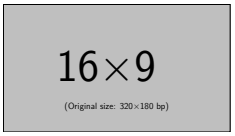

Controls Organization

Bootstrap, Quality

Prototype Document
A convergence of twisty dependencies. 0b-verge-proto-doc

First Last

Prior: 664e

RACI
Responsible: First Last
Authority: Management, Senior
Contact: Leadership
Inform: Staff

.1 Preface

A framework for the design of operational controls, incorporating authorization, and iterative optimization, as a structured development process, is not easily phrased, and the implementation is equally complex. Yet, simultaneous framework and operational development, with authorized implementation, maximum value, least time, and lowest cost is fundamental underpinning of nearly every project; regardless of technicality. Many years ago, I learned two valuable lessons from the pharma industry: "If it isn't documented, it didn't happen," and what I will characterize as, "Repeatability is the mother of improvement."

What begins as a system process development template, an initiative I've revisited many times, quickly collides with a dilemma: should I abandon the generic framework with abstract inputs and outputs—applicable to any project—in favor of real-world integration examples? The former is inherently inaccessible for education, lacking the elucidation, and verification an example provides, and which is also a great asset for development; while the latter introduces an application barrier, the example data must be filtered out, with every new template application. So, why not both; a process framework, with examples!

The challenge is that framework development and example-populated framework development are separate, yet interdependent, projects. Their integration essentially forms a third project. As details are refined and applied, the time to iterate the development cycles is compounded. Although using examples is the best way to uncover optimization opportunities, populating frameworks with examples becomes a distraction from refining the framework itself.

The simple yet crucial answer is that you must provide examples *with* templates—especially *process templates*—because without input and output examples, process understanding is elusive. Even generative AI struggles with example free abstract process; prompt for a novel application, of multiple preexisting activities or objects, and that results in difficulty—given a solution, generative AI cannot even understand the problem, if it is truly a novel one. If the idea is truly new, the AI system doesn't identify the solution (without first bootstrapping a simulation), even when the prompts specifically converge on the required components to solve the problem, the step of *inventing their application* and *assembling the solution* is illusive, if it hasn't been done before.

Take a moment to prompt AI about the pros-and-cons of coconut oil as a lamp fuel, and heat source. A solution using this fuel has critical barriers, yet comes with significant benefits, all about which generative AI will pontificate, if the solution is provided. Yet, discovery of this optimization is missed by generative AI, presumably because prior support is not represented in the training. (At least at the time of this writing.) Google doesn't fare much better, try getting beyond a paraffin, canola, and olive oil comparisons for lamp fuel; and, the health aspect of using the various oils on your skin. These are popular topics and comprise the first 50,000 or so search results, primarily ad revenue inspired blogs about patio tiki torches and the romance inspiring domestic oil lamp. The critical challenge of using coconut oil as a lamp fuel is the burn temperature. It it burns so hot that cotton wicks are quickly vaporized, extinguishing the flame as it descends into the fuel. Glass fiber wicks are no more viable. The heat causes the glass to melt, and form mushroom like structures on the tips, which prevent fuel from wicking up, in a few hours, the flame is out, and the wick is ruined. Half of the solution is to use carbon felt, a material with the consistency of regular felt, and a burn temperature suitable to provide protection from acetylene welding. The relatively nominal challenge of crafting a coconut oil lamp from carbon felt can be a life saver in a winter emergency power-loss situation. It will burn for a long time, and the flame is hot enough to make marketable difference indoors, when it's freezing, and a primary heating system is non-functional. There is another hurdle

though, the energy density of coconut oil is so great that it requires a flue, like the tall smokestack of a blast furnace creates convection currents feeding oxygen to the fire, or the way the glass bulb of a hurricane lamp draws oxygen up from the bottom to the flame. Ironically, hurricane glass bulbs are sold as decorative candle accessories, and possibly as flame safety, or breeze protection; yet they don't come with holes in the bottom to foster a healthy flame and reduce the soot produced by oxygen starved candle flame. I digress. While generative AI may pontificate the pros-and-cons, recapitulate data sheets, detail physical properties, support the accuracy and viability of a carbon felt coconut oil lamp; go ahead, try and prompt for it. Describe the goal and barriers; but without actually describing and characterizing the solution components and details of their assembly; as if you didn't know those details. Identifying the components and their assembly is illusive to generative AI. Even prompting specifically for a solution that enables coconut oil as a fuel in a lamp, doesn't get any useful results. The generative AI tool doesn't understand why the flame goes out. I'll even give a reward of some sort to the first person who can show me prompts that lead a commodity AI platform to identify how the components of a coconut oil lamp solve the problem, without actually describing the bulb or wick. Bonus points if you can identify coconut oil as the alternative fuel, without actually offering it as an alternative. As for the health aspects, I've given up searching for answers, but it stands to reason that coconut oil is at least better than any other option.

Generative AI demonstrates *what we call understanding,* but describing understanding is as illusive as defining intelligence. That's the foundation of a theme that will be reoccurring in this book. Only after a novel solution is provided (or ironically, after we are presented with solutions that are new to us), then explanation becomes much more straightforward. While generative AI may expand on every aspect of, how, or why a solution works, in great detail, it lacks the originality, the capacity of authenticity required to truly invent something. Much like the juvenile "know-it-all" personality exhibits an amazing degree of knowledge, yet with little understanding of application, an inability to innovate, and probably acute symptoms of the Dunning-Kruger effect.

This book is not about generative AI as much as it is about the importants of observations, and controls, for the improvement of process and outcome.

The difficulty of discussing process generically, without specific application examples, led to the creation of a place marker for a guidance document called "Governance, Risk, and Compliance Sourcebook," and eventually the subtitle that better describes this work, *"A Controlled Operating Documents Approach."* The document was intended to catalog various tools of operational control, for later development, as needed, so focus could remain on developing the operation at hand which inspired the most recent framework initiative effort.

So, where are we? Is the next step a documentation of an operations framework, or the application of the framework to document implementation of my pet project? As it happens, it turns out both would become my new project. Fortunately, for anyone reading this far, that means the guidance won't simply be a GRC template sourcebook, but an actual book outlining the sequential roll out, of foundational documents, their authorization, and development, at a fictitious organization, called the *The Controls Organization.* An initial document, or three, will define documents themselves, their cataloging and authorization; subsequent details will develop other aspects of process, quality, compliance, controls, and improvement. And, of course no framework would be complete without an example demonstration application!

The pet project is the implementation of PKGSRC under a hybrid of Quality frameworks, including NIST RMF, Multi-Vee, Iceburg, DevOps, GxP, and DevSecOps. In practice, there are significant challenges maintaining *Platform, OS,* and *Software qualification controls*, through *revision* and *release cycles*. I will attempt to develop the guidance to address, and resolve, those challenges, in the most

generic, and optimal way possible!

If you are interested in that sort of thing, and especially as it applies to development of controlled compute platforms for scientific applications, read on! And, if you know a thing or two, please reach out, this is a work in progress, and I would love to expand the list of co-authors and collaborators!

Very Respectfully,
George Georgalis
Principal Author
george@galis.org

Part I

Operating Document Prototype

I.1 Motive

I.2 Scope

I.3 Information

I.3.1 Glossary

I.4 Control

I.4.1 Prepare

I.4.2 Categorize

I.4.3 Select

I.4.4 Implement

I.4.5 Assess

I.4.6 Authorize

I.4.7 Monitor

I.4.8 Customer

I.4.9 Partner

I.4.10 Vendor

I.5 Appendix

I.6 References

I.7 History

Part II

Protocols

II.1 0d-precept-jobname

0a-prepare: Prepare
0b-verge: Converge
0c-object: Objective
0d-precept: Precept
0e-perf: Perform
0f-princ: Principle
1a-cat: Categorize
1b-plan: Plan
1c-stand: Standard
1d-reg: Regulation
1e-sec: Security
1f-pol: Policy
1g-res: Research
2a-sel: Select
2b-des: Design
2c-req: Requirement
2d-prima: Primitive
2e-proto: Prototype
2f-spec: Specification
2g-user: guidance
3a-imple: Implementation
3b-exec: Execute
3c-proc: Proceed
3d-inst: Instruct
4a-assess: Assess
4b-check: Test
5a-auth: Authorize
6a-mon: Monitor
7a-client: Customer
8a-peer: Partner
9a-vend: Vendor

II.2 Operating Documents

II.2.1 Global Contents

0b-verge-proto-doc

titlePrototype Document
subtitleA convergence of twisty dependencies.
sectionPreface

1g-guide-grc-sourcebook

titleGovernence, Risk, and Compliance Sourcebook
subtitleA controled operating documents approach
sectionScope
sectionInformation
subsectionPurpose
subsectionGlossary
subsectionprep
subsectionPrepare Operation XXXXXXXXXX Sections
subsubsectionCAPA
subsectionprototype
subsectionPlan
subsectionPlan Categorize
subsubsectionRequirement
subsubsectionUser Guidance
subsectionImplementation Primatives

3b-exec-pkgsrc-admin

titlePKGSRC for Dynamic Requirements
subtitlePatching, deploying, and certifying binaries, for multiple releases, and operational cycles
sectionIntroduction
sectionPlan
sectionRequirement
sectionUser Guidance
sectionImplementation Primitives
sectionGlossary