
  - Initial
  - Repeatable
  - Defined
  - Qualitatively managed
  - Optimized

2.4 Software Quality Standards

Characteristics of the Maturity levels



2.4 Software Quality Standards

- The disadvantages of CMMI
  - may not be suitable for every organization.
  - it may add overhead in terms of documentation.
  - may require additional resources and knowledge required in smaller organizations to initiate CMMI-based process improvement.
  - may require a considerable amount of time and effort for implementation.
  - require a major shift in organizational culture and attitude.