中国通信服务
CHINA COMSERVICE

发起人高层会议总结报告

RAGHAV S. NANDYAL

SITARA TECHNOLOGIES PVT. LTD.

一、总体发现

以实现赞助目标，使用名为 SPRUM 的专有战略管理工具，使用 Measurements® Systemic Process Review Using Measurements - SPRUM®（Raghavan S. Nandyal 的注册商标）进行系统流程审查，以获得更深入的见解“如何以可靠的测量指标为重点，提高高成熟度流程的有效性并使其实用化”。 这次我们依托 CMMI V3.0 标准开展了 CMMI 5 级基准评估。通过文档核查、人员访谈到量化分析评审的全面诊断，整体评估结果非常客观，比较贴近我们当前的实际运行状态。评估过程中确实暴露出了一些短板和提升空间，这些问题并非局部偶发，而是贯穿在整体管理链条中的关键环节。为此，公司高层、EPG 过程改进小组以及项目、质量、研发等核心团队围绕评估结果进行了集中研讨。大家总体上达成了高度共识：本次评估结果客观、可信，与我们当前实际运行情况的匹配度高达 95%。

二、经验教训

我们按照 工程过程 (Engineering)、项目管理 (Project Management)、支持过程 (Support) 以及 过程管理 (Process Management) 四大核心领域进行深度的归纳与总结。

工程过程（Engineering Processes）

需求获取与政策整合： 需明确如何有效获取、理解和捕获受产品影响的利益相关者（特别是涉及乡村管理与行政领域）的最终用户输入。同时，应确保产品能够与中国的乡村管理政策深度整合。提升同行评审与代码走查有效性： 亟需提高代码走查的有效性，以解决编码标准的语义违反问题（如类和方法名称难懂、逻辑完整性检查不足）。此外，必须解决因函数返回值直接嵌入代码、缺少中间赋值而导致单元测试用例难以建立的痛点。鉴于目前有多种 AI 解决方案可用，应认真考虑引入 SonarQube/SonarCloud 和 Codacy 等行业黄金标准工具。 规范技术命名与降低维护开销：必须纠正代码中的拼写错误与不规范命名，以降低基线代码的维护开销。应避免类名与方法名极度相似（如类名 StatisticsWebAccountTask 中包含无返回值方法 statisticsWebAccount()）而导致代码调用时难以阅读和理解的情况，方法应基于其实现的实际功能清晰命名。 一致性逻辑校验与测试关联： 针对开发中出现的多个不一致逻辑校验实例（如在 getRate 方法中，对 first 和 second 参数的判空与判零逻辑不

一致）进行修正，以降低测试和系统性验证代码构造的复杂性。同时，应将需求评审缺陷密度、系统测试缺陷密度与代码走查缺陷、单元测试缺陷密度进行关联，并通过产品集成的类似考虑来改进集成测试。

项目管理过程（Project Management Processes）

科学估算与引入复杂度因子：尽管目前已使用功能点进行规模估算，但在建立项目缓冲（Buffer）时，未使用数据复杂度和问题复杂度因素。应改变当前随意添加百分比增量的做法，充分考虑产品需支持多个农村行政办公室、数据格式不标准以及需符合国家乡村治理要求的复杂背景。 优化生命周期时间分配结构： 必须重新合理规划和分配时间。当前规划中大约 40-45% 的时间分配给编码阶段，多达 35-45% 分配给前端生命周期阶段，而用于审查、代码走查和测试等关键筛选活动的时间仅占约 10%。应为单元测试、集成测试、系统测试以及中间代码走查等更重要的调试阶段分配和计划足够的时间。 过程绩效趋势的动态监控： 项目监控不能仅停留在“拥有”绩效模型这一必要条件上。应通过过程绩效模型（PPM）的变更来动态跟踪过程绩效趋势，切实理解和识别哪些过程监控方面真正帮助项目改善了最终成果。 风险控制与异常值科学分析： 严禁未经分析就直接剔除异常值。必须首先判断其贡献是源于普通原因还是特殊原因。若识别为过程控制

外部因素，应将其与风险管理相关联，并在将问题正确归因后再进行后续的有效解决。

支持过程（Support Processes）

聚焦底层可控因素的质量审计： 质量保证审计必须利用流程绩效模型之间的相互依赖关系，将焦点聚集于最低层的可控因素上(例如聚焦于那些影响需求评审缺陷密度的因素，如需求文档详细程度)，并及时剔除不显著的因素，提升审计效益。 量化数据与模型的配置管理： 配置管理实践应进一步涵盖量化数据管理领域。必须妥善保留和维护数据以及过程模型的先前配置版本，以便组织能够清晰了解绩效随时间演变的历程。 因果分析的前提保障：在因果分析和问题解决前，严禁随意剥离异常值。必须对异常值进行深入分析，判断其是由普通原因还是特殊原因所致。只有在将问题正确归因于过程内部或（过程控制外部的）风险管理后，才能开展行之有效的因果分析。

过程管理过程（Process Management Processes）

建立量化模型的级联与关联： 探索并建立流程绩效模型（PPM）之间的相互关联与依赖关系。例如：通过“客户满意度 =1.0764 - 1.310 × 遗留缺陷密度”向底层级联，明确遗留缺陷密度对需求评审缺陷密度、系统测试缺陷密度的依赖，以及后者对需求文档详细程度和需求评审覆盖率的层层依赖，形成有机的量化管理链条。 全生命周期缺陷归因与阶段改进： 必须扭转缺陷密度度

量只关注需求管理和设计等过程内部阶段的现状。应认识到缺陷也可能由于错误构建、错误发布和现场测试报告而注入。必须将系统测试缺陷准确归因到各生命周期阶段，以确保阶段性改进措施能够有效实施。完善组织级治理与生产率建模： 纠正将整体生产率完全等同于或过度依赖于编码生产率的片面治理实践。代码变更往往是由于测试暴露了代码的不足之处，如果在治理和建模中不考虑导致代码变更的测试工作量，将会产生错误的管理导向，引发不正确的治理实践。支持模型递归使用的基础设施建设：组织实施基础设施以及过程资产库（PAL）中的相关过程基础设施，必须进行升级与优化。必须提供强有力的底层支持，以满足未来对定量模型进行递归使用的需求，打破当前各量化模型之间相互脱节、孤立的局面。 全员量化思维的培训与人员发展： 组织级培训与人员发展必须覆盖那些负责提供原始度量数据的一线人员。应通过针对性培训，确保他们能够以“量化思维”而非单纯的“合规思维”来参与项目和过程，从源头保证量化管理数据的真实性与有效性。

三、现实意义

上述所有发现项对企业而言，其核心意义在于推动企业完成从“人治经验型”向“数据驱动型”的根本性治理跃迁——通过将分散在工程、项目管理、支持与过程管理四大领域的零散痛点（如编码随意、估算粗放、模型脱节、考核片面）系统性地转化为一环扣一环的量化管理链条（从需求阶段即以复杂度因子科学估算缓冲，到强制门禁与时间再分配将缺陷拦截在早期，再到建立缺陷密度与客户满意度间的级联回归模型，最终修正以偏概全的治理指标），企业不仅能够直接削减 20%-35%的返工与维护成本、将项目预算偏差率收窄至 ±10%以内，更能在数字乡村等战略赛道上以“合规即入场券”的姿态构筑差异化竞争优势；同时，这套体系让管理层拥有了提前预判风险的“仪表盘”，让工程师的绩效从“堆砌代码”转向“交付高质量功能点”，从而彻底告别“临上线前通宵堵枪眼”的被动局面，最终将过往随人员流失而蒸发的隐形资产沉淀为组织级的过程资产库，以 20%的过程监控精力锁定 80%的交付确定性，为企业的长期规模化发展与品牌公信力筑就坚实底座。

四、改进措施

EPG 应首先通过组织级培训覆盖那些负责提供原始度量数据的一线人员，引导他们从应付检查的“合规思维”向驱动改进的“量化思维”转变，从源头保证数据的真实性；同时，必须升级配置管理实践，将量化数据与过程模型的先前配置版本纳入版本控制，以确保组织绩效随时间演变的历程可追溯。在此基础上，优化实

施基础设施及过程资产库（PAL），使其能够提供强有力的底层技术支撑，以满足未来对定量模型进行递归使用的需求，彻底打破当前各量化模型之间相互脱节、孤立的局面。针对研发一线的痛点，EPG 应正式引入 SonarQube/SonarCloud 或 Codacy 等行业黄金标准工具来提高代码走查的有效性。通过建立硬性编码规范，重点根治编码标准的语义违反、类名与方法名极度相似导致的易读性差、以及因返回值直接嵌入调用缺少中间赋值而导致单测用例难以建立的实际问题，并强制修正类似 getRate 方法中对参数判空与判零逻辑不一致的漏洞，以降低测试和系统性验证的复杂度。此外，必须在组织级标准过程中重构项目计划模板，纠正当前编码和前端生命周期占用 80%-90% 时间、而评审与测试仅占 10% 的不合理资源错配，为单元测试、集成测试、系统测试以及中间代码走查等更重要的调试阶段计划并分配足够的时间，实现质量前移。同时，建立全生命周期的缺陷追溯机制，扭转过去缺陷密度度量只关注需求和设计等内部阶段的片面现状，将系统测试缺陷、错误构建、错误发布和现场测试报告准确归因到各自注入的生命周期阶段，确保阶段性改进措施能够精准实施。在此阶段，EPG 应指导项目深入探索并建立流程绩效模型（PPM）之间的级联与多级依赖关系（如将客户满意度层层关联至遗留缺陷密度、需求评审缺陷密度、系统测试缺陷密度，进而追溯至需求文档详细程度和评审覆盖率），形成有机的量化管理链条。在管理机制

上，要在基于功能点的规模估算标准中正式引入数据复杂度和问题复杂度因子（特别是针对对接多农村行政办公室、数据格式不标准及乡村治理政策整合的项目），严禁随意添加百分比增量；项目监控必须通过跟踪 PPM 模型的变更来分析过程绩效趋势，切实识别哪些监控方面真正帮助项目改善了最终成果，而非仅仅流于拥有模型的形式。出台硬性统计规范，严禁未经分析就直接剔除数据中的异常值，必须首先通过分析判断其源于普通原因还是特殊原因，属于过程控制外部因素的与风险管理相关联，属于过程内部的则进行深入的因果分析与彻底解决。最后，PQA 质量保证审计必须利用模型的相互依赖关系，将焦点聚集于最低层的可控因素上并剔除不显著因素，同时考核整体生产率时必须综合考虑导致代码变更的测试工作量，严禁将其完全等同于编码生产率，从而在组织高层建立起健康的治理导向，防止引发错误的治理实践。

我在此授权并同意您本人和 SITARA Technologies 在 SITARA 的出版渠道上分享我们的评估成果，在 SITARA Technologies 认为合适的情况下宣传我们的评估成果。

中通服创立信息科技有限责任公司

发起人：钱小峰 xiaofeng qian

2026 年 6 月 27 日

EXECUTIVE SESSION BRIEFING: SPONSOR FEEDBACK

Overall findings

To achieve sponsorship objectives, we employed the proprietary strategic management tool SPRUM — Systemic Process Review Using Measurements® (a registered trademark of Raghavan S. Nandyal) — to conduct a systematic process review, gaining deeper insights into "how to enhance the effectiveness and pragmatism of high-maturity processes with a focus on reliable measurement indicators."

This time, we conducted a CMMI Level 5 benchmark appraisal based on the CMMI V3.0 standard. Through a comprehensive diagnosis covering document reviews, personnel interviews, and quantitative analysis assessments, the overall appraisal results were highly objective and closely reflected our current operational status. During the appraisal, certain weaknesses and areas for improvement were indeed exposed — these issues are not isolated or incidental, but rather critical links throughout the entire management chain. To address them, our senior management, the EPG (Engineering Process Group), and core teams from

project management, quality, and R&D held intensive discussions around the appraisal results. Overall, we reached a strong consensus: the appraisal results are objective and credible, with a 95% match to our actual current operations.

Lessons Learned

We conducted an in-depth synthesis and summary across four core areas: Engineering Processes, Project Management Processes, Support Processes, and Process Management Processes.

Engineering Processes

Requirements Elicitation and Policy Integration: Clarify how to effectively elicit, understand, and capture end-user input from stakeholders affected by the product (especially those involved in rural management and administrative domains). At the same time, ensure that the product is deeply integrated with China' s rural management policies.

Improve the Effectiveness of Peer Reviews and Code Walkthroughs: Urgently enhance the effectiveness of code walkthroughs to address semantic violations of coding

standards (e.g., obscure class and method names, insufficient logic completeness checks). Furthermore, resolve the pain point where unit test cases are difficult to set up because function return values are embedded directly in code without intermediate assignments. Given that various AI-based solutions are now available, serious consideration should be given to introducing industry gold-standard tools such as SonarQube/SonarCloud and Codacy.

Standardise Technical Naming and Reduce Maintenance Overhead: Correct typos and non-standard naming in the code to reduce maintenance overhead in the baseline code. Avoid situations where class names and method names are extremely similar (e.g., class StatisticsWebAccountTask contains a void method statisticsWebAccount()) that makes the code difficult to read and understand during invocation. Methods should be clearly named based on the actual functionality they implement.

Consistency Logic Validation and Test Association: Correct multiple instances of inconsistent logic validation

observed during development (e.g., in the getRate method, inconsistent null-check and zero-check logic for the first and second parameters) to reduce the complexity of test construction and systematic verification. Additionally, correlate requirements review defect density, system test defect density with code walkthrough defect density and unit test defect density, and improve integration testing through similar considerations applied to product integration.

Project Management Processes

Scientific Estimation and Introduction of Complexity Factors: Although function points are currently used for size estimation, data complexity and problem complexity factors are not used when establishing project buffers. The current practice of arbitrarily adding percentage increments should be changed, fully considering the complex context where the product must support multiple rural administrative offices, non-standard data formats, and compliance with national rural governance requirements. Optimise Lifecycle Time Allocation Structure: Reasonably re-plan and re-allocate time. Currently, approximately 40-

45% of the time is allocated to the coding phase, and as much as 35-45% to front-end lifecycle phases, while only about 10% is left for critical screening activities such as reviews, code walkthroughs, and testing. Sufficient time should be allocated and planned for more important debugging phases, including unit testing, integration testing, system testing, and intermediate code walkthroughs. Dynamic Monitoring of Process Performance Trends: Project monitoring should not stop at merely "having" a performance model as a necessary condition. Instead, dynamically track process performance trends through changes in the Process Performance Models (PPMs), so as to genuinely understand and identify which aspects of process monitoring truly help projects improve final outcomes. Risk Control and Scientific Analysis of Outliers: It is strictly forbidden to eliminate outliers directly without analysis. It is essential to first determine whether their contribution arises from common causes or special causes. If identified as external factors outside process control, they should be linked to risk management and resolved effectively only after correctly attributing the issue.

SUPPORT PROCESSES The core defects in the support field are mainly exposed in the disconnection of responsibilities between code review and quality assurance, poor version control of quantitative assets, and the lack of transparency in decision analysis. In the calls before code assignment, the phenomenon of calling the method FileUtil.Savelamge(...) frequently occurs (for example, before siteCheckRecord.setInfraredImgUrls(ailmgUrls);). It is currently unclear whether Savelamge is a deliberate typo or should be SaveImage. Although this incorrect naming choice may not directly destroy product functions, it is crucial to ensure that the code can be maintained by personnel other than the author and to achieve code reusability. Such code walkthroughs and specification considerations failed to effectively intercept, indicating that the relevant responsibilities were not effectively implemented in the Product Quality Assurance (PQA) and peer review links. Although the organization has established Process Performance Models (PPM) at the organizational level, there is currently no sign or record to show when these models were established, and their correlation and effectiveness with earlier versions of models. Due to the inability to clarify what models existed in the past, how these models evolved and improved over time, and the next development plan for

these models in the future, these key quantitative assets failed to be included in good version control and configuration management. It is suggested that when determining evaluation criteria and setting decision criteria weights, the project should add rules or logical descriptions for the weight setting of some decision criteria, otherwise it will not be conducive to improving and guaranteeing the objectivity of the final decision. Due to a large number of bad coding methods in the code that directly call function return values without assigning them to temporary variables and negate the return values in if logic, testing personnel can hardly perform isolated and effective tests on these complex logical structures during quality verification and validation, severely hindering the implementation of verification work.

Process Management Processes

Quality Audits Focused on Bottom-Level Controllable Factors: Quality assurance audits must leverage the interdependencies among process performance models, focusing on the lowest-level controllable factors (e.g., focusing on factors that affect requirements review defect density, such as the level of detail in requirements documents), and promptly remove non-significant factors to

enhance audit effectiveness.

Configuration Management of Quantitative Data and Models: Configuration management practices should be further extended to cover the domain of quantitative data management. Previous configuration versions of data and process models must be properly retained and maintained, so that the organisation can clearly understand the evolution of performance over time. Prerequisites for Causal Analysis: Before conducting causal analysis and problem solving, it is strictly forbidden to arbitrarily remove outliers. Outliers must be thoroughly analysed to determine whether they are due to common or special causes. Only after correctly attributing the issue to either internal process factors or (external to process control) risk management can effective causal analysis be carried out.

RELEVANCE

Establish Cascading and Interrelationships of Quantitative Models: Explore and establish the interconnections and dependencies among Process Performance Models (PPMs). For example, cascade from "Customer Satisfaction = 1.0764 − 1.310 × Defect Density Escaped" downward, clarifying the

dependency of escaped defect density on requirements review defect density and system test defect density, and the further layer-by-layer dependency of the latter on requirements document detail and requirements review coverage, thereby forming an organic quantitative management chain.

Defect Attribution Across the Full Lifecycle and Phase-Based Improvement: The current practice of measuring defect density only for internal phases such as requirements management and design must be reversed. It should be recognised that defects can also be injected due to faulty builds, faulty releases, and field test reports. System test defects must be accurately attributed to each lifecycle phase to ensure that phase-specific improvement actions can be effectively implemented. Improve Organisational-Level Governance and Productivity Modelling: Correct the one-sided governance practice that equates overall productivity entirely or excessively with coding productivity. Code changes are often caused by tests exposing deficiencies in the code. If the testing effort that drives code changes is

not considered in governance and modelling, it will create misleading management directions and lead to incorrect governance practices. Infrastructure Support for Recursive Use of Models: The organisational implementation infrastructure and the related process infrastructure in the Process Asset Library (PAL) must be upgraded and optimised. Strong underlying support must be provided to meet future needs for recursive use of quantitative models, breaking the current situation where quantitative models are disconnected and isolated from one another. Training and People Development for Quantitative Thinking Across the Organisation: Organisational-level training and people development must cover the frontline personnel who provide raw measurement data. Targeted training should ensure that they participate in projects and processes with a "quantitative mindset" rather than a mere "compliance mindset," thereby guaranteeing the authenticity and effectiveness of quantitative management data from the source.

IMPROVEMENT ACTIONS

Armed

I hereby authorize and agree that you and SITARA Technologies may share our appraisal results on SITARA's publishing channels and promote them as deemed appropriate by SITARA Technologies.

China Comservice Enrising Information Technology Co.,

Lt

Sponsor: 钱小峰 xiaofeng qian

June 27th 2026