Cooks, Recipes and Ingredients

Andy J. Nolan and Andrew C. Pickard
Rolls-Royce plc
Andy.Nolan@rolls-royce.com, Andrew.C.Pickard@rolls-royce.com

Abstract: To make a meal, you need ingredients and a recipe. A recipe defines sequencing, quantities, timing etc. This is analogous to a project's processes (ingredients) and life-cycle (recipe).

For a project, the attributes of cost, schedule and quality are properties that emerge from the recipes and ingredients. But how important is the recipe? This paper has found instances where a project's recipe had a 16-fold cost difference using the exact same ingredients. This suggests that a good cook can make a great meal almost regardless of the ingredients.

Many Project Managers inadvertently become chiefs of their projects and create new recipes in their attempt to recover their project. However, few managers are aware of the outcome of the recipes they create. When things turn out unexpectedly, generally badly, they blame the ingredients and not the recipe.

This paper shows how a business can characterize a recipe to meet business goals, define it in a structured way (a reference model) and then use that definition to plan and monitor a project. The method has been used at Rolls-Royce since 2002 and has been shown to improve project success, halving the level of scrap and rework whilst holding schedule. In one case, this method brought a 45% cost reduction to a project with only a small increase to schedule.

The concepts behind this paper are not new, but the notation used made it easy to define the ideal recipe, plan projects and to track them against the ideal recipe. This paper describes the background, benefits and methods to develop your own project recipes.

This paper is aimed at project managers, process architects and systems engineers. It defines a unified language to bridge all three needs in a language that can be understood by all.

The Important of a good recipe

Let us take the example of the systems and software V model shown in figure 1 (ref: 1). In the context of this paper, the processes are "ingredients" and how those processes are weaved together is defined as a "recipe". Assuming any project was to follow this recipe as shown, what would the outcome be in terms of product maturity, project cost and schedule? Quality, cost and schedule are attributes that emerge from the recipe and of the ingredients. Change the recipe and you get a different result. For example:

- How would the outcome differ if we did reviews early or late?
- How would the outcome differ if we did early integration and test?
- How would the outcome differ if we delayed requirements capture to the end of the project?

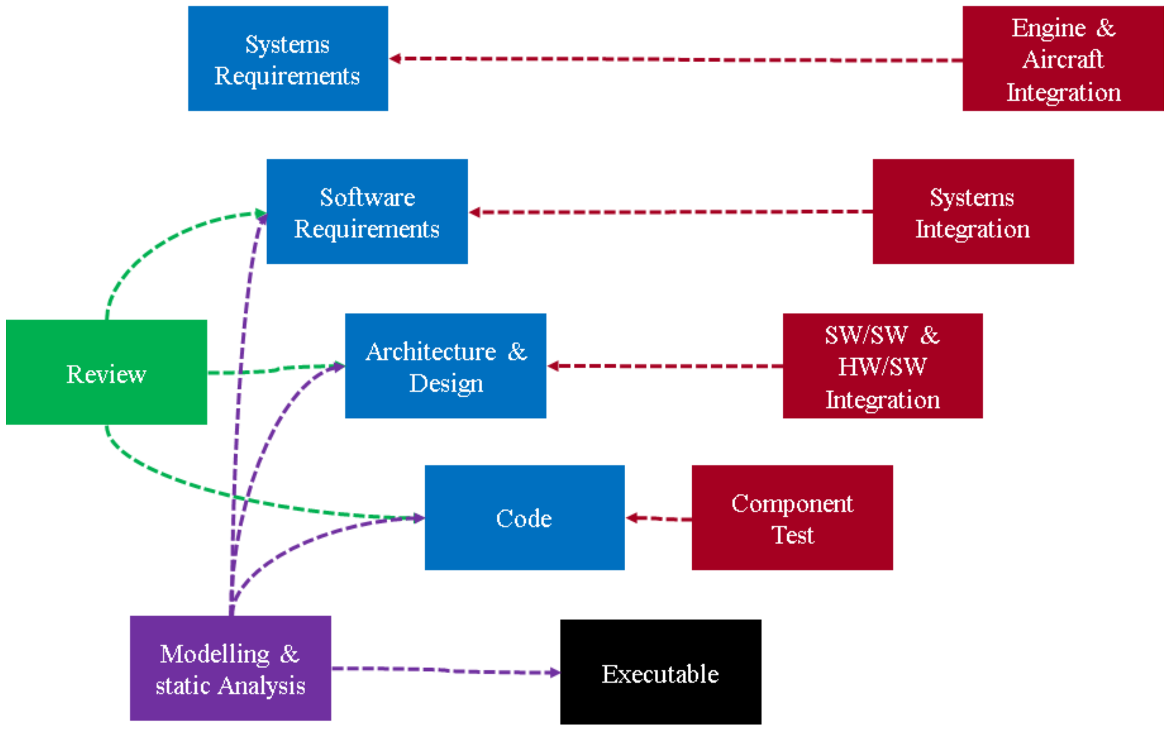
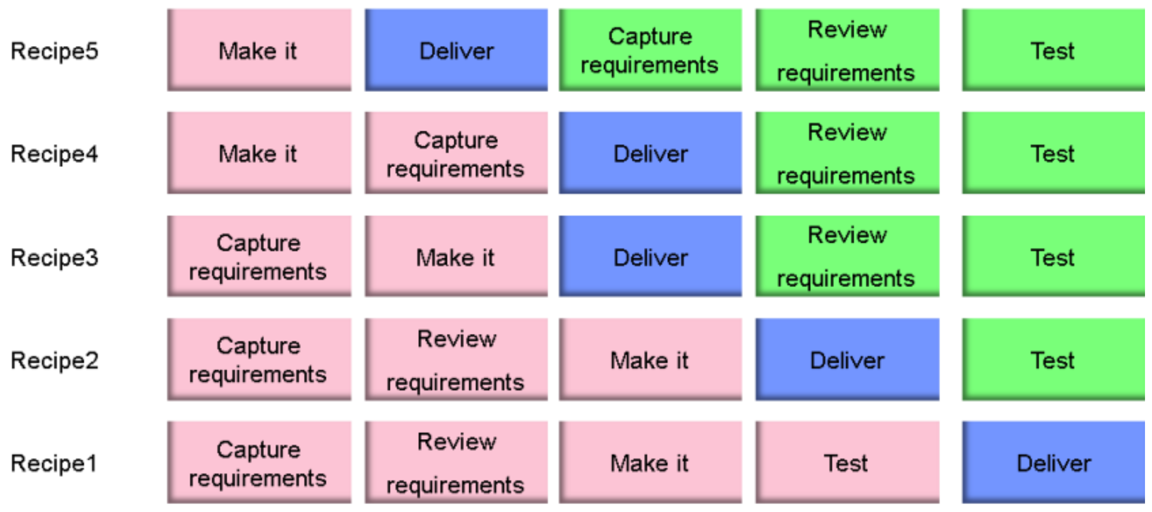


Figure 1: An overview of the software development process

Figure 2 illustrates the importance of recipe. (ref: 2, 3). Assume that there are only 5 processes involved; capture the requirements, review the requirements, make it, test it and then deliver it. In this example we shall assume the same team in each case.

Each row in figure 2 shows a different recipe. For each of these recipes, consider how the outcome will differ. Despite using the exact same processes (ingredients), the result of sequencing the processes differently will have a dramatic effect on



Recipe5	Make it	Deliver	Capture requirements	Review requirements	Test
Recipe4	Make it	Capture requirements	Deliver	Review requirements	Test
Recipe3	Capture requirements	Make it	Deliver	Review requirements	Test
Recipe2	Capture requirements	Review requirements	Make it	Deliver	Test
Recipe1	Capture requirements	Review requirements	Make it	Test	Deliver

Figure 2: Different recipes based on the same ingredients (processes)

outcome – cost, quality and schedule. In the 5 recipes, the costs will approximately double with each recipe as you move up the diagram.

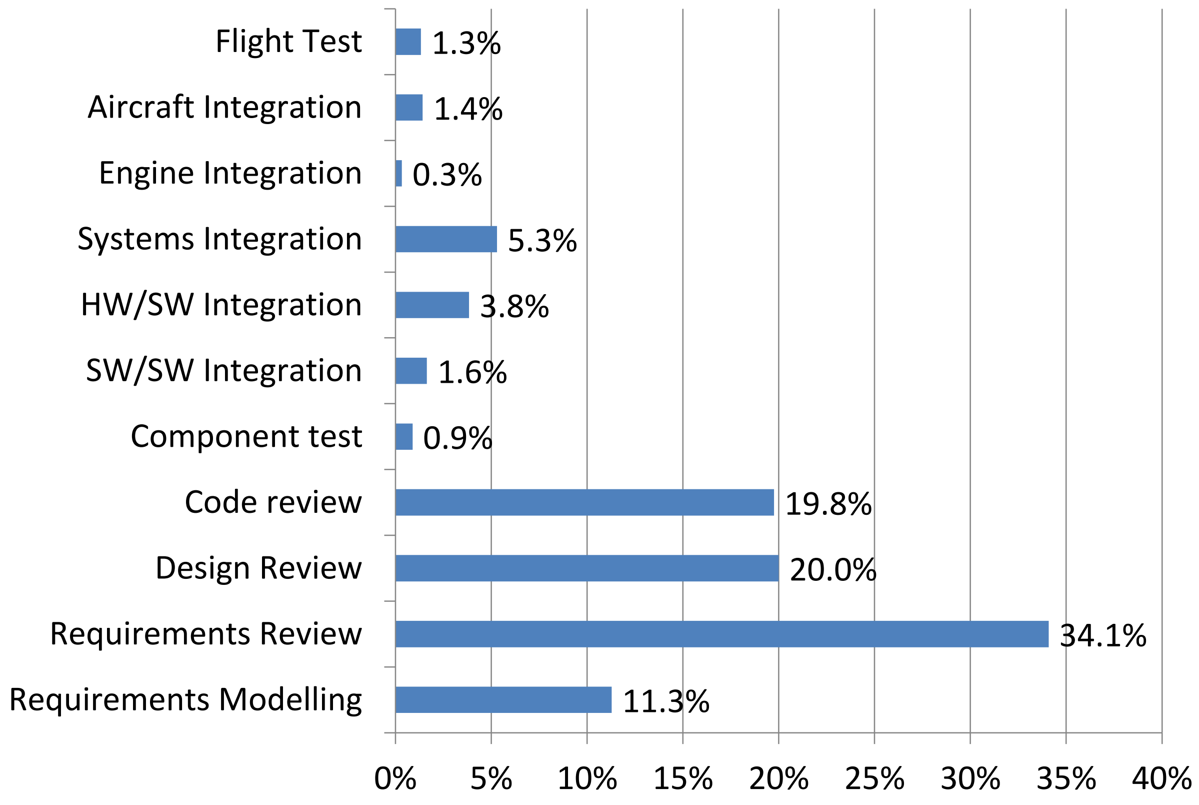


Figure 3: Where defects are detected

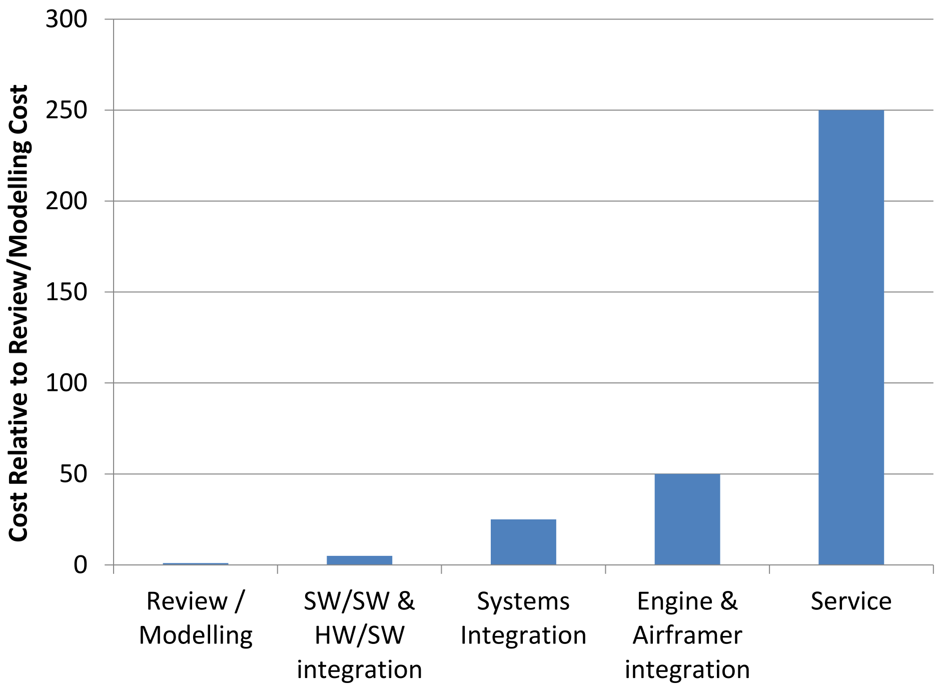


Figure 4: The cost to find and correct a defect

Why is this? Figure 3 shows where the defects are detected and figure 4 shows the cost to detect and correct a defect (ref: 2). If you move an "effective" process from the correct place in the recipe to the late (wrong) place where it adds little or late value, then there will be late defects detected and a corresponding escalation in cost to correct those defects.

A successful meal is determined by both the quality of the ingredients and the recipe used. Too much or too little ingredients, too soon or too late and the meal will be different. Success is not only determined by what you do and when you do it!

But this is well understood – or is it. Figure 2 would suggest that this is not always the case. These are real examples taken from industry.

Too often, a project manager becomes a process architect – a chief in this analogy. In desperation to recover a project, they will develop a recipe that meets a short term goal but may cost more in the long run. Also, very few projects could take their project plan and explain what property (goal) it was intended to achieve or even if it can meet a business or project goal.

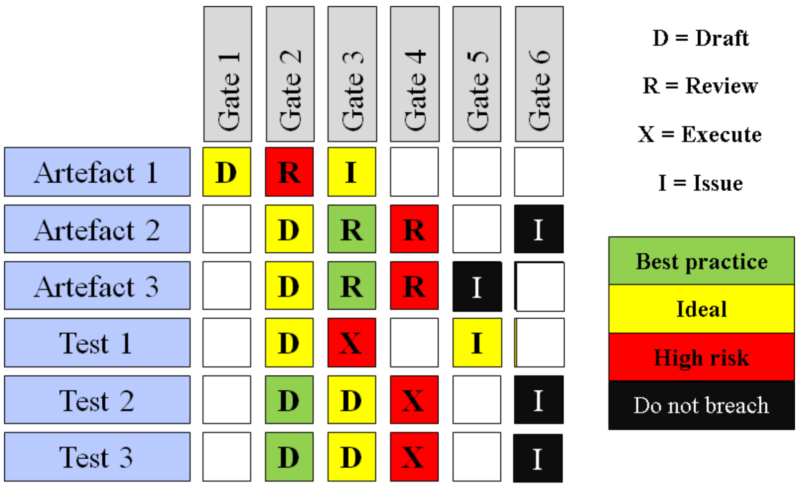
After a project has experienced difficulties, it is tempting to dig downwards to find the reason, or to find who to blame. It is tempting to launch yet another initiative to improve the ingredients. This paper proposes that you should first focus on the recipe.

We needed a way to define the ideal recipe and to define it in a way that we could guide project managers as well as quantify the effect of non-compliance to this ideal. In our attempt to define recipes for the business, in the form of reference models, we used the concept of a Gated Process.

The Gated Process

The Gated Process was developed at Rolls-Royce in 2002 to define and lock down the ideal recipe in the form of a reference model (ref: 2). The model can then be used to both plan and track a project. In tracking the project for compliance to the reference model it is possible to measure achievement toward the project and business goals.

The Gated Process is a method for integrating and managing processes to ensure repeated project success. A Gated Process integrates together a series of processes by defining what has to happen and



	Gate 1	Gate 2	Gate 3	Gate 4	Gate 5	Gate 6
Artefact 1	🟨 D	🟥 R	🟨 I			
Artefact 2		🟨 D	🟩 R	🟥 R		⬛ I
Artefact 3		🟨 D	🟩 R	🟥 R	⬛ I	
Test 1		🟨 D	🟥 X		🟨 I	
Test 2		🟩 D	🟨 D	🟥 X		⬛ I
Test 3		🟩 D	🟨 D	🟥 X		⬛ I

D = Draft
R = Review
X = Execute
I = Issue

🟩	Best practice
🟨	Ideal
🟥	High risk
⬛	Do not breach

Figure 5: The gated process

when, in order to meet a specific goal.

The Gated Process (figure 5) is a 2-dimensional representation of the project. Across the top of the matrix are the gates representing time sequence order. Down the page are the products and processes. The state that products/processes must be in by any particular gate is shown in the centre. The colour coding of the states indicates the criticality of that product/process state in meeting the business goal. Black indicates that if this process/product is not in the required state then the gate must slip because there is too high a risk to the project and business goals. Red indicates that these products/processes should only be missed after considering the risk implications. Yellow represents the realistic state to be in and Green indicates best practice.

For example, in figure 5, it is ideal to have “Test 2” drafted by gate 2, but it starts to have an impact if the test misses gate 3. If the test is not executed by gate 4 this will seriously breach a project goal and finally, if the test is not completed and issued by gate 6 then the project closure date will need to slip.

The implicit and probably the biggest value of this tool is that it encourages continuous and repeated communication between all stakeholders and gives visibility of a breach in the achievement of deliverables for a gate, leading to a slip of that gate. Every time multiple black colored boxes appear in the overview, there is indeed suddenly a "legitimate" and de-facto accepted reason for delay.

The Gated Process becomes a reference model for the best practice for a project, for meeting a predefined set of goals. There may be different models for different goals e.g. a shortest possible schedule would look different to the highest maturity product. Also the lowest short term cost solution would look very different to the lowest long term cost recipe.

Once defined, the project can use the Gated Process to plan their project by adding dates to the gates. The Gantt chart drops out easily from this matrix. Alternatively, a project can check a plan for its compliance to this Gated Process and derive a compliance scope indicating the level of risk to the project. We have found it most useful to turn the Gated Process into estimation, planning and tracking tool. This ensures that estimates plans and tracking are aligned and integrated.

During project execution, a project can track compliance against the Gated Process and can see, in advance, if there is a risk to a key milestone and if there is a risk to the project goals. At each gate, the project manager completes a spreadsheet indicating the actual maturity of each product/process e.g. if we expected 100 tests to be drafted at a gate and the team has delivered 80% then 80% of the earned maturity will be claimed. Figure 6 shows some example outputs. In this example, the project manager has indicated compliance at gate 3. This then turns into both a compliance chart and earned value (or earned maturity) chart shown below the matrix. In this example, we know there is a risk to the final project deliverables (gate 6) unless there is corrective action.

If the processes and products are weighted for value (e.g. cost), then the tool will report earned-value. At Rolls-Royce we also weighted the processes and products for their contribution to product maturity and therefore the Gated Process reported earned-maturity. The Gated Process will report any project goal so long as you can weigh the process and products for their contribution to your goal.

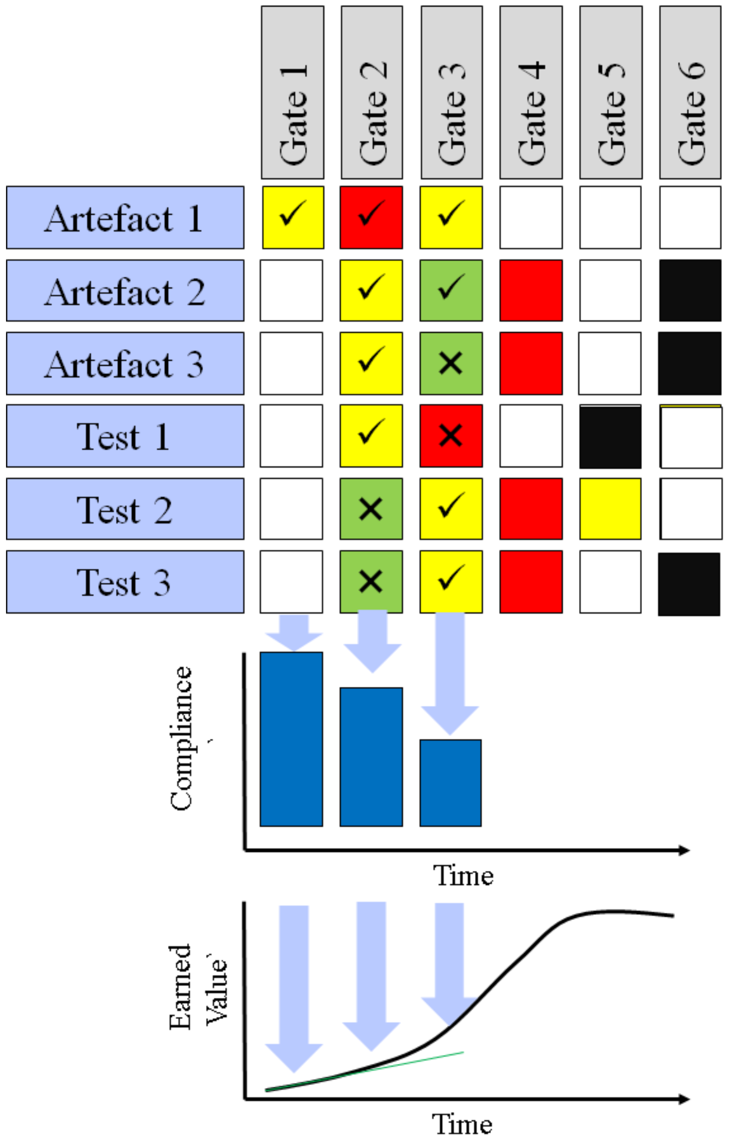


Figure 6. The Gated Process and Progress Reporting

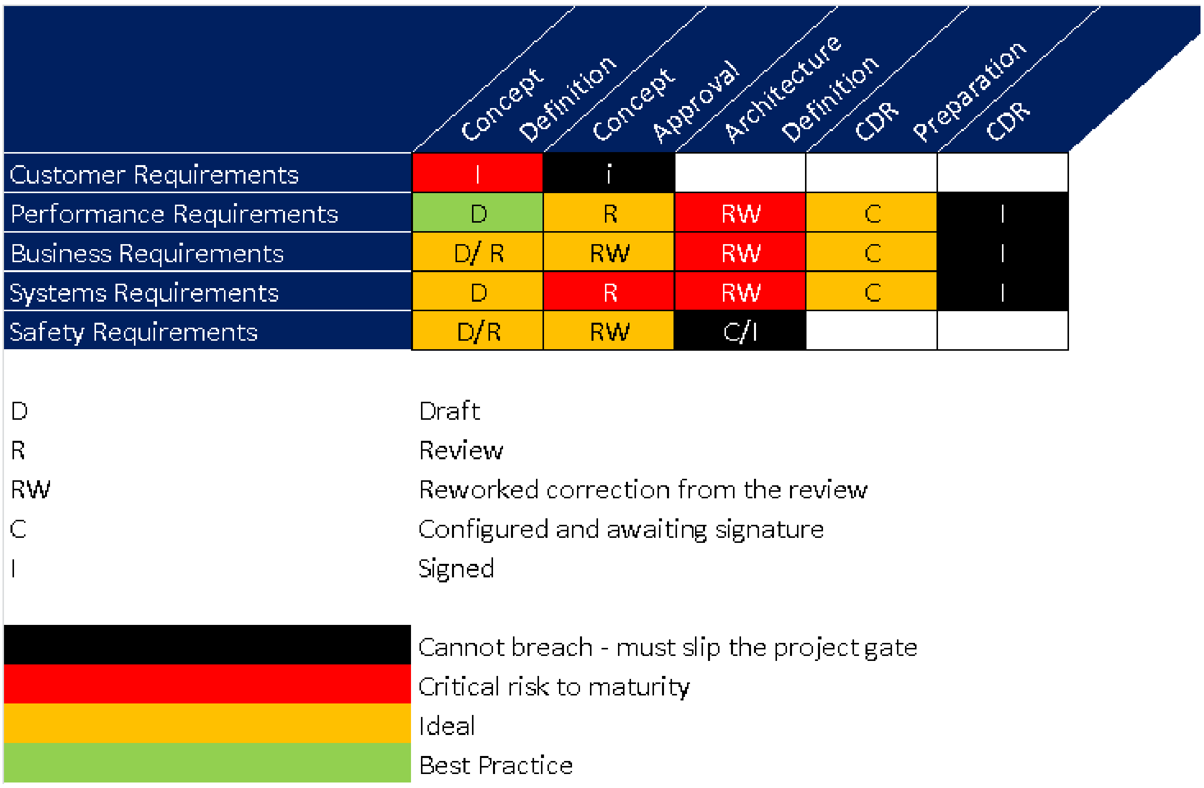
The Gated Process and the notation used allow a business to:

- Define the optimal process life-cycle (recipe or reference model)
- Generate plans and schedule from the reference model.
- Objectively track a project against this optimal life cycle.
- Provide an early warning of any risks to the project or business goals.
- Quantify any risks to the project and business goals if there is a slippage/delay.

- Provide a framework for root cause analysis.

Why we used the Gated Process

Many projects may track only a few key products (e.g. requirements issue or test completion) or a project will track progress at major milestones e.g. Critical Design Review (CDR) (ref: 4 - 6). Because of the richness of the Gated Process notation it is possible to easily track at a much finer granularity, and to track in a way to provide early warnings of any risks to critical project goals.



	Concept Definition	Concept Approval	Architecture Definition	CDR Preparation	CDR
Customer Requirements	🟥 I	⬛ i	⬜	⬜	⬜
Performance Requirements	🟩 D	🟧 R	🟥 RW	🟧 C	⬛ I
Business Requirements	🟧 D/ R	🟧 RW	🟥 RW	🟧 C	⬛ I
Systems Requirements	🟧 D	🟥 R	🟥 RW	🟧 C	⬛ I
Safety Requirements	🟧 D/R	🟧 RW	⬛ C/I	⬜	⬜

D	Draft
R	Review
RW	Reworked correction from the review
C	Configured and awaiting signature
I	Signed

⬛	Cannot breach - must slip the project gate
🟥	Critical risk to maturity
🟧	Ideal
🟩	Best Practice

Figure 7. Checkpoints and Project Gates

Rather than tracking a few critical gates e.g. CDR, you can define and track at more gates to de-risk the critical milestones e.g. Concept Definition, Concept Approval, Architecture Definition, CDR Preparation and CDR, as shown in figure 7. The earlier gates help spot risks to later gates e.g. if the project is non-compliant at "Architecture Definition" then the CDR gate is likely at risk. The additional gates give foresight to risks at achieving the critical gates.

Instead of tracking at an abstract level e.g. requirements, you can fragment the deliverables to reflect key dependencies and risk e.g. Customer Requirements, Performance Requirements, Business Requirements, Systems Requirements and Safety Requirements, as shown in figure 7. If there is a slip at the front, we can legitimately claim a slip to the project OR quantify the risk to the business and project goals.

Rather than tracking products and processes in their final completed state (e.g. issued requirements), you can track the various stages of the requirements development e.g. Draft, Review, Rework, Configure and Issue, again as shown in figure 7. This refinement allows for a finer tracking. If the requirements have not been reworked in time, then the requirements issue will be at risk.

Many organisations will have monitors around their critical processes and products but the Gated Process can track all products and processes to a high level of fidelity.

Where once we would have tracked a single product, "Requirements Issue", at a single gate e.g. CDR, we can now track at a finer granularity. In the example shown in figure 7 we have 5 states, 5 gates and 5 products = 125 definable and traceable items but expressed in only 5 lines on the table.

In addition, the Gated Process notation colours the gate-states in accordance with their value to meeting the business and project goals. This weighting can then be used to quantify risks to achieving the busies or project goals.

A typical Gated Process would be fragmented into 15 – 20 gates and around 100 products and processes and up to 4 levels of criticality. There would be between 350 and 450 gate-state transitions in the centre of the matrix which would take the equivalent of 350 – 450 lines on a Gantt chart. Despite this, the matrix will fit onto a single sheet of paper. This makes it very easy to view, see connections, share with customer and suppliers, and is easy to measure compliance.

The Gated Process has the following advantages:

- Where projects tend to only track a few critical products, the Gated Process can easily track all products and processes.
- Where typically projects will only monitor major deliverables at major milestones, the gated Process can track at a significantly higher fidelity according to where the risks lie.
- Where projects may not be able to show the outcome of their plans (time, cost and quality), by assessing plans against the reference model the project can "quantify" the compliance at achieving the project and business goals (or risk of not achieving).
- The notation is very easy to turn into a standard project schedule by simply adding dates to the gates.
- Most projects are very complex and the notation can express this in a simple way that anyone can understand – projects, engineers, customers and suppliers.
- Ifs possible to "quantify" the impact of any late inputs or delays, that could risk a milestone and there for the project goals.
- We have refined our models to monitor project inputs and dependencies. This enables us to quickly assess the impact of any delays at the front of the project.

- We are now able to define and track the maturity of products at key handover points. Immature products will inevitability lead to design iteration and scrap & rework.
- Where once we would use 10 – 20 discrete monitors to track a project, we can now represent all the charts in a single unifying monitor.
- By adding roles against each process or product, the Gated Process because a robust "contractual" definition of what has to happen and when.

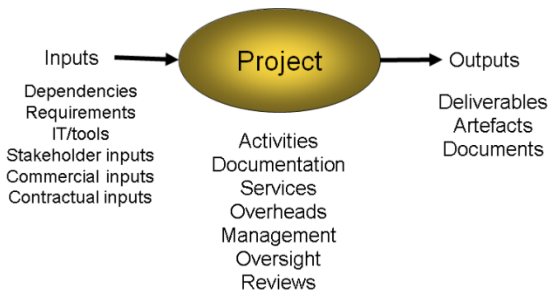
How to develop a gated process

Step 1: Understand the project goals

Understand the goals for the project in terms of time, cost and/or quality. The goals will be used to assign criticalities (see step 6).

Step 2: Understand the products and processes

Identify all the products and processes within the scope of the project. Also consider any dependence that need to be tracked. The granularity of the tracking should reflect the risk. Fragment the products and processes around the risky areas of your project to increase the fidelity of tracking.



Step 3: Identify states of products and processes

Understand the states that products and processes can be in as they mature e.g. S=Start, D=Draft, R=Reviewed, RW=Reworked, I=Issued. In the case of tests the states may be S=start, D=Draft, R=Run, Rv=Review, E=Formal Execution, C=Complete. Processes can also have states e.g. P=Planned, S=started, D=Documented, C=Complete. The more states you can give the more refined the model.

Step 4: Identify the Gates

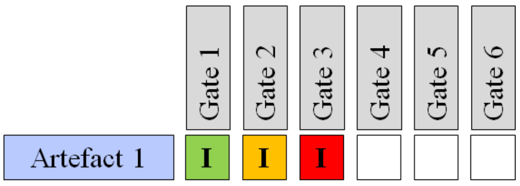
Fragment the project into a series of gates e.g. Business Approval, Project Start, Concept Review, Preliminary Design Review, Critical Design Review etc. Include any of your business and customer gates as necessary. Considerer adding review gates ahead of critical gates to provide early warning of any slippage. We tried to pick gates with an approximately linear earned value. For example, the jump between requirements issue and deliver for test could represent 50% of the value of a project. Instead, we would bridge this wide gap with sub-gates e.g. Architecture Review, Design Review, First Build, Bench Test etc.

Step 5: Construct the reference model

Create a 2-D matrix with the gates on one axis (in time sequence order) and products and processes on the other axis (in approximately time sequence order). For each product and activity, indicate the state it needs to be in at any gate. Note it is not necessary to mark every gate but only those which are meaningful.

Step 6: Assign criticality to the states

For each state transition, assign a colour for its importance at achieving the project goals. If necessary duplicate some states to indicate the changing in importance with each gate i.e. in the following figure, it is ideal (green) to issue "Product 1" at gate 1, required to be issued (orange) at gate 2 and a critical risk if it is not issues by gate 3 (red). We typically had 3 to 4 levels of criticality. In one instance, the criticality referred to the level in the Project organizational hierarchy you had to approach if you wanted to break that gate-state.



When scoring the criticality levels for each gate-state, we started by consulting the process experts for any supporting data. The initial definition may not be accurate but after each pass through the Gated Process, during Post Project Reviews (step 10) we continued to refine the reference model based on actual isses and escapes.

Often there is the opposite problem that the process experts tend to over-estimate the importance of gate-states. This has the effect of creating a rigid model. In these cases, we used a relative scoring method i.e. most gate-states are important, it's just that some are more important.

Step 7: Define the adherence measure and compliance rules

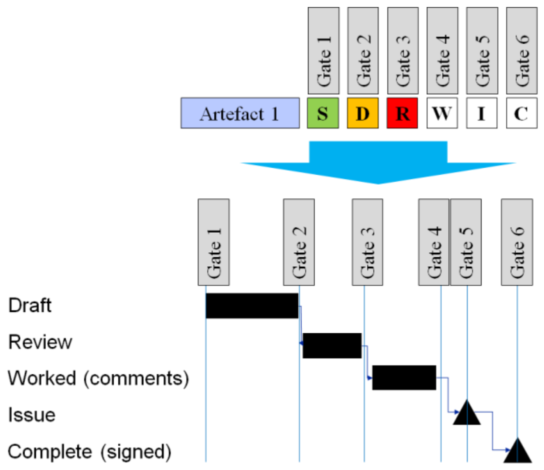
Define how you intend to measure adherence. Looking at any single gate, would you prefer to simply count the % of achievements e.g. 8 products were in the correct state but 2 were non-compliant therefore we have 80% compliance. Alternatively weight the products according to value (e.g. effort) or weight according to criticality. We tended to do both.

Define the rules for gate pass such that if a project falls short, the gate-date will have to be slipped. Examples include:

- 95% or greater products and processes are in the correct state (bean count).
- 95% or greater "value" earned.
- 95% or greater of risk mitigated (remaining work will not affect risk).

Step 8: Develop project plans and schedules

Project plans and schedules can be generated quickly by assigning dates to the gates. In this example, you can see how a Gantt chart can be generated from the Gated Process. Over time we developed rules for the ratio of time between gates e.g. if the time between gate 1 and 2 is 2 weeks then the time between gate 2 and gate 3 needs to be 4 weeks.



Step 9: Gate reviews

At the date assigned to the gate, you will perform a gate review. The purpose of a gate review is to provide a formal mechanism to assess adherence to the Gated Process and identify risks to the project.

For the gate under review, seek evidence of the state of the artefacts relative to the planned state. Collate the scores together to create a single adherence score for the gate. Assess the overall risk to the project and take corrective action.

Step 10: Post Project Review

At post project reviews, identify any emergent issues and trace them to the Gated Process. Look for ways to refine the Gated Process to reduce the likelihood of issues in the future. Typically this means either adding additional products (or fragmenting products around areas of risk) or increasing the criticality of gate-states.

Also, track the adherence measures over time to see if there are common problems. This led to an improved planning process and additional gates around troublesome phases of the project.

Experiences using the Gated Process

These are a few of the experiences we had using the Gated Process:

- The Gated Process is project management for non-project managers. It does not require project management skills or project tools but is expressed in a language that the team, customer and suppliers could understand. We issued a copy of the Gated Process for every member of the team.
- We had been tracking scrap and rework escaping software packages of work. In general, for a non-Gated Process project, we had measured an average of 50% scrap and rework (see ref 2). For projects that complied to the Gated Process, the Scrap and Rework rate dropped to 25%. This is because products were matured at the right time using the right processes. As figures 3 and 4 showed, earlier detection of errors reduce the cost of correction.

- In additional to scrap and rework that escape a package of work, there was a reduction in internal design iterations during development. In one case, the cost of a software package of work dropped by 45% through compliance to the Gated Process. This happened because the previous manager, desperate to make progress, would encourage the handover of immature products between teams. Once we set the maturity targets (the gate-state criticality), hand over became better and overall costs reduced. By slowing the project down, the net effect was to reduce costs by 45% with no impact to schedule – slow down in order to speed up!
- Over a number of packages of work we noticed that the projects tended to struggle to comply with the Gated Process mid-project e.g. the start and end of the projects had good compliance but we struggled towards the middle of the project. This meant that we were breaching some of the critical gates. This led us to develop rules for gate durations (time between gates) to ensure we did not over-stress the project.
- The Gated Process is very deterministic assuming the work being performed is relatively deterministic. If there was too much risk/uncertainty in the work, this would lead to gate slippages (or poor gate compliance) which in turn impacted the project goals. We added a filter at the start of the Gated Process to remove any high risk work items and working these asynchronously to the Gated Process using agile development methods. The work would then be re-introduced into a “production” build once its maturity had achieved a defined level.
- Given the granularity of the Gated Process, it was possible to turn the tool into estimation, planning and tracking tool. For each process and product, we created a parametric cost estimation equation and assigned a resource group. It was then possible to quickly generate the estimate of schedule, cost and resource needed for each instance of the Gated Process. Monitoring was automated using check lists and the tool would generate the necessary earned-value or earned-maturity charts.
- When analysing issues at post project revise, we observed that almost all issues could be traced to con-compliance to the Gated Process. The issues helped us define the criticality of the gate-states to reduce the possibility of the problem reoccurring. It also helps re-enforce why we needed the Gated Process and why we needed compliance.
- Breaking the states down into sub states helped create pragmatic recipes. It is tempting to insist on requirements being formally issued before starting work. This is not ideal for many reasons. But the new state of “rework” meant the designers and testers could work off requirements that were about 90% mature at a much earlier date. This meant we could reduce schedules, work concurrently but manage the maturity of products at handover. If schedule

was not critical we would have worked with 100% mature items at handover but this would create long schedules. The 90% maturity value led to more pragmatic schedules with acceptable and accepted level of scrap & rework.

Conclusions

A project's recipe will have a great effect on the project's success at meeting any goals of cost, quality or schedule. However, many project managers will develop unhealthy recipes as they desperately attempt to recover a struggling project. It was necessary to define a way to express the idea recipe and to "quantify" projects for compliance to it.

The Gated Process is a very simple way to define the ideal life-cycle for your project. Ideal is defined here as meeting your business goals, be it cost, schedule, quality or a combination of all three. The structure notation enables a project to express complex projects in a simple way to enable easy planning and tracking.

We developed our Gated Process to deliver the highest maturity product in a "reasonable" timescale. Short term costs were sacrificed in favour of long term project cost reduction. The colour coding for us was biased around breaches to product maturity. This had the effect of reducing scrap and rework from 50% to 25%.

In one instance, we had a cost reduction with no impact to schedule. This was because before the introduction of the Gated Process, concurrent working meant that most teams worked at risk, thus increasing intra-package design iterations. By introducing the Gated Process, slowing down the front of the project, we were able to maintain the project schedule whilst reducing project costs by 45%.

The Gated Process enabled us to embed, enforce, track and control compliance to an ideal life-cycle. The notation was simple, easy to understand, readily available to all tam members (customers and suppliers). It was easy to monitor and quantify risk.

The notation was rich, providing a high level of visibility and control and yet an entire project could fit onto a single sheet of paper. In many ways, the Gated Process is easier to read then a Gantt chart.

We have found it most useful to turn the Gated Process into an estimation, planning and tracking tool. This ensures that estimates, plans, resources and tracking are integrated into a single tool. This meant that all aspect of the project management pivoted around achieving the project goals.

References

1. Forsberg, K., Mooz, H. and Cotterman, H. "Visualizing Project Management", 3rd ed. New York, NY, USA: J. Wiley & Sons, 2005
2. Nolan, A.J. and Pickard, A.C. "Reducing Scrap and Rework", 23rd INCOSE International Symposium, Philadelphia, PA, 2013

3. Pickard, A.C. and Nolan, A.J. "How Cost Effective is Your V&V?" 23rd INCOSE International Symposium, Philadelphia, PA, 2013

4. US Department of Defense "Technical Reviews and Audits for Systems, Equipment and Computer Software", MIL-STD-1521, 1985

5. Chao, L.P. and Ishii, K. "Design Process Error-Proofing; Benchmarking Gate and Phased Review Life-Cycle Models", DETC2005-84235, Proceedings of the ASME 2005 International Design Engineering Technical Conferences and Computers and Information Engineering Conference, Long Beach, CA, 2005

6. Cooper, R.G. "Doing it Right; Winning with New Products", Ivey Business Journal, July/August 2000. Also see www.stage-gate.com.

Biography

Andrew Nolan joined Rolls-Royce in 1989 after completing a degree at Sheffield University. He is the

Chief of Software Improvement for Rolls-Royce based in the UK. He is a Fellow of the British Computer Society and a chartered Engineer in software engineering. Andrew has spent over a decade managing large scale software projects as well as a decade improving the way Rolls-Royce manages projects.

Andrew Pickard joined Rolls-Royce in 1977 after completing a Ph.D. at Cambridge University in Fatigue and Fracture of Metals and Alloys. He is a Rolls-Royce Associate Fellow in System Engineering, a Fellow of the Institute of Materials, Minerals and Mining, a Chartered Engineer and a member of SAE International and of INCOSE. He is Vice-Chair of the SAE Aerospace Council and represents Rolls-Royce on the INCOSE Corporate Advisory Board.